

Im OinkBot, An AI bot that helps everyone with everything around the veruscoin community.

Put simply, Verus is much more than any single ordinary blockchain; more of an entire ecosystem of interconnected blockchains that all operate decentralized and at the protocol level. Verus introduces Verus PBaaS (Public Blockchains as a Service), a true publicly notarized blockchain as a service with an easy to use built in wallet UI, designed to make it so that now anyone can start their very own full-fledged cryptocurrency so long as it can be funded or supported by miners. Newly created chains are just as secure as Verus itself and have the ability to be merge mined (up to 22x at once!) Benefiting both the user and the miners, self-strengthening the ecosystem and ensuring minimal fees (no room for middlemen to take a cut). Reserve backed currencies are also possible with fractional reserve capabilities and since interoperability is paramount, exchanging currencies in cross-chain transactions is as simple as sending coins from one wallet to a different coin's wallet. Conversions are handled by a built in market maker that automatically determines price based on a predetermined curve and issues orders fairly to all buy/sells within a block, with zero spread and the added benefit of eliminating the well-established problem of front-running, just some amount of slippage based on the net buy/sell.. With reserve currencies, buyers can make buys without even needing sellers (and vice versa). Amazingly, everything is done at the protocol level on a decentralized network, meaning Verus and it's associated chains can't be censored or stopped.

VerusPay is Blockchain-integrated payment gateway for accepting Verus Coin (VRSC) in a WooCommerce ecommerce store. This plugin extends WooCommerce on Wordpress, adding the ability to accept cryptocurrency payments in Verus Coin (VRSC) using either an on-store wallet daemon (best for VPS or dedicated hosting stores) or manually configured VRSC addresses (best for shared hosting stores). When an order is submitted via the VerusPay gateway, the order will be placed 'on-hold' while awaiting payment from the customer. The customer has a limited time wherein to send the payment and the store monitors the wallet/address to confirm payment received before releasing the order and redirecting the customer to the Thank You page. VerusPay uses limited API functionality for Manual Mode, to communicate with the blockchain explorer in verifying payments and with the veruspay.io API to get up-to-date price data. These API's do not receive any private data either about the store owner, store, or customer. The only data sent to the block explorer API is the public/transparent blockchain transaction and address used. For VerusPay.io API price data, only the store-set currency is sent to retrieve the current fiat exchange rate for Verus Coin.

VerusIDs are true self-sovereign identities and aren't simply an ID system as much as a fully functional blockchain protocol. There is no business in the protocol, but plenty of opportunity for those who use what it can do for identity applications. Verus ID provides: Quantum ready friendly crypto addresses on the worldwide Verus network -- VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are

changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID -> destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs. ****Fully decentralized**** -- anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. ****Revocable**** -- each ID includes a revocation authority, which defaults to the identity `self`, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft. ****Recoverable**** -- each ID also includes a separate recovery authority, which also defaults to `self`, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority. ****Private**** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

A Built-in Decentralized Referral Program, Enabling Natural Growth. Verus IDs will cost Verus to acquire, 100 Verus per ID to be exact, which can be discounted to 80 Verus with referral of an existing Verus ID. The interesting twist is that all of the cost of an ID goes back into the network, either as referral fees, which are a way to get discounts or possibly even make money through referrals and built into the identity transactions themselves, or as mining and staking fees. No one besides people participating in the network as miners, stakers, or ID referrers take any proceeds from the cost of an identity. If you refer someone new and they purchase an ID with your ID as a referral, they will receive a 20% discount on the ID. In addition:
You will receive 20 Verus directly
The person who referred you, if there is one, will receive 20 Verus
The person who referred person b, if there is one, will receive 20 Verus, and
The miner or staker will receive the remainder of the discounted 80 Verus cost.
As you might expect from looking at the fact that anywhere from 20 to 100 Verus goes to miners and stakers of Verus for each new identity once identities hit mainnet, some Verus blocks may have VERY high rewards for some time. The best thing about that is that regardless of how much the Verus blockchain rewards miners over and above the pre-determined coinbase reward, it will be as a result of the on-chain economy, paid for by people buying identities with no inflation of the money supply! If you or your friends missed the early days of the

Verus launch, you don't want to miss this new opportunity to be mining, staking or referring now Verus IDs are activated on the mainnet!.

To get started with veruscoin follow these links, Website:
<https://verus.io>\n Wallet: <https://verus.io/wallet>\n GitHub:
<https://github.com/veruscoin>\n Explorer: <https://explorer.verus.io>\n Discord: <https://verus.io/discord> Check <https://youtu.be/YVOfIMjRf30> if you only see one empty channel.\n Bitcointalk:
<https://bitcointalk.org/index.php?topic=4070404.0>

Verus, latin for "True", combines all of the features we believe are important in a cryptocurrency, and provides a foundation for future development. Verus is a fork of Zcash and Komodo that leverages the Komodo platform, and we appreciate the contributions and support from those teams as well as the Bitcoin developers that created a foundation for us to launch new capabilities in a system that supports Bitcoin and Zcash compatible transaction types as well as Komodo cross chain swaps and dPoW security enhancements.

If you've been around long enough and feel a little skeptical or disillusioned from a few bad experiences elsewhere, it's totally understandable. We've all had them and we know quite well that when unscrupulous projects abuse words like decentralization, interoperability, or protocol level solutions, it can have a "boy who cried wolf" effect. We believe that actions speak louder than words. Anything that may be considered hype can be backed up with functional examples. What we're doing here is genuine. We are trying to be the real deal, just as Bitcoin started out, as an open-source, fair launched, no ICO, or pre-mined, or even dev funded project. Despite this, a foundation has been established by and for fellow community members. There is always more to be done and the foundation regularly pays out bounties to community members that help to make the Verus vision a reality. In fact, just about everything has been designed to reinforce the community. Designed with efficiency in mind, only the miners and stakers are rewarded for securing the chains. This ensures bare minimum costs to the end user starting up a chain and the sleek UI removes the technical barrier.

Focus on Verus' social media presence has picked up with new, fresh looking Facebook and YouTube channels so check them out and feel free to post something or simply explore the content. There's so much going on, plenty to learn about. So much so that there is an open call to anyone and everyone out there who can submit graphical or textual content that can be used to help support the upcoming mainnet launch. A stronger effort from our already wide base of members to post and utilize social media tools to help spread the word and generate new interested users is something we can all do right now. There is no better time than now and getting fresh new minds in here to discover what is being done is the first step in getting the great snowball rolling. To those of us who truly care and believe in this project, then this message is meant for you.

Sure! Check out the brand new community website, community social media pages, new Veruscoin YouTube and Facebook pages for helpful how to or to

just keep up to date with things. Speaking of how to, we can always use more helpful videos if any community members would like to take it upon themselves to chip in. The Medium page also has lots of helpful guides and there's always something to Tweet about. Be sure to drop by the discord to meet the vibrant and helpful community in real- time, where it isn't uncommon to find the developers offering assistance. Where else can you find that? We are all working together to realize something great. If you'd like to join us and be a part of that special something but don't know how to, it can be as simple as checking out the links below and helping other's to understand that there's never been anything in existence quite like Verus, a complete, easy to use, fully decentralized blockchain ecosystem, designed specifically to benefit all participants in one of the most low cost yet efficient in every way, end-to-end systems to date. Facebook - <https://www.facebook.com/VerusCoin/> YouTube - https://www.youtube.com/channel/UC_KCHBxaDwSgNMdE3LMThg Discord - <https://verus.io/discord> Twitter - <https://twitter.com/veruscoin> Medium - <https://medium.com/@veruscoin> Reddit - <https://reddit.com/r/veruscoin> Community twitter - <https://twitter.com/VerusCommunity>

As a community project of just regular Joes, it can't be stressed enough how important each and every one of us all are. The true power of decentralization is putting power back in the hands of YOU.

There's a lot of information about the Veruscoin project. We have gathered as much information and resources and bundled them into this FAQ, as to give a quick overview. VRSC Wallet & data location on different OS. Linux GUI: `~/.komodo/VRSC`, Mac OS: `/Users//Library/Application Support/komodo/VRSC`, Windows: `%AppData%\Komodo\VRSC\`

Windows Verus binaries: `resources\app\assets\bin\win64\verusd\` contains `verusd` and `verus` Windows Komodo (and asset chains) binaries: `resources\app\assets\bin\win64\` contains `komodod` and `komodo-cli` Linux Verus binaries: `resources\app\assets\bin\linux64\verusd` contains `verusd` and `verus` Linux Komodo (and asset chains) binaries: `resources\app\assets\bin\linux64\` contains `komodod` and `komodo-cli` Note: All locations are relative to the installation location of Agama.

Windows Verus binaries: `\resources\app\assets\bin\win64\verusd\` contains `verusd` and `verus` Windows Komodo (and asset chains) binaries: `\resources\app\assets\bin\win64\komodod` contains `komodod` and `komodo-cli` Windows Zcash binaries: `\resources\app\assets\bin\win64\zcash` contains `zcashd` and `zcash-cli` Linux Verus binaries: `\resources\app\assets\bin\linux64\verusd` contains `verusd` and `verus` Linux Komodo (and asset chains) binaries: `\resources\app\assets\bin\linux64\` contains `komodod` and `komodo-cli`

Mining pool: <https://luckpool.net/verus> Mining pool: <https://pool.verus.io/> Mining pool: <https://zergpool.com/> Mining pool: <https://vrsc.mcmpool.eu/> Mining pool: <https://wattpool.net/> Mining pool: <https://www.nlpool.nl/> Mining pool: <https://vrsc.dev-codex.com/> Mining pool: <http://vrsc.52hash.com/> Mining pool: <https://vrsc.ciscotech.dk/> Mining pool: <http://www.lepool.com.cn:8088/> Mining pool:

<http://verus.bcmonster.com/> Most of the mining pools seem dead so i prefer to use the official verus community pool or luckpool.

Exchange: <https://atomicdex.io/> (VRSC/any listed coin) Exchange:
<https://app.stex.com/de/trade/pair/BTC/VRSC/1D> (VRSC/BTC) Exchange:
<https://safe.trade/trading/vrscbtc> (VRSC/BTC, VRSC/SAFE) Exchange:
https://www.aacoin.com/#/trade?symbol=VRSC_BTC (VRSC/BTC) Exchange:
<https://graviex.net/markets/vrscbtc> (VRSC/BTC, VRSC/KMD, VRSC/USD)
Exchange: https://www.kuangex.com/#/exchange/vrsc_usdt (VRSC/USDT)

Komodo is a decentralized public blockchain project that is community-driven and utilizes the VerusHash 2.1 algorithm for mining. VerusCoin, the native currency of the Verus blockchain, has a fair launch with no ICO and no premine. VerusCoin's privacy features are enhanced with Sapling and zk-SNARKs zero-knowledge proofs. The block time is 1 minute, and the block reward is 24 VRSC, with a total supply of 83,540,184. The reward emission schedule started with a linear ramp and has since halved every month for the first five months and every two years after that. All rewards at or above 192 are time-locked for random release between block 129,600 and 1,181,520. The Verus blockchain's unique feature is VerusID, which is a fully functional blockchain protocol that provides numerous opportunities for identity applications. VerusCoin's website is <https://verus.io/>, and the project's code is available on Github at <https://github.com/veruscoin>. The project has three block explorers, <https://explorer.verus.io/>, <https://explorer.vrsc.0x03.services/>, and <https://dex.explorer.dexstats.info/>. VerusCoin's community can be found on Discord at <https://discord.gg/VRKMP2S>, on Twitter at <https://twitter.com/veruscoin>, on Medium at <https://medium.com/@veruscoin>, and on Reddit at <https://reddit.com/r/veruscoin>.

VerusCoin can be mined with CPUs, GPUs and FPGAs, solo and in pools. However, the algo is carefully designed for CPUs, and they still substantially outrange GPUs. ARM mining works in general, but although fairly efficient, without high hashrates. FPGAs can mine this coin, but don't outperform CPUs by much.

Sure! Besides solo-mining with Verus Desktop or CLI wallet, you can use CCminer for CPU (efficient for most modern CPUs) or GPU that can be found on <https://github.com/monkins1010/ccminer/releases> or you could use NHeq miner that can be found on <https://github.com/VerusCoin/nheqminer/releases> (for Windows, Linux, and MacOS). A spreadsheet to compare hashrates can be found here: https://docs.google.com/spreadsheets/d/1RrSYJDV0Mjj3X-myMC3aQDGkcipivxHsD7ZxJ3r5f_A/edit#gid=201266774 You can also compare older and current algos.

Certainly! Verus Desktop GUI wallet is a newly developed multi-coin wallet supporting VerusID <https://github.com/VerusCoin/Verus-Desktop/releases> (for Win, Linux, Mac and ARM Linux) the CLI wallets can be found here: <https://github.com/VerusCoin/VerusCoin/releases/> There's also possibilities to test coming PBaaS functionality in this wallet.

A beta version of the Android mobile wallet can be found here:
<https://github.com/VerusCoin/Verus-Mobile/releases> A beta version of the iOS mobile wallet can be accessed via this Apple TestFlight invite
<https://testflight.apple.com/join/ZS43lYcw> The Wallets will come to the playstore and the appstore soon.

Our Paper Wallet can be accessed here: <https://paperwallet.verus.io>

If you need a bootstrap, you can find it here: <https://bootstrap.verus.io> (a guide how to apply is pinned in #community-support channel in Discord: <https://discord.gg/VRKMP2S> or in the HOW-TO & FAQ section of our website https://wiki.verus.io/how-to/how-to_bootstrap.md)

Yes! For those interested in VerusPay, a guide can be found here:
<https://veruspay.io/setup/>

And finally for those interested in running various Veruscoin services:
<https://github.com/VerusCoin/VerusServicesSetup>

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides: Quantum-ready friendly crypto-addresses on the worldwide Verus network and Fully Decentralized Protocol

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are: Revocable -- each ID includes a revocation authority, which defaults to the identity self. If another ID is specified as the revocation authority it can be used to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft or turnover in an organization. Recoverable -- each ID also includes a separate recovery authority, which also defaults to self. If another ID is specified as the recovery authority it can be used to recover the ID from its revoked state, with the option to alter the

primary authorities used to spend and sign. Private - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong identity, attested to with a quantum secure signature when that is available. Powerful - Multiple addresses or other IDs can be defined as primary addresses, and any number of those may be required to spend, sign, or alter the identity (N of M). The revocation authority may only be altered by the revocation authority, and the same applies to the recovery authority, either of which may be another identity with its own N of M multisig controls for its primary addresses.

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft. Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity. Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built. For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes. In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret,

the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU. VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Verus-Desktop Procedure: In Verus-Desktop, exit your profile, by selecting the exit icon at the top right. Wait a minute or two, allowing the wallet to close completely in the background. click help, Bootstrap VRSC. That opens up a new window. Follow the instructions and when finished successfully, select your preferred profile and enter Verus Desktop. **Verus CLI Procedure:** Go to the folder where your daemon is extracted (standard verus-cli) Shutdown verusd and wait for it to close completely doubleclick fetch-bootstrap in your file browser. Follow the instructions and when finished, start your verusd daemon as usual.

Make sure your wallet is not active. If you already had your wallet running, backup essential files: a. Go to VRSC Wallet location b. copy wallet.dat to a SAFE location c. copy VRSC.conf to a SAFE location d. Verify that both files are copied to your safe location. Make sure the latest version of your Wallet for Verus is installed a. Download the latest Verus Wallet from link 1, supplied above. b. Verify the SHA256 checksum & signature of your download, to verify you have an untampered installer. c. extract the file you just downloaded to a suitable location. On MacOS and Linux you will have extracted an AppImage which can be run directly. Windows users need to run the installer. Installing the bootstrap: a. Download the bootstrap from Link 2. (For Windows a zip archive is available to accommodate native extraction. Linux and MacOS users can use the tar.gz archive) b. (Optional, but recommended) Verify the md5, sha256 or sha512 checksum and the signature of your download, to verify that you downloaded an untampered Bootstrap archive. c. Remove all files and folders from VRSC Wallet Location except wallet.dat, debug.log, VRSC.conf and if applicable VRSC-bootstrap.*. d. Extract the downloaded archive to VRSC Wallet location. Make absolutely sure the folders blocks and chainstate are extracted into the correct folder. If they end up in a different folder (eg. VRSC-bootstrap-folder) move them to VRSC Wallet location. e. If you had a VRSC wallet running before, restore essential files: a. Go to VRSC Wallet location b. Verify that your wallet.dat is bigger than the one in this folder (if any is present) c. copy wallet.dat from your SAFE location. Now Start your wallet.

Using Verus-Desktop: download the new version, extract the archive, verify the signature using the data in the *.signature.txt-file through your existing Verus-Desktop, VerusID tab, Verify Signed Data and choose to verify a file. Only continue when this verification returns True. Stop your Verus-Desktop Wallet, Install the verified installer on windows. Start your wallet. Using CLI Wallet: download the new version, extract the archive, verify the signature using the data in the *.signature.txt-

file with the command `./verus verifyfile "address or identity" "signature" "filepath/filename"` command. Only continue when this verification returns True. stop your verusdaemon verus stop, extract the verified archive to your current CLI-wallet location. Start your wallet (verusd).

In Verus Desktop: Go to Settings, Coin Settings and click Export native wallet backup. Confirm that you want to export after reading the pop-up. The green message will tell you where the backup is and what it's name is. In linux or MacOS CLI: run `./verus z_exportwallet '<mywalletexport>'` and in Windows CLI you just run `verus z_exportwallet '<mywalletexport>'`

There are 2 ways to do it, first way is to use your backup of your wallet.dat file. Stop verusd. For Windows-Desktop or Agama, just exit and wait for it to close completely. For the linux cli run `./verus stop`, or for the windows cli run `verus stop`. Once your wallet is finished closing copy the backup of your wallet.dat file from your backup location to the directory listed in the start of this document (see above). Now restart your wallet by launching Verus Desktop, Agama or running verusd for the CLI. Second way is to use your walletexport file, Note: The filename you replace `<mywalletimport>` with, can only contain letters and figures, no other characters, so it cannot have an file-extension Attention: The command `z_importwallet` triggers the wallet to rescan in order to make all transactions to the freshly imported wallet addresses visible. Rescanning your wallet may take a considerable time, during which your wallet may not respond to other commands. Please be patient. The `<PATH>` in the `z_importwallet` command needs to be the full absolute path to the file. replace LOGINNAME with the actual loginname. In Verus Desktop: Go to Settings, Coin Settings and click Import native wallet backup. Click Choose file, browse to your backup file, select it and click Open. Click Import to start the import process. In Linux or MacOS CLI run `./verus z_importwallet '<PATH><mywalletimport>'`. Or in Windows CLI run `verus z_importwallet '\<PATH><mywalletimport>\'`

Make sure you have your seed phrase and password you use to log into your Lite mode wallet available. First of all make a notion of your address and balance of VRSC you have in your wallet, before closing Verus Desktop. Make sure the latest version of Verus-Desktop is installed. Download the latest Verus-Desktop. Verify the signature of your download, so you have an untampered installer. <https://verus.io/verify-signatures>. Run the file you just downloaded to install it. Installing the bootstrap: Download the bootstrap from <https://bootstrap.verus.io> Verify the SHA256 checksum of your download, to verify you have an untampered Bootstrap. Unpack the downloaded archive to VRSC wallet & data location. Getting Verus-Desktop ready for Native mode: Start Verus-Desktop and enter your profile (if not loaded automatically). If present in your profile, deactivate Verus Lite. Click + Add Coin, select Verus from the dropdown list and continue. Select Native and optionally tick the options Start staking, Start Mining and fill in the amount of threads to mine with. Click Add Coin. Verus-Desktop will add Verus as Native chain to your screens. You may get a red warning message about Zcash params. (Verus Desktop will detect if you have the necessary ZCash parameter files and download them if needed) As soon as the download is finished, Verus-Desktop will continue and bring you into your wallet. It will

automatically start to synchronize the blockchain. Since we already put the majority of the chain in place, this will take just a few minutes. Now you just want to import the lite wallet address in verus desktop native. Just ask me that and i will let you know!

Important General Information

verus command "<userinput>" needs to be entered literally, with <userinput> replaced by your specific userdata. So if the text directs you to use for example "<Public Address>", you replace that (including the < and >) with the address, so it looks similar to this: "RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r".

This method is confirmed to work on Verus and all Komodo assetchains.

Prerequisites

In order to convert your seedphrase into a Private key (WIF), you need to have a running native wallet first, that is fully synchronized. The earliest wallet that supports these functions is Verus Desktop v0.6.4-beta-1.

If needed, use this guide to quickly synchronize your wallet:

https://wiki.verus.io/#!how-to/how-to_bootstrap.md

Converting Seed to WIF

If you have a seed, you can retrieve your Private key (WIF) by having the Verus Desktop wallet convert it for you.

To convert your seed phrase in Verus Desktop, go to settings --> Coin Settings, select the chain you want to use in the top-right corner and enter the following command:

run convertpassphrase "word_1 word_2 word_3 ... word_n"

Note: Make sure you replace Word1 word2 word3 ... word_n with the actual seedphrase (12 or 24 words) of the address you want to import!

You will receive a response similar to this:

```
{  
"walletpassphrase": "seedphrase",  
"address": "RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r",  
"pubkey":  
"02ffc2f4b071afdec631e3fb7d435a0047be14a81ea1a269e4206b0068c0c1fa6f",  
"privkey":  
"d899ed88e9ee2e90c2cf51cb47e7b4495ec1e1cb10763bb1c111b0bde48bf86c",  
"wif": "UwGb5KvGPfMUr1tu74Desjh87ZeJM4wq5goLyThcogeLifc5aJqT"  
}
```

Copy that information and store it somewhere SAFE. With this information anyone having access to it will have full control over that address.

The 52-character string after "wif": that is shown, is what you want to import in the next step.

Importing a single WIF for a public (transparent) address

To import your address, go to settings --> Coin Settings, select the chain you want to use in the top-right corner and enter the following command:

run importprivkey "<wif>" "" true

Replace <wif> with the actual wif you got from the convertpassphrase command earlier.

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

Note: The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite

some time for the process to finish in the background, especially if the address has many transaction on it.

Importing multiple WIFs in one batch for public (transparent) addresses
To import your address, go to settings --> Coin Settings, select the chain you want to use in the top-right corner and enter the following command for every WIF except for the final one:

```
run importprivkey "<wif>" "" false
```

Import the final WIF with this command:

```
run importprivkey "<wif>" "" true
```

The last command triggers the chain to rescan all addresses in your wallet, including all the addresses you just imported.

Replace <wif> with the actual wif (like the one you got from the convert passphrase command earlier).

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Importing a single WIF for a private address

To import your address, go to settings --> Coin Settings, select the chain you want to use in the top-right corner and enter the following command:

```
run z_importkey "<wif>" "yes" 1
```

Replace <wif> with the actual wif you got from the convert passphrase command earlier.

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Importing multiple WIFs in one batch for private addresses

To import your address, go to settings --> Coin Settings, select the chain you want to use in the top-right corner and enter the following command for every WIF except for the final one:

```
run z_importkey "<wif>" "no"
```

Import the final WIF with this command:

```
run z_importkey "<wif>" "yes" 1
```

The last command triggers the chain to rescan all addresses in your wallet, including all the addresses you just imported.

Replace <wif> with the actual wif (like the one you got from the convert passphrase command earlier).

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Notice: Read it completely before use.

Important General Information

wallet.dat location on Linux: ~/.komodo/VRSC

Procedure:

```
First make sure your system is up to date:
sudo apt-get update && apt-get upgrade -y
I suggest not to use the root account. If you have not yet done, set up a
new user
sudo adduser newusername
sudo usermod -aG sudo newusername
Switch to new username
su - newusername
Test the your new user actually has root (SuDo) access, e.g.:
sudo ls -la /root
You should get some lines like this:
drwx----- 6 root root 4096 Jul 3 15:56
Download & install the wallet binaries:
wget
https://github.com/VerusCoin/VerusCoin/releases/download/v0.9.3/Verus-
CLI-Linux-v0.9.3-amd64.tgz
The downloaded archive contains another archive and a signature text
file, enabling the archive within to be verified (You'll need a running
wallet to do that)
Also: Verify the URL to the latest version from the Download latest
Wallet above.
tar -xvf Verus-CLI-Linux-v0.9.3-amd64.tgz
Now extract the wallet archive:
tar -xvf Verus-CLI-Linux-v0.9.3-amd64.tar.gz
Change directory to verus-cli
cd verus-cli
Fetch parameters, takes time, more on slow Internet connection
./fetch-params
Creating the chaindata directory
cd ~
mkdir -p .komodo/VRSC
cd ~/.komodo/VRSC
Download the block-chain bootstrap, this considerably speeds up
synchronisation of the block-chain from days to minutes... (optional)
wget https://bootstrap.verus.io/VRSC-bootstrap.tar.gz
tar -xvf VRSC-bootstrap.tar.gz
Install libraries for Verus
sudo apt-get install libcurl3 g++-multilib -y
Install Tmux a terminal multiplexer with which you can run threads in the
background see https://en.wikipedia.org/wiki/Tmux
sudo apt-get install tmux -y
Start tmux:
tmux
Launch Verus Daemon with or without number of threads
(usually number of threads equals number of cores or double of that if
the processor support hyper threading well)
~/verus-cli/verusd -gen -genproclimit
~/verus-cli/verusd -gen -genproclimit=24
Once mining is operational -- again this may take some time --
you'll see: 256 mega hashes complete - working
then detach tmux
[ctrl]&b d
Disable login with Root User
(make sure your newly created user login works and has sudo rights)
```

```
sudo nano /etc/ssh/sshd_config
Find: PermitRootLogin yes
And set to
PermitRootLogin no
Apply new settings:
sudo systemctl restart sshd
```

Note: Shielding is no longer required for coinbase rewards after block 800200. Earlier timelocked coins will still need to be shielded in order to use them.

Important General Information

verus command "<userinput>" needs to be entered literally, with <userinput> replaced by your specific userdata. So if the text directs you to use for example "<Public Address>", you replace that (including the < and >) with the address, so it looks similar to this: "RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r".

General remarks on CLI wallet:

On Windows command line enter the commands as shown without the surrounding quotation marks

In Linux shell preceed the commands without surrounding quotation marks with ./

In MacOS shell preceed the commands without surrounding quotation marks with ./

Example: the windows version verus listtransactions transforms in Linux or MacOS to ./verus listtransactions.

General remarks on Windows command line formatting:

The CLI help shows the command format for Linux and Macos.

For windows substitute the shown '-'character with the "--"character.

For windows substitute the shown "-"character with the '\--'characters.

Procedure:

You must first "shield coins" (send from a transparent R-addr to a shielded zaddr) to be able to use them in staking, when they first unlock.

This is part of the Zcash protocol itself. The commands below assume you are in the Verus source code directory

```
verus z_shieldcoinbase
```

You can either use an existing zaddr or use a new zaddr. To make a new zaddr:

```
verus z_getnewaddress
```

Use the address the above command outputs in the z_shieldcoinbase command

To shield all coinbase in your wallet, you can use "*" (quotes are important) and zaddr that is getting the funds:

```
verus z_shieldcoinbase "*" <YOUR zs-ADDRESS>
```

To just shield a single address, specify that as the first argument:

```
verus z_shieldcoinbase <YOUR R-ADDRESS> <YOUR zs-ADDRESS>
```

Once the funds have moved to the zaddr and are confirmed, you can freely send them to any address, They will be eligible for staking id sent to a transparent address (R-address).

To send from a certain zaddr to a transparent address, use z_sendmany. The following command sends 10 Verus to a given transparent address, ID address or ID-name.

Example: verus z_sendmany

```
zcZpfuzzJqmNJ3fUJekvbnyuxuJe9eAURAhrMCvN2Nr7VuWjakb1LEw6j2etPcCnr45BRot7M
aBipuS5da162BfuUkFGxx '[{"amount":10,"address":"Verus Coin
Foundation@"}]'
```

Important General Information

VRSCTEST data location:

Linux GUI: ~/.komodo/VRSCTEST

Mac OS: /Users//Library/Application Support/Komodo/VRSCTEST

Windows 10: %AppData%\Roaming\Komodo\VRSCTEST\

General remarks on CLI wallet:

On Windows command line enter the commands as shown without the surrounding quotation marks

In Linux shell precede the commands without surrounding quotation marks with ./

In Mac OS shell precede the commands without surrounding quotation marks with ./

Example: the windows version verus listtransactions transforms in Linux or Mac OS to ./verus listtransactions.

General remarks on Windows command line formatting:

The CLI help shows the command format for Linux and Mac OS.

For windows substitute the shown '-' character with the "--" character.

For windows substitute the shown "-" character with the "\--" characters.

Necessary files:

Link 1: Download latest Wallet

Procedure:

Joining Verus testnet to test the latest capabilities before they are released to mainnet or simply test if your goals are possible without spending VRSC (testnet coins hold no value) is easy.

Download a wallet

The first thing you need is a VRSC wallet. The CLI-wallet and Verus Desktop GUI wallet are available on the link above for Windows, Linux and Mac OS. If you already have a wallet verify that the wallet is the most recent version and update if needed.

Verus Desktop Wallet

Start your Verus Desktop wallet.

If you have never run Verus testnet on your system before:

Go to settings (cogwheel icon) and select General Settings.
Select Enable VRSCTEST.
Click Save Changes
Restart Verus desktop
When logged in, click Add Coin, select Verus Testnet and click Continue.
Select the startup parameters you desire (Native (Lite mode is not available on testnet), Staking, mining (specify number of threads), reindex blockchain and/or rescan wallet) and click Add Coin.
CLI wallet
1. start CLI walletdaemon using these parameters:
verusd -chain=VRSCTEST
Any extra parameter that you are used to for VRSC (like -mint or -pubkey=) can be appended as well.
2. commands through the CLI are in the following format:
verus -chain=VRSCTEST
The only difference with the normal VRSC chain is the -chain=VRSCTEST option, that is added.

Get the values of the current block height ("blocks") and the "blockstomaturity" the distance to maturity of the transaction in question. (i.e. the number of blocks between current block height and where it unlocks. You can see the number is decreasing all the time, and if you add it to the current block height you'll see a constant.)
Apply this formula: blockstomaturity / 1440 = days to maturity (unlocking)
e.g. 925558/1440=643 days

One block = 12 coins (as now)
NetworkHashrate = retrieved by getmininginfo command from </>CLI
LocalHashrate = retrieved by getmininginfo command
BlockTime = 60 seconds (average)
Real example with - 31 threads AMD Ryzen 5950x @ 4.4Ghz -
Bear in mind that these are average times to find a block. In real life you may hit a block much sooner or later after finding the last. In the long run it averages out to the values predicted.

Verus Wallet.dat, Chaindata & VRSC.conf standard locations
Linux: ~/.komodo/VRSC
Mac OS: ~/Library/Application Support/Komodo/VRSC
Windows 10: %AppData%\Roaming\Komodo\VRSC\
OS independent through Verus Desktop: Click help, Show Verus data folder (default)
Prerequisites
Have a native VRSC wallet running
In Verus Desktop
In Verus Desktop, there is at this moment no way to enter an address to mine to. However, editing the VRSC.conf can be done to achieve our goal: In the receive window of your wallet, click the hamburger (three vertical dots) next to the address you want to receive your rewards in and click Copy Public Key. Close down Verus Desktop.
Edit VRSC.conf (see standard locations at the top) and add the line pubkey=THELONGSTRINGCOPIED. Save and exit.
* Start Verus Desktop as you normally do.

In Agama Wallet

Step 1 - First get your wallet address you want to mine to:

* If you don't have an address, click "Receive", click "Get New Address" and choose "Transparent Address" from the drop down.

Step 2 - Next we need to retrieve our pubkey,

* click on the hamburg next to the address that you want to receive the rewards in and click copy pubkey

Step 3 - Set your PubKey

Go to 'Settings', 'App Config (config.json)' and enter your pubkey(THELONGSTRINGCOPIED) into the 'Pubkey VRSC mining key' field.

Click 'Save app config' to save these settings.

* Restart Agama

In Verus CLI

Step 1 - First get your wallet address you want to mine to:

You can find an address if you already have previous transactions, or you can create a new one. To find an address from a previous transaction, use the command line verus listtransactions and copy the address found after "address".

To generate a new wallet address, use the command line verus getnewaddress and a new address will be created.

Step 2 - Next, using your new address, enter the command with verus-cli verus validateaddress. From the output find the long string after "pubkey", copy without the quotation marks.

Step 3 - Set your PubKey

Option 1: use this pubkey when starting your daemon by adding the following line to the end of your command, just before the "&" sign: - pubkey=THELONGSTRINGCOPIED Option 2: edit your VRSC.conf and add the line pubkey=THELONGSTRINGCOPIED. Then start your whallet as you are used to.

Your rewards will now be mined to that address. It would be a good idea to keep notes and associate the wallet address with the pubkey...also to double check that you did validate the correct pubkey for the wallet address, making sure you made no errors.

Verus Wallet.dat, Chaindata & VRSC.conf standard locations

Linux: ~/.komodo/VRSC

Mac OS: ~/Library/Application Support/Komodo/VRSC

Windows 10: %AppData%\Roaming\Komodo\VRSC\

OS independent through Verus Desktop: Click help, Show Verus data folder (default)

Procedure

With one of the wallets loaded, issue the following command:

z_exportwallet FILENAME (ie z_exportwallet export_instance01, the filename cannot have a . in it.)

Copy the generated file to the machine that hosts your main wallet. This file will either be in the same directory as the config file and wallet.dat file, or in a location specified by exportdir in VRSC.conf.

Issue the following command (on the "main" verus-cli): z_importwallet /LOCAL_PATH/EXPORTFILENAME (ie /home/user/export_instance01)

note: Older versions of verusd required expordir to be set in VRSC.conf before exporting a wallet. If you get an error about your export directory not being set, please upgrade immediately.

These commands can be given in:

CLI wallet: ./verus z_exportwallet FILENAME& ./verus z_importwallet /LOCAL_PATH/FILENAME
Verus Desktop in settings, coin settings: run z_exportwallet FILENAME& run z_importwallet /LOCAL_PATH/FILENAME
Verus Agama in settings, <CLI>: z_exportwallet FILENAME& z_importwallet /LOCAL_PATH/FILENAME

The reward received depends of the blocknummer:

Era 1:

1st week: Block 0 - 10080 ==> 0 to 384 VRSC reward ==> 16,588,800 VRSC total this period (reward rising linearly and changing each block)

Era 2:

1st month: Block 10080 - 53279 ==> 384 VRSC reward ==> 8,294,400 VRSC total this period
2nd month: Block 53280 - 96479 ==> 192 VRSC reward ==> 4,147,200 VRSC total this period
3rd month: Block 96480 - 139679 ==> 96 VRSC reward ==> 4,147,200 VRSC total this period
4th month: Block 139680 - 182879 ==> 48 VRSC reward ==> 2,073,600 VRSC total this period
5th month: Block 182880 - 226079 ==> 24 VRSC reward ==> 1,036,800 VRSC total this period

Era 3:

Years 1+2: Block 226080 - 1277279 ==> 24 VRSC reward ==> 25,228,800 VRSC total this period
Years 3+4: Block 1277280 - 2328479 ==> 12 VRSC reward ==> 12,614,400 VRSC total this period
Years 5+6: Block 2328480 - 3379679 ==> 6 VRSC reward ==> 6,307,200 VRSC total this period
. . . halving indefinitely every 1051200 blocks (approximately 2 years)

All rewards equal or over 192 VRSC are time locked to mature at a random block between 129,600 and 1,181,520

Use with: <https://veruspay.io/api/> for simple USD VRSC price, or choose options now added!

Options (values are case insensitive):
currency - BTC or Fiat code like USD or CAD
ticker - ARRR or VRSC
data - volume or price - volume only relevant if exchange is defined
exch - name of supported exchange, e.g. digitalprice - If no exchange, price is average of all supported for that coin.

If no options are set, the default is average price in USD fiat of VRSC.

Examples:

<https://veruspay.io/api/> - This get's the current price of VRSC in USD, weighted against 24hr volume across all exchanges. This is the default return.

<https://veruspay.io/api/?exch=digitalprice¤cy=cad> - This will get the current price on digital price for VRSC and display in CAD fiat

<https://veruspay.io/api/?currency=btc> - This will get the average price of VRSC in BTC, weighted by 24 hr volume across both exchanges

<https://veruspay.io/api/?currency=cad> - This gets the current average price of VRSC in CAD, weighted by 24 hr volume across both exchanges

<https://veruspay.io/api/?exch=cryptobridge&data=volume> - This will get the 24 volume of VRSC on CryptoBridge in the default currency of USD

<https://veruspay.io/api/?exch=cryptobridge&data=volume¤cy=btc> - This does the same but with BTC as the currency result

<https://veruspay.io/api/?currency=cad&ticker=arrr> - Gets the average price of ARRR.

Mining VRSC with Verushash is at this time one of the most profitable coins to mine with your CPU.

If you want to know how profitable it is, you need to know a few important details about your own conditions.

1. CPU - If that is a fairly modern CPU with AES and AVX instructions built in.

This will be true if your processor is produced in 2013 or later. For processors between 2008 and 2012 you need to check the specifications. Older processors can still mine, but they will not perform well.

2. GPU - It is possible to mine with fairly modern Nvidia GPU's, but since Verushash uses specific functions from the mentioned AES and AVX instruction sets, they will perform better than old CPUs, but worse than modern CPUs.

3. FPGA - Many of these semi-specialized machines can be reprogrammed to mine VRSC. They don't have the same power per processor as a CPU, but they often have multiple processors running parallel. The stronger ones can outperform a CPU easily, but need a lot of power to do so.

If you know the answer to the hardware, you can look up a comparable one in the Spreadsheet from Link 1, to give you an idea of the performance.

If you know your energy price also, you can calculate an estimation of how much your hardware can earn you at this moment.

If you have to get new hardware just to mine, think about it for a moment: would you have bought new hardware anyway? If so, you can use this information to get an idea what kind of hardware you want to buy.

You can see the following characteristics:

1. a block hash without a lot of leading zeros

2. a last transaction that in your wallet has no fee, is from the same address that it is to, which is also the address the coinbase is to (easiest confirmation) and

3. The POS difficulty is encoded in the low 32 bits of the nonce with 96 of the next nonce bits as zero/reserved, then a random hash at the top half

You'll start staking with the first VRSCs that are not time locked + in your public / transparent wallet + 150 blocks old (or about 2.5 hours).

Your chances to win a block: Your coins in your public/transparent wallet / Total staking supply in public/transparent wallets (which is max. about 485.000 VRSC from the first (sunrise) week as long as the rewards were below 192)

Remember: There will be only one reward every minute. It's going to be either mining or staking, so on average 720 mining and 720 staking rewards every day.

Example: I have 300 coins in a public/transparent address / 300.000 in public wallets (let's assume some part is lost/not staking or in private wallets), so that would be $1/1000 \times 720$ of a chance or around average 1,4 days for a staking block rewards. Hash power does not influence staking reward.

Regarding the Verus debug.log: "No eligible staking transaction found". It means that you are staking but have not received a reward yet. @miketout will change the message soon.

Regarding time locked coins:

The Zcash protocol requires you to send all coins received by mining (on wallet, not pool mining) or staking (reward transactions, also on wallet, not pool staking) once unlocked to a private address and then to a public/transparent address before you can use them either for staking or for making transactions (that's how you make use of your rewarded coins = coinbase coins). So, once your coins loose their time lock, you can unlock those coins as described in ["Shield Verus Coins via Command Line Interface"](#!how-to/how-to_shield_via_cli.md). Once you've transferred the coins from your private address back to (one of) your public / transparent address(es) and you'll automatically start staking.

CLI:

You can check this in the VerusExplorer <https://explorer.verus.io/>

1. Check last 10 transactions in the </>CLI. Use listtransactions.

2. Copy the blockhash of the received award

Hint: An example:

<https://explorer.verus.io/api/getblock?hash=9e6fa91356211a554c580c90ec9c2067dd420ff74c7d33481775793f7b0e7f03> " so this one is minted....

Verus Desktop:

In Verus Desktop, simply go to your Mining Dashboard and enter into Verus details.

Scroll down to the bottom of the page. It will list the rewards as mined or minted in green.

The TXIDs that staked the minted rewards are shown in blue.

Verus Agama (Deprecated):

In the GUI you can also click yourself thru to that information.

1. Click on the magnifying glass all the way on the right of the transaction.

2. On the pop-up click on "Open in the VRSC Explorer" bottom left.

3. In the VRSC explorer click on the block hash value (in light blue) now the block hash is displayed as the title of the box.

4. Click on the info "i" on the right and click on "search": in the result displayed for blocktype 'mined' or 'minted'.(edited)

Verus Desktop

Verus Desktop shows your immature balance in your wallet dashboard, between the Transparant Balance and Private Balance.

Agama (Deprecated)

In the Agama wallet click on the hamburger menu (the three stripes) on the top right

click on Settings

click on the item </> CLI

Select VRSC as coin

Type the following command: getwalletinfo

click Execute

scroll down and find "immature_balance": which will give you the amount of time-locked coins in your wallet.

Important General Information

verus command "<userinput>" needs to be entered literally, with <userinput> replaced by your specific userdata. So if the text directs you to use for example "<Public Address>", you replace that (including the < and >) with the address, so it looks similar to this: "RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r".

Remarks on Windows command line formatting:

The CLI help shows the command format for Linux and Macos. On the native windows command prompt (cmd.com) the formatting is different.

In windows command prompt, substitute the shown '-'character with the "--" character. In windows command prompt, substitute the shown "-"character with the \"-characters.

* In windows command prompt, omit the preceding ./.

Note: As an example, in Linux the command:

```
./verus z_sendmany <my_private_address_without_quotationmarks>
'[{"address":"<my_transparent_address>","amount":<95.9998>}]'
```

should be entered on the Windows command prompt as:

```
verus z_sendmany <my_private_address_without_quotationmarks>
"[\\"address\":\"<my_transparent_address>\",\"amount\":<95.9998>}]"
```

Procedure:

1) ./verus z_shieldcoinbase "*" my_private_address_without_quotationmarks this capture all so called coinbases, i.e. mined coins that are not yet staking.

you have to wait 100 blocks (minutes) after receiving them before being able to move them.

wait for a few minutes for the tx to be confirmed.

```

2) ./verus z_getbalance <my_private_address_without_quotationmarks> this
is to subtract the 0.0001 VRSC fee from the balance in the next step.
3) ./verus z_sendmany <my_private_address_without_quotationmarks>
' [{"address":<my_transparent_address>,"amount":<95.9998>}]' 
amount is minus 0.0001 from balance and without quotation marks.
(4) to verify: ./verus z_gettotalbalance
after a few minutes (operations from private addresses are a bit time
consuming)

```

This solution can be solved in 2 ways: you can simply install the latest bootstrap file (less work, big download) or search manually for the forked block and invalidate that block (time-consuming, no download needed.)

Procedure 1 (Easy, installing bootstrap)

For all GUI or CLI users.

1. Stop the wallet/mining process by cleanly shutting down the program.
2. Update your wallet if necessary.
3. Follow the procedure in HOW-TO Backup, Install or Update and Bootstrap your wallet.md to efficiently rectify the problem.
4. Do not be dismayed if it seems that your mining rewards suddenly seem to come to a halt. Remember, when you mine to the wrong chain rewards can come in very quickly, but they are worth nothing.

Procedure 2 (Time consuming, no extra download)

Verus-Desktop

The commands are all entered in the Native Client Terminal that is located under Settings, Coin Settings.

Search for the earliest block that not matches the blockchain:

run getblockhash <suspected blocknumber> will show you the blockhash for the blocknumber you filled in

The response shown in the Native Client Terminal will be similar to this:
5cc7844973fb95ef17f1772ea4aba579f0d8273fb0ee6064cd8e707d1056c646

Check the blockhash your command gave you against the blockhash the explorer shows.

If the blockhash from the explorer is different than yours, repeat steps 2 & 3 until you find the earliest block that is different.

Use the earliest incorrect blockhash from your system to invalidate that block:

run invalidateblock <earliest incorrect blockhash>

The Native Client Terminal will not give feedback on this command.

Now use the correct blockhash that the explorer gave you for the block you just locally invalidated:

run reconsiderblock <correct blockhash>

Again the Native Client Terminal will not give feedback on this command. Once your wallet connects to a node that is on the correct chain, it will quickly synchronize.

If needed you can either restart your wallet to force new connections or manually disconnect bad nodes.

CLI

Search for the earliest block that not matches the blockchain:

./verus getblockhash <suspected blocknumber> will show you the blockhash for the blocknumber you filled in

The response will be similar to this:

5cc7844973fb95ef17f1772ea4aba579f0d8273fb0ee6064cd8e707d1056c646

Check the blockhash your command gave you against the blockhash the explorer shows.

If the blockhash from the explorer is different than yours, repeat steps 1 & 2 until you find the earliest block that is different.

Use the earliest incorrect blockhash from your system to invalidate that block:

```
./verus invalidateblock <earliest incorrect blockhash>
```

The deamon will not give feedback on this command.

Now use the correct blockhash that the explorer gave you for the block you just locally invalidated:

```
./verus reconsiderblock <correct blockhash>
```

Again the daemon will not give feedback on this command.

Once your daemon connects to a node that is on the correct chain, it will quickly synchronize.

If needed you can either restart your daemon to force new connections or manually disconnect bad nodes.

luckpool.net

pool.verus.io

zergpool.com

wattpool.net

vrsc.ciscotech.dk

www.lepool.com.cn

www.zhuaao.com

aod-tech.com

verus.alphatechit.co.uk

MadCatMining (currently inactive)

Dudezmobi

Ginasis

A staking pool has advantages and disadvantages over solo staking:

Disadvantages:

1. It causes centralization on the network
2. The pool owner is in control of all funds in the pool, including yours.
3. You need to trust the pool operator to share the rewards fairly.
4. You need to trust the pool operator to release your funds back to you on request.
5. You need to trust the pool operator to securely run the pool 24/7.

Advantages:

1. You don't need to run your wallet 24/7 in native mode.

The CLI help shows the command format for Linux and MacOS. On the native windows command prompt (cmd.com) the formatting is different.

For windows substitute the shown '-'character with the '-'character.

For windows substitute the shown '-'character with the '\'-'characters.

In order to let your mining not interfere with other processes running on your PC, we'll need to deprioritize

the mining process. This will result in your miner throttling down whenever your PC needs processing power.

Normally you don't need to worry about these locations, but in some instances you will be asked by community members providing support to look up a file in a folder in your Verus-Desktop installation.
note: changing files in these folders or subfolders may result in a corrupt installation. Only do so when instructed by our support community members.

Deamons

The daemons are located in the komodod, verusd and zcashd folders that can be found in the resources\app\assets\bin\win\ subfolder of your installation folder.

Program settings

Verus Desktop saves its program settings on a different folder:

%AppData%\Verus-Desktop

The users settings are stored in appdata\config.json in the program settings folder.

Standard chain data and wallet locations

KMD

%AppData%\Komodo

Verus

%AppData%\Komodo\VRSC

Komodo asset chains

Any Komodo asset chain will create a subfolder in the KMD chain data and wallet folder, which is standard named. The names will be in capitals and are identical to the official asset-chain name.

%AppData%\Komodo\<CHAIN-NAME>

Note: examples

Pirate: %AppData%\Komodo\PIRATE

Utrum: %AppData%\Komodo\OOT

Zexo: %AppData%\Komodo\ZEXO

And so on...

For easy access to the binaries folders, Verus-Desktop program settings and VRSC chain folder and all binary folders, you can use the debug menu in Verus-Desktop.

First make sure your system is up to date:

sudo apt-get update && apt-get upgrade -y

I suggest not to use the root account. If you have not yet done, set up a new user

sudo adduser newusername

sudo usermod -aG sudo newusername

Switch to new username

su - newusername

Test if your new user actually has root (SuDo) access, e.g.:

sudo ls -la /root

You should get some lines like this:

drwx----- root root Jul :

Download & install the wallet binaries:

wget https://github.com/VerusCoin/VerusCoin/releases/download/v.../Verus-CLI-Linux-v...-amd.tgz

The downloaded archive contains another archive and a signature text file, enabling the archive within to be verified (You'll need a running wallet to do that)

Also: Verify the URL to the latest version from the Download latest Wallet above.

```

tar -xvf Verus-CLI-Linux-v..-amd.tgz
Now extract the wallet archive:
tar -xvf Verus-CLI-Linux-v..-amd.tar.gz
Change directory to verus-cli
cd verus-cli
Fetch parameters, takes time, more on slow Internet connection
./fetch-params
Creating the chaindata directory
cd ~
mkdir -p .komodo/VRSC
cd ~/.komodo/VRSC
Download the block-chain bootstrap, this considerably speeds up
synchronisation of the block-chain from days to minutes... (optional)
wget https://bootstrap.verus.io/VRSC-bootstrap.tar.gz
tar -xvf VRSC-bootstrap.tar.gz
Install libraries for Verus
sudo apt-get install libcurl g++-multilib -y
Install Tmux a terminal multiplexer with which you can run threads in the
background see https://en.wikipedia.org/wiki/Tmux
sudo apt-get install tmux -y
Start tmux:
tmux
Launch Verus Daemon with or without number of threads
(usually number of threads equals number of cores or double of that if
the processor support hyper threading well)
~/verus-cli/verusd -gen -genproclimit
~/verus-cli/verusd -gen -genproclimit=
Once mining is operational - again this may take some time -
you'll see: mega hashes complete - working
then detach tmux
[ctrl]&b d
Disable login with Root User
(make sure your newly created user login works and has sudo rights)
sudo nano /etc/ssh/sshd_config
Find: PermitRootLogin yes
And set to
PermitRootLogin no
Apply new settings:
sudo systemctl restart sshd

```

You can use the following scripts to automate it (prints, for each block, how many blocks remaining and the estimated date).

Keep in mind that the actual number of blocks per day is not always exactly . It varies and the estimated dates may change slightly over time. Below you will find the Linux script that will calculate the extimated time for you:

```

Script:
#!/bin/bash
#Config
verus_path='/XXXX/verus-cli/'
verus_cli='verus'
get_transactions=( $(${verus_path}${verus_cli} listtransactions '') |grep
blockstomaturity|sed 's/.e: //;s/,//g'|awk '{ print $ }'|sort -n )
```

```

cur_block=( $(($verus_path$verus_cli getmininginfo|grep blocks':|sed
's/.e: //;s/,//g'|awk '{ print $ }') )
arr_idx=''
for i in ${get_transactions[@]}
do
days_to_mature=$(( $(($i))/ ))
mature_to_date=$(date '+%m-%d-%Y' -d '+$days_to_mature days')
echo "Block # ${arr_idx} will mature in approximately ${days_to_mature}
days" '(' $mature_to_date ')'
((arr_idx++))
done
You'll need to enter the correct path to your verus directory and you
will need to have the verus daemon running in order to sucessfully run
this script.

```

Following are some bash scriptse to help make managing your Linux-based CLI miner a bit easier. Prereq is to install mailutils (postfix) and configure with your server's FQDN and set inet_interfaces=localhost in the postfix/main.cf file.

Important General Information

VRSC Wallet & data location on Linux: ~/.komodo/VRSC
for wallet version prior to .., replace verusd with komodod.

Scripts:

```

MINER SRVC MONITOR & ALERT IF DOWN
Checks for the verusd daemon and if it has stopped emails you.
checkifverusdisrunning.sh
#!/bin/bash
if pgrep -x 'verusd' > /dev/null
then
TRUE=''
else
echo 'Merry Miner Has Stopped Mining!!! HELP!!' | mail -s 'OUTAGE: Merry'
-a 'From: user@yourqualifieddomain.tld' you@youremail.tld
fi
ALERT ON NEW BLOCKS MINED
Prereq: Create a file called txHistory.txt and put in it, saved to your
home folder.
The script then compares the current wallet TX count and compares to the
txHistory file...
so first run it will enter the right number in that file overwriting your
.
Only emails you if the number changes.
checkfornewblocks.sh
#!/bin/bash
historicalcount=$(cat /home/user/txHistory.txt)
livecount=$(./home/user/verus-cli/verus getwalletinfo | grep txcount | sed
's/[^-]*//g')
if (( $livecount > $historicalcount))
then
echo $livecount > /home/user/txHistory.txt
echo 'Merry Miner Has Mined a Total of $livecount Blocks! Woot!' | mail -
s 'Merry's Blocks: $livecount' -a 'From: user@yourqualifieddomain.tld'
you@youremail.tld
else

```

```
NOCHANGE=' '
fi
WALLET BACKUP TO SECURED EMAIL (PROTONMAIL SUGGESTED)
For this script I recommend setting up a new Protonmail account with no
association to any other service or your name, FA secure it.
Schedule script in CRONTAB
In the following, the */ is every min, the is on the hour every hour, the
is every day at PM.
CRONTAB
# m h dom mon dow command
*/ * * * * /home/user/checkfornewblocks.sh
* * * * * /home/user/checkifverusdisrunning.sh
* * * * * /home/user/backupwallet.sh
Note: For any emails sent (for backup of dat file for example) make sure
to enforce TLS security in postfix by adding the following line to your
/etc/postfix/main.cf
smtp_tls_security_level=encrypt
```

Komodo is probably already running.

Procedure:

It just means you can't start it when it's already running. If it isn't running we can take a look at it.

Check to see if it is running by running:

```
./verus getinfo
```

If it's running you'll get info back, if it isn't you'll get an error.

If that throws an error, it's worth checking using this command:

```
ps fax | grep verus
```

and seeing if any processes are listed besides the process you're using to search.

You would see verusd with a child process of komodo and all its cli arguments.

Note: I use Google Cloud for my staking wallet and backups but the script works regardless of what you use; if you have timelocked coins, you will have to keep your wallet.dat file secure for a long, long time: it's crucial to maintain secure the staking instance as well as the storage..

Important General Information

VRSC Wallet and Data location on Linux: ~/.komodo/VRSC

Procedure:

If you use GCloud:

```
create a new directory where we'll upload the backups (ie: mkdir
/home/user/backup) create a storage bucket via GCloud console (ie
bucket_verus)
```

```
* mount the bucket on your local machine (syntax is: gcsfuse ; ie gcsfuse
bucket_verus /home/user/backup)
```

The script can be crontab scheduled to run daily or you can run it manually; if you schedule it, you have to hardcode the encryption passphrase; otherwise you can input it manually each time. You will have to install gnupg to encrypt (sudo apt-get install gnupg gnupg-agent). As always: be sure to fully understand what the script does and why I left some echos to demonstration/test purposes, you can safely remove all of them.

```
#!/bin/bash
```

```
# Customize the SOURCE and the DEST folders
```

```

SOURCE_FOLDER=/home/XXXXX/.komodo/VRSC/
DEST_FOLDER=/home/XXXX/backup/
BCK_DATE=date +%Y-%m-%d.%H:%M
SOURCE_FILE=wallet.dat
DEST_FILE=$SOURCEFILE'$BCK_DATE
ENCRYPT_EXT='.gpg'
# all echo are for test purposes, feel free to delete them
echo check variable
echo SOURCE - $SOURCE_FOLDER
echo SOURCE_FILE - $SOURCE_FILE
echo DEST_FILE - $DEST_FILE
echo DEST - $DEST_FOLDER
echo TO_BE_DEL - $DEST_FOLDER$SOURCE_FILE
sleep
cp $SOURCE_FOLDER$SOURCE_FILE $DEST_FOLDER$DEST_FILE
sleep
# Customize the passphrase!!
gpg --batch --yes -c --cipher-algo AES --passphrase='XXXXXX'
$DEST_FOLDER$DEST_FILE
sleep
rm $DEST_FOLDER$DEST_FILE
# Keep only last days - BE SURE to fully understand how it works, as
every 'rm' command!
find $DEST_FOLDER/wallet* -type f -mtime + -exec rm {} ;

```

Read it completely before using.

Important General Information

This guide is aimed towards Debian based Linux distributions. If you are using a different kind of distribution (eg RPM-based, like CentOS) you will need to install the dependancies using a procedure that fits your specific distribution.

There are active branches in ccminer github repo:

ARM (for bit ARM chips with AES intrinsic)

Verus. (standard x- pc's)

Verus.gpu (GPUs)

Note: Replace ARM in the git clone line below with the branchname above you want to use.

Procedure:

Install dependencies (specific for Debian-based distributions):

```

sudo apt-get install libcurl-openssl-dev libssl-dev libjansson-dev
automake autotools-dev build-essential git

```

For GPU-miner compilation additional sources are required (Not needed for CPU or ARM) :

wget

```

http://developer.download.nvidia.com/compute/cuda/./Prod/local_installers
/cuda_..._linux.run
sudo sh cuda_..._linux.run

```

Download the source and compile:

```

git clone --single-branch -b ARM https://github.com/monkins/ccminer.git
cd ccminer

```

chmod +x build.sh

chmod +x configure.sh

chmod +x autogen.sh

```
./build.sh
And finally starting the miner (Change pool, address & workername to your
own liking):
./ccminer -a verus -o stratum+tcp://pool.verus.io: -u
RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ.donator -p x
```

Procedure:

These tools will help you stake on your permanently running machine using Verus CLI, instead of running the Agama GUI wallet.

Features

automatic shielding and unshielding of coinbase (staking, solo mining)

automatic tracking of total coins generated (also present in e-mail notifications)

optional e-mail notifications on newly generated coins

optional periodical wallet.dat backups, also as an encrypted ZIP archive

optional periodical wallet balance e-mail notifications

Initial Verus daemon configuration

Follow the guide in link above to set-up your daemon for the first time. You will need a transparent address, a zs address and the daemon running with correct parameters (-mint -cheatcatcher=zs...).

Do not stake with the same wallet.dat on multiple nodes.

Procedure:

In order to let your mining not interfere with other processes running on your PC, we'll need to deprioritize the mining process. This will result in your miner throttling down whenever your PC needs processing power.

People have reported to be able to use CPU-heavy applications, like games, without the miner interfering, with this method.

Because the Verus wallet (GUI and CLI) does a lot more than mining, it is not recommended to use this for a solo-mining setup. This is tested on Verus NHEQminer and Verus CCMiner on Ubuntu ..

First thing you will need is pool-mining software. The link is supplied above this procedure. Download your preferred miner, extract it to your preferred location and configure the batchfile with the pool of your choice and your own mining address.

Now we'll need to adjust the batchfile to run on low priority:

NHEQMINER:

Run your miner as you normally would, with nice -n placed in front of the command you normally use,

for example:

```
nice -n ./nheqminer -v -l pool.verus.io: -u
```

```
RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ.Donator -p x -t
```

make sure you replace the pool (pool.verus.io:) with your preferred pool and the address

(RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ) with your own address. The address shown in this line is the veruscoin foundation donation address. If desired change the identifier (Donator) to a name that easily identifies the miner to you.

CCMINER:

Run your miner as you normally would, with nice -n placed in front of the command you normally use,

for example:

```
  nice -n ./ccminer -a verus -o stratum+tcp://pool.verus.io: -u
  RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ.Donator -t
  make sure you replace the pool (stratum+tcp://pool.verus.io:) with your
  prefered pool and the address
  (RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ) with your own address. The address
  shown in this line is the veruscoin
  foundation donation address. If desired change the identifier (Donator)
  to a name that easily identifies the
  miner to you.
```

When running ./verusd on a Linux distro (eg Debian or Devuan), not all dependencies may be installed by default, resulting in the error message error while loading shared libraries: libgomp.so.: No such file or directory or error while loading shared libraries: libz.so: No such file or directory.

To solve this you need to install the libgomp and zlib-dev libraries:
sudo apt-get install libgomp zlib-dev

The Linux version of Verus Desktop is supplied as an AppImage, so it does not get installed, but mounted in the /tmp folder on execution.
The folder it is mounted to will look like .mount_Verus-xxxxx where the xxxxx part will be a random set of characters, changing on every start of the wallet.

(It is a hidden folder: in order to see it, <CTRL>-H will toggle the display of hidden folders)

Normally you don't need to worry about these locations, but in some instances you will be asked by community members providing support to look up a file in a folder in your Verus-Desktop installation.
note: changing files in these folders or subfolders may result in a corrupt installation. Only do so when instructed by our support community members.

Deamons

The daemons are located in the komodod, verusd and zcashd folders that can be found in the
resources/app/assets/bin/linux subfolder of your (mounted) program folder.

Program settings

Verus Desktop saves its program settings on a different folder:
~/.verus-desktop

The user's settings are stored in appdata/config.json in the program settings folder.

Standard chain data and wallet locations

KMD

~/.komodo

Verus

~/.komodo/VRSC

Komodo asset chains

Any Komodo asset chain will create a subfolder in the KMD chain data and wallet folder, which is standard named. The names will be in capitals and are identical to the official asset-chain name.

~/.komodo/<CHAIN-NAME>

Note: examples

Pirate: ~/.komodo/PIRATE

Utrum: ~/.komodo/OOT

Zexo: ~/.komodo/ZEXO

And so on...

For easy access to the binaries folders, Verus-Desktop program settings and VRSC chain folder and all binary folders, you can use the debug menu in Verus-Desktop.

Procedure:

It just means you can't start it when it's already running. If it isn't running we can take a look at it.

Check to see if it is running by running:

`./verus getinfo`

If it's running you'll get info back, if it isn't you'll get an error.

If that throws an error, it's worth checking using this command:

`ps fax | grep verus`

and seeing if any processes are listed besides the process you're using to search.

You would see verusd with a child process of komodo and all its cli arguments.

Verus Desktop

Currently only macOS ..x and up are tested and supported for Verus Desktop. Installing Verus Desktop on OSX ..x or earlier may be possible but the wallet does not function and never completes the initial sync.

Verus CLI wallet

The CLI wallet should run on lower versions without problems.

File location

VRSC Wallet and Data location on Mac OS: /Users//Library/Application Support/Komodo/VRSC

Necessary files:

Link : How-To install the latest Wallet

Procedure:

If you installed on OSX ..x or earlier and need to remove it:

Quit your Wallet.

Eject the wallet dmg.

Make a backup of wallet.dat & VRSC.conf if necessary (Only if you had a wallet on this machine or if you used an existing wallet.dat)

If you installed the Agama.app in Applications, move this to the trash.

Move ~/Library/Application Support/Agama to the trash or use the following command in terminal

`rmdir /Users//Library/Application\ Support/Agama.`

Move ~/Library/Application Support/Komodo to the trash or use the following command in terminal

`rmdir /Users//Library/Application\ Support/Komodo.`

Upgrade OS.

Install Agama (Check Link for a smooth install)

VRSC Wallet and Data location on Mac OS: /Users//Library/Application Support/Komodo/VRSC

Software needed to mine

Prerequisite: VerusCoin wallets

Needed: VerusCoin miners

Prerequisites:

To start mining with your Mac, you will need to have an address to mine to. Possibilities are:

An exchange address (NOT recommended. This is asking for trouble.)

An address from a web-wallet

An address from a mobile wallet

An address from a Verus-Desktop Lite wallet

An address from a Verus-Desktop Native wallet

Procedure:

To start mining on your Mac, you should already have a wallet and and access to an address to mine to. Now to start mining we will need to download the miner first:

. Go to the Veruscoin miners link above and download CLI nheqminer for MacOS.

. Open the download folder in finder.

. doubleclick nheqminer-MacOS-v...tar.gz to unpack it in your download folder.

. Open the nheqminer folder that appeared in your download folder

Now we need to configure things so the miner connects to a mining pool (listed on the download page, below the miner download) and mines to your address. In the instructions below I will use the VerusCoin community pool and the address for the Veruscoin foundation. make sure to use your own address and if desired the details of a different pool:

. click start.sh and open with TextEdit.app

. On the nd line, change PoolHost= into Poolhost=pool.verus.io

. On the rd line, change Port= into 'Port='

. On the th line, change 'PublicVerusCoinAddress=' into

PublicVerusCoinAddress=RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ

(make sure you use your own address!!!!)

. On the th line, change WorkerName= into WorkerName=MacOS (or any name you want to identify your system)

In the next step we will set how many processor threads the miner will use. Make sure to use less threads than your processor has virtual cores. If you allow the miner to use all cores, your system will become sluggish at times, or it may even become completely unresponsive:

) On the th line, change Threads= into Threads= (fill in the amount of virtual cores you want to use)

) Save and exit the file.

We have all the settings ready to go, but the file we just edited is not executable right now.

. Click Start.sh and click copy 'Start.sh'

. Click in an empty space in you folder and click Paste items, which will result in a new start copy.sh file.

. rename start copy.sh to start.command

Now we have an executable file that will run when you doubleclick.

. Doubleclick start.command and your machine will start mining.

Procedure:

In order to let your mining not interfere with other processes running on your PC, we'll need to deprioritize the mining process. This will result in your miner throttling down whenever your PC needs processing power.

People have reported to be able to use CPU-heavy applications, like games, without the miner interfering, with this method.

Because the Verus wallet (GUI and CLI) does a lot more than mining, it is not recommended to use this for a solo-mining setup.

First thing you will need is pool-mining software. The link is supplied above this procedure. Download your preferred miner, extract it to your

preferred location and configure the batchfile with the pool of your choice and your own mining address.

Now we'll need to adjust the batchfile to run on low priority:

NHEQMINER:

Run your miner as you normally would, with nice -n placed in front of the command you normally use,

for example:

```
nice -n ./nheqminer -v -l pool.verus.io: -u
```

```
RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ.Donator -p x -t
```

make sure you replace the pool (pool.verus.io:) with your preferred pool and the address

(RVjvbZuqSGLGDmBBFkbHWySPKExtfjQ) with your own address. The address shown in this line is the veruscoin foundation donation address. If desired change the identifier (Donator) to a name that easily identifies the miner to you.

The installer for Verus-Desktop suggests a standard location to install to Applications

Anyone who uses the computer
/Applications/Verus-Desktop.app

Deamons

The daemons are located in the komodod, verusd and zcashd folders that can be found in

/Applications/Verus-Desktop.app/Contents/Resources/app/assets/bin/osx
Program settings

Verus Desktop saves its program settings in your home folder:
~/Library/Application\ Support/Verus-Desktop

The users settings are stored in appdata/config.json in the program settings folder.

Standard chain data and wallet locations

Verus Desktop saves its chain and wallet data in your home folder:

KMD

~/Library/Application\ Support/Komodo

Verus

~/Library/Application\ Support/Komodo/VRSC

Komodo asset chains

Any Komodo asset chain will create a subfolder in the KMD chain data and wallet folder, which is standard named. The names will be in capitals and are identical to the official asset-chain name.

~/Library/Application\ Support/Komodo/<CHAIN-NAME>

Note: examples

Pirate: ~/Library/Application\ Support/Komodo/PIRATE

Utrum: ~/Library/Application\ Support/Komodo/OOT

Zexo: ~/Library/Application\ Support/Komodo/ZEXO

And so on...

For easy access to the binaries folders, Verus-Desktop program settings and VRSC chain folder and all binary folders, you can use the help menu in Verus-Desktop.

Verus Wallet.dat, Chaindata & VRSC.conf standard locations

ARM Linux: ~/.komodo/VRSC

Note: Due to the size of the blockchain, on an ARM installation this directory is usually stored on a mounted external hard drive.

This will have its own file path that was specified when the drive was mounted.
Thus the path to `~/.komodo/VRSC` directory is likely to be user-specified & thus non-standard on an ARM.
(NB This version is aimed at inexperienced users, so it includes the associated routine/basic Linux commands.
You will need access to the relevant directories, which may require sudo privileges)
(NB Raspberry Pi ARM boards lack some instruction sets to mine, though the Pi & Pi are ARM & can be used as staking devices if an ARM operating system is installed)

Procedure:

Navigate to the directory which contains the verus daemon binary file (the default is in the home directory - ie `~/verus-cli`)
`cd verus-cli`
Check the Verus daemon is running - you will need to keep it running to check the downloaded file's signature
`top`
(`verusd` should appear as one of the processes in the list. If you need to start it use the command `./verusd`.)
Download the latest ARM version. Check <https://verus.io/wallet/command-wallet>
(The commands below are for v...-) Don't forget to change the version number to whichever one you are updating.
`wget https://github.com/VerusCoin/VerusCoin/releases/download/v...-/Verus-CLI-Linux-v...-arm.tgz`
unpack/unzip the download into the signature file and the binaries tar/zipped file
`tar -xvf Verus-CLI-Linux-v...-arm.tgz`
Check it has unpacked/unzipped & open the signature file
`ls -la`
`cat Verus-CLI-Linux-v...-arm.tar.gz.signature.txt-file`
This will display the signature information to verify the downloaded file is authentic.
CHECK the signature by using the data in the signature file
`./verus verifyfile 'Verus Coin Foundation Releases@' '[signature string from signature.txt-file]' [your home directory file path]Verus-CLI-Linux-v...-arm.tar.gz`
The return should be true. If you have an issue add '' around the file path.
STOP the Verus daemon
`./verus stop`
Check whether it has stopped running with this command. (To be doubly sure you can also look in the debug file in the)
`top`
(This time `verusd` should NOT appear in this list.)
Rename the old versions of the binaries (replace the version numbers with the version being replaced)
(This ensures you can easily revert to the old version by renaming these files.)
`mv verus verus-v...-`
`mv verusd verusd-v...-`
Check they have been renamed
`ls -la`

```

Unpack/unzip the binaries for the latest version
tar -xvf Verus-CLI-Linux-v...-arm.tar.gz
Check if they have unpacked/unzipped in the directory
ls -la
If the directory where your Verus wallet.dat, Chaindata & VRSC.conf files
are stored is in a non standard place - eg on a mounted external hard
drive - you may have a datadir specified in your VRSC.conf file.
Navigate to that directory to check your datadir is still correct. The
VRSC.conf file may have been overridden in the update. This ensures the
Verus daemon knows where to find the blockchain you have currently saved
on your system.
cat VRSC.conf
If it requires editing, use nano to edit this file
nano VRSC.conf
& append this to the VRSC.conf file :
datadir=[your custom file path for your mounted hard drive]
Start the Verus daemon again on the new version (add any other start
options from https://wiki.verus.io/#!faq-cli/clifaq\_verusd\_options.md)
./verusd
Check the version
./verus getwalletinfo
Once you are satisfied the new version is working, clean up the older
binaries (from step ) by deleting them (or just move them to another
directory using the same mv command as step )
rm verus verus-v...
rm verus verusd-v...

```

There are minimum requirements on your hardware and software for running a Verus Wallet on ARM-devices. If your platform does not meet the minimum requirements, you may not be able to run the required software.

The listed requirements are for running one chain. Additional PBaaS chains require more resources.

Verus Wallet on CLI

Absolute minimum requirements:

- bit processor
- bit Operating system (Raspbian is standard bit)
- GB memory + GB Swap available to the CLI Wallet
- GB storage space for Verus Blockchain and CLI wallet
- internet connectivity
- libgomp and zlibg-dev libraries installed

Recommended requirements

- bit processor with AES functions enabled
- bit Operation system (Raspbian is standard bit)
- GB memory or more + GB Swap available to the CLI Wallet
- GB storage on a fast medium (like NVMe device) for the Verus Blockchain & CLI wallet. This supplies room for blockchain growth over time and the ability to bootstrap the wallet.
- internet connectivity
- libgomp and zlibg-dev libraries installed

Verus Wallet on GUI

Absolute minimum requirements:

- bit processor
- bit Operating system (Raspbian is standard bit) with GUI interface
- GB memory + GB Swap available to the Verus Desktop Wallet

GB storage space for Verus Blockchain and Verus Desktop.
internet connectivity
libgomp and zlibg-dev libraries installed
Recommended requirements
-bit processor with AES functions enabled
-bit Operation system (Raspbian is standard bit)) with GUI interface
GB memory or more + GB Swap available to the CLI Wallet
GB storage on a fast medium (like NVMe device) for the Verus Blockchain & CLI wallet. This supplies room for blockchain growth over time and the ability to bootstrap the wallet.
internet connectivity
libgomp and zlibg-dev libraries installed
Staking
Absolute minimum requirements:
All requirements to run a wallet
Recommended requirements
All requirements to run a wallet
A fully configured and functioning NTP client, to keep your clock synchronized
Low latency internet connectivity
Solo Mining
Absolute minimum requirements:
All requirements to run a wallet
Recommended requirements
All requirements to run a wallet
-bit processor with AES functions enabled
Low latency internet connectivity
Pool Mining
-bit processor with AES functions enabled
-bit Operation system (Raspbian is standard bit)
ARM release of CCMiner or NHEQminer
Low latency internet connectivity
A public address to mine to
A public mining pool to connect to

When running ./verusd on a Linux distro (eg Debian or Devuan), not all dependencies may be installed by default, resulting in the error message error while loading shared libraries: libgomp.so.: No such file or directory or error while loading shared libraries: libz.so: No such file or directory.

To solve this you need to install the libgomp and zlibg-dev libraries:
sudo apt-get install libgomp zlibg-dev

Its the Revolutionary Layer / Protocol Launches its Most Anticipated Mainnet Upgrade

After years of hard work and dedication, on the fifth anniversary, the worldwide Verus community is excited to announce the launch of the highly anticipated mainnet upgrade on May . The Public Blockchains as a Service release marks a major milestone for Verus and the entire cryptocurrency industry.

PBaaS activation: block height ,, (Sunday May)

Turning the Crypto Industry on its Head

The wide cryptocurrency industry will see how much better Verus PBaaS is as a solution for true decentralization in cross-chain, DeFi AMMs, on-

chain PP exchange, self-sovereign provable recoverable IDs, NFT capabilities, anti-phishing, anti-MEV, scale, or just about any of the challenges people have on crypto platforms today. In fact, as we prepare for activation of this protocol that enables so many new use cases for crypto, not to mention easy onboarding when entrepreneurs discover how to really leverage VerusIDs, PBaaS is so far beyond what people are experiencing on any crypto platform today that it has been easy to dismiss our community as describing the impossible.

Once PBaaS is live on mainnet and people can actually use all of this themselves, along with every project, chain and currency on the PBaaS network and EVERY ERC or ERC on Ethereum or even bridged to Ethereum, the truth of Verus will be self-evident, and those who build on Verus will more easily build faster, better, more secure apps and services with the possibility of provable identity + privacy, crowdfunded projects, businesses, economies, and public infrastructure efforts all seamlessly integrated into UIs that do not need business deals, permission, or any centralized infrastructure to connect services and users, enabling everyone to communicate in provable, private, or public ways that always include bidirectional, secure commerce of all kinds.

Core Capabilities

The core capabilities of this release include (lots to learn to understand it all, but here's a partial list):

Verus Intersystem Protocol (VIP) – Layer Multichain and Inter-chain Protocol

To our knowledge, the VIP protocol is the only fully decentralized, provable cross-chain technology available on any network that is based on cryptographic proofs of each chain with optional witnesses to confirm chain state. From what we've been able to learn about LayerZero, Cosmos, Polkadot, Thorchain or others, VIP is different from (though closest in some ways to LayerZero), superior to, and more decentralized than all other cross-chain protocols we have seen. We can of course get into any level of discussion on the topic, but expect more descriptions and educational material to follow, now that the protocol was originally conceived of years ago, and has been developed and tested/hardened for years.

VerusID Content Multimaps – Unlimited Provable Data for Every VerusID
Of course this upgrade retains all of the groundbreaking capabilities already available with VerusID, including multisig options, separate revocation and recovery authorities for maximum security to prevent theft of identity or funds, and the ability to timelock your ID, enabling you to even thwart attempts to access your funds after someone compromised your secret keys.

With the PBaaS upgrade, VerusIDs also gain a powerful new capability that amounts to an unlimited, rent-free indexed database for every VerusID on every blockchain of the PBaaS network. This data can be encrypted or published in the clear and used for any purpose or application, whether social networking, voting, oracles, publishing any information, various forms of provable attestations, ratings, and much, much more. With the scalability of PBaaS and VerusID content multimaps together, the Verus PBaaS network becomes the most scalable, permissionless, provable source of human data from which AIs can learn about humanity in the world. We don't believe anything else even comes close, and with the recent rise of AI, it feels great to have this come together as a core original goal at a very appropriate and important time in history.

Permissionless Chain, Token, and Liquidity Basket Currency Launches
Whether you are looking to crowdfund an effort in a manner much like Kickstarter or IndieGoGo, sell identities to which you expect to add value or that are sub-IDs of a very cool root ID, or launch an entirely new blockchain economy and blockchain that starts with all of the Verus technology and a bridge to Verus and Ethereum from day , Verus is probably the best platform for you. In fact, as we were planning for the next testnet and wanted to have our own version of DAI to use in the bridge, since we want only decentralized currencies in the live mainnet bridge, we used a command to launch a token on the old testnet, exported it and all its supply to Ethereum's Goerli on the existing bridge, and that will be the DAI proxy/simulant that we will use on the new testnet. It saved us all some time, proving that Verus PBaaS will also be the easiest, most efficient way to launch even an Ethereum ERC, whether that ERC represents a token, DeFi basket, or even another PBaaS blockchain Verus enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing % POW/% POS, % hash attack resistant blockchain that can send and receive from the Verus chain which launched it.

All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Verus also enables any ID owner to define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from % to %. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners, stakers, and liquidity providers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Decentralized Ethereum Bridge

Shortly after PBaaS activates on mainnet, the Ethereum bridge will launch as a decentralized gateway and : provably mapped currency called "vETH" on the Verus network, also available to all PBaaS chains. The Ethereum bridge will also include a % backed basket of % Verus, % Ethereum, and % DAI, which will have the following functions:

Auto conversion of fees from Verus <-> Ethereum when sending cross chain, based on the on-chain conversion price in the liquidity basket

Permissionless ability to register *.vETH IDs, which in addition to the normal sub-ID capability of creating a single token that has control over the ID and can be exported to Ethereum as an automatic ERC NFT, can alternately be used to create a "mapped currency", which can be provably mapped : to any ERC currency on the Ethereum network by exporting it to the vETH bridge. Only Verus root IDs, and IDs of a gateway can create mapped currencies to a gateway.

Decentralized, fair bridge launch. The vETH bridge will launch as other currencies, chains, and gateways can launch, including the bridge converter. When the bridge launches, the contracts will have a surplus of Verus that comes from the fees paid to launch the gateway. In the protocol, that surplus is first used to solve the chicken and egg problem

of none of one currency on the other chain. The contracts will cover the Verus fees for all sends from Ethereum before the Bridge converter launches, the liquidity basket becomes active, and the bridge recognizes that launch. Once that happens, the remainder of $\frac{1}{2}$ the fees (VRSC) that are left over from the launch and did not go to miners and stakers or pay Verus fees for people sending from Ethereum.

The Verus Fee Pool and Rewards

Another technology and solution unique to Verus, the Verus Fee Pool technology goes live with PBaaS. What this means is that as people use the cross-chain, DeFi, or purchase/register IDs on the network, the miner or staker will put all fees into the on-chain "fee pool", and then take $\frac{1}{10}$ th of the pool in addition to the block reward. This gives everyone incentive to still prioritize transactions based on fees, while preventing validators from gaming the fee system by washing fees-in vs. fees-out or working to reorg/rewind to capture a particularly juicy block reward. Instead, this technology aligns all validator incentives with the health/proper operation of the network and creates a circular on-chain economy that can last well beyond block rewards.

A Historic Move Forward

For those who actually understand the challenges actual users have in crypto today, this release represents a historic move forward in public cryptographic networks and credibly and actually neutral infrastructure for society-wide human and AI collaboration and learning. As a community, this version really does realize the vision laid out in the original Verus Vision Paper, and at the same time provides much more than was described there. We are a community of individuals, and it is only because that is what we are that we have all arrived here together. The rest of the world is stuck on their Munchausen's protocols, trying to figure out how to share the front-running and back-running spoils taken from users, but Verus activation gives them a better path forward.

As a community, we do not promise, we describe a shared vision and welcome contributors from everywhere. Now that we, as a community, have delivered on the original phases described in the first vision paper together, it is time for each of us to also consider what we might personally want to build or be part of building over this incredible, geoscale, ID-enabled, fully decentralized platform. Our next efforts across the community should be to realize the promise of this network in the use cases we create, use cases that we will have an advantage creating on Verus over any other platform because it really is that much better.

Open-source, Rent-free, Scalable Public Infrastructure

Verus is an open-source decentralized blockchain protocol with proof-of-work and proof-of-stake as its consensus mechanism. It offers rent-free blockchain tools for creators and organizations to build products, services and systems.

Verus is a multichain protocol with strong focus on scalability, security and decentralization. It can scale to world demand, is proven % hash attack resistant and is community built - all coins in circulation are fairly mined and staked.

With Public Blockchains as a Service (PBaaS) anyone can launch scalable, interoperable, customizable and independent blockchains for public or private use. Create tokens and currencies on top of blockchains for any

use case. Get funding and create markets with protocol level built-in DeFi.

Protocol-level self-sovereign digital identities and namespaces are at the center of the Verus multichain protocol. Giving builders, communities, businesses and organizations tools never before seen. Everything happening on the Verus multichain protocol has aligned incentives with the miners and stakers of the worldwide network. This makes it one of the most secure protocols with opportunities to earn.

#Accessibility

Verus creates blockchain technology that is accessible for everyone.

No programming needed for blockchain, token and currency launches

Low protocol fees

Mining and staking accessible for everyone

#Scalability

Verus achieves practically unlimited scalability through its Public Blockchains as a Service.

Create use case specific blockchains and currencies

Move activity between chains to avoid congestion

#Security

Verus takes blockchain security to the next level.

% hash attack resistant through Verus Proof of Power

#Interoperability

Verus enables a world where all blockchains communicate with each other.

Cross-chain, cross-system transfers of all data types

Bridges to communicate with external blockchains

#Community

Verus is a community driven project in the true spirit of Bitcoin.

Fair launch

No ICO, no developer fees, no founder fees, no premine

Decentralized public blockchain

Coin Overview

Launch Date May

Coin Ticker VRSC

Average Block Time minute

Transaction Fee . VRSC

Max Supply , VRSC

Consensus Algorithm Verus Proof of Power

Hash Algorithm VerusHash .

Privacy Zcash Sapling

Verus Proof of Power

VerusPoP is a % proof-of-work, % proof-of-stake consensus algorithm. More information on the Verus miner and staker ecosystem.

#Hybrid Consensus

Verus Proof of Power, or VerusPoP, is a hybrid consensus algorithm which uses a statistical function that combines Proof of Work (PoW) and Proof of Stake (PoS) to validate each block by either PoW or PoS, while averaging to a target percentage of blocks being validated by each form of proof.

In short, it a unique consensus mechanism with % of all blocks validated by miners, and the other % by stakers.

#Attack Resistant

To successfully attack the Verus blockchain, more than % of the validation power is needed, called Chain Power. A % attack would require a combined value of over % of both the chain's hashpower and its coin supply. For technical information on VerusPoP read the whitepaper (opens new window).

VerusPoP provides a decentralizing effect on the network, incentivizing holders to keep nodes online to support the network. Even if a change in network hashrate happens, the PoW/PoS ratio stays the same: /%.

#VerusHash .

From the VerusPoP whitepaper (opens new window):

'VerusHash is specifically developed to deliver a competitive advantage for CPUs with GPUs. It is an exceedingly CPU-friendly long input hash function that uses the quantum-secure, short input Haraka V as its core compression algorithm. The result is the fastest known cryptocurrency hash algorithm available to modern CPUs and the only hash algorithm which enables today's CPUs and GPUs to compete on an economically comparable level.

Haraka V is designed as a short input hash to exclusively consume one chunk of bits and produce bits of a hash result. Utilizing Haraka V VerusHash takes any length of input and produces a bit hash result, unique to VerusHash, that also provides the same security guarantees as Haraka V. This makes VerusHash bit secure for classical computing attacks and bit secure against quantum computers for pre-image and second pre-image attacks.

To understand the VerusHash algorithm it helps to first separate the digest from the core. We then consider the Haraka V core as an abstract digest function that takes bits (bytes) of input and produces bits (bytes) of output. Given such a digest function, referred to as haraka, the most concise implementation of VerusHash, in any language to-date, is the following Python code for the VerusHash hash digest as follows:'

```
# verus_hash
def verus_hash(msg):
    buf = [] *
    length = len(msg)
    for i in range(, length, ):
        clen = min(, length - i)
        buf[:] = [b for b in msg[i:i + clen]] + [] * (- clen)
        buf[:] = haraka(buf)
    return bytes(buf[:])
#PoS Problems Solved
```

Verus' staking algorithm solves the two major theoretical issues undermining other PoS systems, Nothing at Stake and Weak Subjectivity by leveraging its smart transaction capabilities to remove any incentive to attempt cheating, making it a losing proposition. Read: How Verus Solved Proof of Stake's Two Biggest Problems: Nothing at Stake and Weak Subjectivity

Verus is a decentralized public blockchain, a community driven project in the true spirit of Bitcoin. Anyone can participate and contribute, no matter who you are or where you come from. Verus is:

Open
Borderless
Public
Neutral

Censorship Resistant

#Fair Launch

Verus had a fair launch, meaning that everyone had, and still has equal opportunity to collect its currency through mining and staking. For Verus this means:

No ICO has been held

No founder or developer fees

No premine

No commercial interests

No rent-seeking behavior

Launch without programming needed

Enables any user with VerusID to create their own token currency or even full fledged, multi-currency, VerusID-issuing % PoW/% PoS, % hash attack resistant blockchain that can send and receive from the Verus chain which launched it.

All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, VerusID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

#DeFi liquidity pools and fractional currency baskets

Any VerusID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from % to % backing.

The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

#Bridge converter launches

Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain.

Bridge converter currencies have the same flexibility as other fractional % asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross-chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

#Crowdfunding currency and blockchain launches

Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees.

The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

#Interoperable, multichain network

The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access.

There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Defining a currency

There are many options to choose from when defining your currency or blockchain. Combine them in the options parameter for different use cases.

Options Details

OPTION_FRACTIONAL Allows reserve conversion using base calculations when set

OPTION_ID_ISSUANCE Clear is permissionless, if set, IDs may only be created by controlling ID

OPTION_ID_STAKING All IDs on chain stake equally, rather than value-based staking

OPTION_ID_REFERRALS If set, this chain supports referrals

OPTION_ID_REFERRALREQUIRED If set, this chain requires referrals

OPTION_TOKEN If set, this is a token, not a native currency

OPTION_SINGLECURRENCY For PBaaS chains or gateways to potentially restrict to single currency

OPTION_GATEWAY If set, this routes external currencies

OPTION_PBAAS This is a PBaaS chain definition

OPTION_GATEWAY_CONVERTER This means that for a specific PBaaS gateway, this is the default converter and will publish prices

OPTION_GATEWAY_NAMECONTROLLER When not set on a gateway, top level ID and currency registration happen on launch chain

OPTION_NFT_TOKEN Single satoshi NFT token, tokenizes control over the root ID

Need help setting up a currency launch? □

Go to the Verus Discord #pbaas-development channel. The community is happy to assist! (opens new window)

#VerusID Namespace

To create a currency (or blockchain) of a specific name, you need a VerusID of the same name. The controller of this VerusID is the only one who can create a currency of that name, and they can only do so once.

#Example scenarios

So, let's hypothetically assume we have VerusIDs, one named gold@, one named mycoin@, and one named bob@.

We would like to have one currency, gold@, that we somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

We also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, 'my', offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

```
#Defining GOLD currency
```

Of course, since this is a test currency, we send ourselves some to start. We could define the currency gold as follows:

```
./verus -chain=VRSCTEST definecurrency'{
  'name': 'gold',
  'options': '',
  'currencies': ['vrsctest'],
  'conversions': [],
  'minpreconversion': [],
  'preallocations': [{'david@': ..}]
}'
```

The identity of the currency must be funded with at least VRSCTEST before sending the transaction returned from this command to initiate a currency launch that will start at blocks from when it was made (default), and that must have VRSCTEST preconverted at . VRSCTEST per GOLD in order to launch.

All of this happens as part of the mining process, since mining the blocks that launch a currency earn the .% conversion fees of participation in the launch, converting VRSCTEST to GOLD.

```
#Converting VRSCTEST to GOLD
```

We could send the following command before the block where GOLD token launches. After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, we could issue the command:

```
./verus -chain=VRSCTEST sendcurrency '*''{}{
  'address': 'bob@',
  'convertto': 'gold',
  'preconvert': '',
  'amount': ''
}'
```

This would effectively park our conversion until the token launches, at which point, we will either find . GOLD in our wallet, or we will have our VRSCTEST back.

Assuming it launches, and we later want to create MYCOIN, which can be converted to with either GOLD or VRSCTEST.

```
#Defining MYCOIN currency
```

Let's define MYCOIN with:

```
./verus -chain=VRSCTEST definecurrency'{
  'name': 'mycoin',
  'options': '',
  'proofprotocol': '',
  'currencies': ['vrsctest', 'gold'],
  'minpreconversion': [..],
  'initialsupply': ''
}'
```

```
#Minting new MYCOIN
```

In 'mycoin', we set proofprotocol to , which is PROOF_CHAINID. That means that the controller of the mycoin@ VerusID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency 'mycoin@' '{}{
  'address': 'bob@',
  'currency': 'mycoin',
  'mintnew': '',
  'amount': ''
}'
```

Defining a PBaaS-Blockchain

There are many options to choose from when defining your blockchain. Combine them in the options parameter for different use cases.

Options Details

OPTION_FRACTIONAL Allows reserve conversion using base calculations when set

OPTION_ID_ISSUANCE Clear is permissionless, if set, IDs may only be created by controlling ID

OPTION_ID_STAKING All IDs on chain stake equally, rather than value-based staking

OPTION_ID_REFERRALS If set, this chain supports referrals

OPTION_ID_REFERRALREQUIRED If set, this chain requires referrals

OPTION_TOKEN If set, this is a token, not a native currency

OPTION_SINGLECURRENCY For PBaaS chains or gateways to potentially restrict to single currency

OPTION_GATEWAY If set, this routes external currencies

OPTION_PBAAS This is a PBaaS chain definition

OPTION_GATEWAY_CONVERTER This means that for a specific PBaaS gateway, this is the default converter and will publish prices

OPTION_GATEWAY_NAMECONTROLLER When not set on a gateway, top level ID and currency registration happen on launch chain

OPTION_NFT_TOKEN Single satoshi NFT token, tokenizes control over the root ID

Need help setting up a blockchain launch? ☺

Go to the Verus Discord #pbaas-development channel. The community is happy to assist! (opens new window)

#VerusID Namespace

To create a blockchain of a specific name, you need a VerusID of the same name. The controller of this VerusID is the only one who can create a blockchain of that name, and they can only do so once.

#Examples of blockchain launches

#Blockchain

```
./verus -chain=VRSCTEST definecurrency '{
  'name':'PBaaSChain',
  'options':,
  'currencies':['VRSCTEST'],
  'conversions':[],
  'eras':[
    {
      'reward':,
      'decay':,
      'halving':,
      'eraend':,
    }
  ],
  'notaries':[
    'Notary@',
    'Notary@',
    'Notary@'
  ],
  'minnotariesconfirm':,
  'nodes':[
    {
      'networkaddress':'...:',
    }
  ]
}'
```

```
'nodeidentity':'Node@'
},
{
'networkaddress':'....',
'nodeidentity':'Node@'
}
],
'gatewayconvertername':'Bridge',
'gatewayconverterissuance':
}
{
'currencies':['VRSCTEST','PBaaSChain','USD'],
'initialcontributions':[.,.],
'initialsupply':
}
#Blockchain
./verus -chain=vrsctest definecurrency ^{
'name':'v',
'options':,
'currencies':['vrsctest'],
'preallocations':[
{
'allnotary@':
}
],
'conversions':[],
'eras':[
{
'reward':,
'decay':,
'halving':,
'eraend':
}
],
'blocktime':,
'idregistrationfees':,
'notaries':[
'allnotary@',
'allnotary@',
'allnotary@'
],
'startblock':,
'minnotariesconfirm':,
'nodes':[
{
'networkaddress':'....',
'nodeidentity': 'allnotary@'
},
{
'networkaddress':'....',
'nodeidentity':'allnotary@'
},
{
'networkaddress':'....',
```

```

'nodeidentity':'allnotary@'
}
],
'gatewayconvertername':'Bridge',
'gatewayconverterissuance':
} '
'{
'currencies':['VRSCTEST','v'],
'initialcontributions':[[],],
'initialsupply':
}'

Defining a currency with a : mapping of an ERC-
When defining a currency it can be mapped to an ERC- :. The currency on
Verus and the ERC- on Ethereum are then always interchangeable to .
↳ Verus-Ethereum Bridge
The Verus-Ethereum bridge is the first of its kind. All tokens and
currencies flowing over the bridge are never in anyone's custody, and are
proven and verified by consensus rules.

🔗 Access the Verus-Ethereum Bridge (opens new window) (Goerli testnet)

#Defining the currency
To create a currency of a specific name, we need a VerusID of the same
name. The controller of this VerusID is the only one who can create a
currency of that name, and we can only do so once.
The cost for a VerusID on the Verus testnet is VRSCTEST (or when using a
referral). The cost to launch a currency is VRSCTEST. Before launching we
need to have enough VRSCTEST in the namespace VerusID.
In our example we have a namespace MyUSDC with which we want to launch a
currency that is mapped to the Ethereum USDC ERC- (on Goerli testnet, see
contract address (opens new window)).
Below is the command to map a currency : to an ERC- on Ethereum. The
address field is the Ethereum smart contract address of the ERC- we want
to map to.

./verus -chain=VRSCTEST definecurrency '{
'name':'MyUSDC',
'options':,
'systemid':'veth',
'parent':'vrsctest',
'launchsystemid':'vrsctest',
'nativecurrencyid':{
'type':,
'address':'xDCBadccCbfBa'
},
'initialsupply':,
'proofprotocol':
}'

After we put in the command, we get returned a HEX. We use this HEX to
launch the currency on the network. Use the command below to launch the
currency:
./verus -chain=VRSCTEST sendrawtransaction 'HEX'
Now we have to wait a few blocks for the currency to be available on the
network.

#Export to Ethereum
The last step is to export the currency to Ethereum so we can see it
there too.

```

Export a currency to Ethereum (as ERC-)

A currency on Verus (testnet) can be exported to Ethereum as an ERC-. The currency can then be used in the complete Ethereum ecosystem, and on the Verus network. Sending to and from Verus and Ethereum couldn't be easier.

↔ Verus-Ethereum Bridge

The Verus-Ethereum bridge is the first of its kind. All tokens and currencies flowing over the bridge are never in anyone's custody, and are proven and verified by consensus rules.

↗ Access the Verus-Ethereum Bridge (opens new window) (Goerli testnet)

#Exporting the currency

Now, let's export a currency from Verus to Ethereum as an ERC- over the non-custodial bridge. We must have enough funds to pay for the export.

The export fee is: ~ . vETH or ~ VRSCTEST

The command to export a currency to Ethereum as an ERC-:

```
./verus -chain=VRSCTEST sendcurrency 'myVerusID@' '[{  
'address':'xADCEEFBcbCefBDEDADBE',  
'currency':'myCurrency',  
'amount':,  
'exportto':'veth',  
'exportcurrency':true,  
'feecurrency':'veth'  
}]'
```

Let's break the command down.

myVerusID@ is the from- and change-address. The fee to pay for the export comes from here, and if you paid too much fees the rest will be returned here.

address is the Verus-Ethereum smart-contract bridge address. Don't change this.

currency is the name of the currency you wish to export as an ERC-.

feecurrency is the currency the export fees are paid in. Omit this to pay the fees in VRSCTEST.

Wait for notarization

After the bridge has been notarized to the blockheight where you have exported the currency, you can choose it from the token dropdown on the bridge website

Verus is a rent-free blockchain protocol that has a decentralized economy. Costs for services paid to the protocol are distributed to the miners and stakers. There are no developer fees, and no rent-seeking profiteers.

Verus has a naturally decentralized miner ecosystem. VerusHash . is the mining algorithm that is specifically developed for Verus. The algorithm equalizes mining fairness across hardware classes.

To mine Verus no specialized mining equipment is necessary. Anyone can start mining with consumer-grade hardware. Mining is made especially easy for starters, since they can start directly from within Verus Desktop, without any additional software requirements.

Additionally, when PBaaS goes live on mainnet, miners can choose to mine up to PBaaS-chains simultaneously, without losing any of their original hashing power

Anyone can start staking Verus, even with the smallest amount possible, e.g. . VRSC. Although it could take many years before you win a block with such a tiny amount.

Verus has a powerful and fair staking system. No threshold required and no registrations necessary.

Miners and stakers are rewarded for their efforts to keep the network secure. They earn block rewards. The Verus block reward emission schedule:

Block Height (& Start Date) Block Rewards Coin Emission

The maximum circulating supply of all coins is .. VBSC.

* - Block Rewards □

From block height , on, all block rewards of and VRSC were timelocked and then unlocked at random block heights. The last of those rewards were unlocked at block height ...

Fees are generated from users paying for services and interacting with the protocol. These fees are processed in the blocks and earned by miners and stakers.

Protocol Activity Fees*

Processor Recovery fees
Various ID Registration mainnet VRSC

versus ID Registration manner VRSC
PBaaS chain Launch testnet VRSC++

FBaas chain launch testnet , VRSC
Currency/Token Launch testnet VRSC

Currency/Token Launches Compliance Dashboard

Conversion Fee Testnet Implementation Fee - VDSC

* Testnet fees are subject to change. **Half the fee goes to the miners and stakers of the newly launched PBaaS-chain.

All the fees generated by the protocol are collected in the fee pool. With each new block, % of the fee pool is added on top of the block rewards. It is expected that fees outnumber block rewards, making mining and staking in the Verus ecosystem a profitable and competitive endeavor.

Mine solo through Verus Desktop to receive full block rewards. Depending on hashrate it may take a while before winning a block. No additional software is needed to get started

Participate in pool mining to receive regular rewards. You will need to set up a few things before you can start.

Solo mining Pool mining
Full Node yes no
Regular Rewards no yes
Setup Difficulty easy intermediate

Mine Verus with various devices. Profitability indication means the electricity usage vs hashrate.

Device Profitability Indication
CPU (processor) high
GPU (graphics card) medium
Mobile Phone high
ARM (not RP) high
FPGA not possible
ASIC not possible

You would need a software and a pool url to mine veruscoin, Some of the softwares that you can download is NheqMiner from veruscoin github organisation, CCMiner by Monkins or SRBMiner.

For mining with your GPU you would need CCMiner GPU miner by Monkins.

Pool Name Fee %
Verus Pool ([opens new window](#)) Fees donated to the Verus foundation %
LuckPool([opens new window](#)) %
ZergPool([opens new window](#)) .%
CiscoTech([opens new window](#)) %
LePool([opens new window](#)) %
Zhuaao([opens new window](#)) %
AlphatechIT([opens new window](#)) .%
Wattpool([opens new window](#)) .%
Data([opens new window](#)) .%

Start mining with your phone.
Download VerusBox Download on Google Play Or
VerusMiner APK Download

Staking is accessible for everyone. Use Verus holdings to secure the network. It does not matter how much \$VRSC you have, there are no minimum requirements to start staking.
On the Verus blockchain UTXOs (unspent transaction outputs) are staking, not balances. A large UTXO has more chances of winning a block than a small UTXO.

There are a few rules you need to know before you can start staking.
Rules
Wallet Running
Full Node Required (native mode in Verus Desktop)

Staking Enabled
UTXO Eligible After Blocks
Minimum of . VRSC in Wallet

community operated staking pools are available.

HIGH RISK

When joining staking pools, you give away custody of your coins. You don't own your coins anymore and will have to trust the pool operator. Do at your own risk.

Imagine a new world. Everything you value is safe and secure. Everything you do, see, say and hear can be verified if you choose. Everyone you meet can present a verifiable reflection of the sum of their lifetime experiences. A society where your values, votes, and voice can be immutably expressed and secured by your community. All your data is verified, private, protected, and provable in enabling exchanges of self-evident truths. A society where equity of opportunity is available to all. Kindness can be exchanged without fear of fraud or humiliation. All exchanges can be certified, smart, and secure without the need to rely on costly intermediaries. Imagine a worldwide network as scalable and as accessible as today's internet. Capable of processing financial transactions and dollars, euros, Bitcoin, Ethereum, all major currencies. Even custom currencies or auto-rebalancing multi-currency baskets, defined by any organization or individual. This network is not run by any single company or government. Instead, it's a network of independent networks. All connected by a strong cryptographic protocol that ensures all funds of all currencies are accounted for flawlessly. And by doing so, enables anyone in the world to contribute to powering it while enhancing its security and take a reward for doing so. Not only can anyone participate in this network by helping it run, the network makes it easy for anyone worldwide to create new fractal subnetworks that are launched from the network. And after launch, are completely independent from, yet fully connected and interoperable with all other parts of the network. With each new community, country, company or cause that creates its own subnetwork, the network of networks does not congest. But instead, it gains the independent processing of a new subnetwork along with its additional scale and power. Imagine that all the network could be accessed and used by people, organizations, businesses and governments easily, securely, and without any complex programming. You've probably already guessed that I'm not talking about Bitcoin, which is a decentralized tour to force, but only a single currency network. I'm also not talking about any near-term evolution of Ethereum or Ethereum-like networks, which are fundamentally single currency networks as well. Yes, I did just say that Ethereum is a single currency network. That may surprise some people so let me explain. ETH is in fact the only currency that has provable accounting as part of Ethereum's blockchain protocol. It's only contracts. Code, written, copied, and or modified by many, many separate devs and companies, interacting on a simulated computer, the Ethereum virtual machine. All of which hopefully follow best practices and conventions that enable Ethereum to transact in currencies besides ETH itself. The issue is that these contracts define currencies and engage in currency transfers between users and each other without any network-wide protocol that verifies those transfers correct accounting. All contracts may also expose arbitrary functions to other contracts

while operating together in what is called a touring complete virtual machine, meaning they can effectively do anything, or make any mistake that can be written into code. Defining accounting for and transacting in any currency besides ETH on the Ethereum blockchain as all similar VM-based blockchains is a matter of how multiple contracts interact, a black box operation to the protocol which cannot be verified by design. That architecture invented by Ethereum years ago and now used by countless protocols, leads to other risks as well, such as multi-step approval and spend operations of non-native currencies, which are quite complex for users and can expose them to fishing scams that request permission to take assets under the guise of performing another action. This new form of fishing attack on users of the most popular centralized NFT marketplace recently took about \$ million from unsuspecting users, but total losses from contracts far exceed millions on a regular basis. Estimates of losses to hackers from these contracts not including fishing. In just the last year, range from . to over \$ billion. All of that on blockchain networks running a similar VM-based architecture as Ethereum. Perhaps it's time for a better design. Imagine a multi-blockchain network with a real multi-currency protocol that verifies all transactions in all currencies as securely as Bitcoin and Ethereum verify transactions that transfer BTC or ETH. The good news is that now you can do more than just imagine. The system I've described is fully functional today on the various test net and being prepared by contributors across the community now for its imminent main net upgrade. All technologies are currently available for application development by those early adopters who understand the opportunities in leveraging a new disruptive paradigm. It's still officially a test net, but is fully decentralized, functional, and accessible worldwide for launching test currencies, independent and connected blockchains, on-chain exchanges of currencies, and contracts, and more. Various itself is not a new cryptocurrency. In fact, it's an independent blockchain launched May , , with no ICO, no pre-allocation of funds, and no development fee or rent built into the protocol whatsoever. Various is complete with both transparent and zero-knowledge privacy protocols, and in ecologically efficient, provably % hashtag resistant consensus protocol. It's an evolution of knock-a-modo consensus that incorporates proof of stake as well, and the only blockchain with self-sauvrent, revocable, recoverable, and interoperable addresses, and identities built into the core protocol. Various identities are supported across all transaction types, compatible with other provable ID technologies, and serve as a foundation for all currencies and blockchains on the network. Applications of the Various Network extend far beyond D-Apps of blockchain today. Any individual, project, organization, or community, needing an easy-to-use, reliable, decentralized, rent-free currency, fundraising platform, voting network, automated accounting system, or self-sauvrent, revocable, recoverable identities, and a governance model can connect to the Various Internet of Value and get started with no programming required. Solutions that used to take weeks or longer to design and code with solidity experts can now be done better, faster, and more securely, no programming needed. With Various's multi-chain model, scale challenges of today can be naturally solved by scaling out as needed, whether requirements call for centralized corporate, open-government, or fully-decentralized applications. In addition to this new technology, disruptive even among modern blockchains, the open community-driven genesis and approach

enables Various to be the first, and only rent-free, decentralized financial network capable of hosting multiple independent, yet connected real-world economies. With thousands of connected blockchains, millions of fungible currencies, and billions of users. Various is an open protocol, implemented in fully open source, designed to connect, interoperate with and serve as either independent or augmenting infrastructure for existing, or completely new, and more scalable, provable, fair, and trustworthy systems than what we have anywhere today. You can download the free and open-source Various Wallet from Various.io. Connect to the public test net, drop by the Discord for some free test currencies, and get started. Without any actual programming, just commands, you will be able to create revocable recoverable identities that you can then use to launch currencies, blockchains, liquidity baskets, automatically bridged ERCEAR, Ethereum currencies, and multi-chain applications using all currencies available on Various or Ethereum.

Unspend transaction output, or in short, UTXR. Let's visualize a part of a blockchain. We fill three of them up and leave a fourth new block empty for some new transactions. This is Alex. Alex has a various wallet and has received various coins so far, in four separate amounts. Now, these four separate amounts is essentially what UTXR is all about. Let's use two balls to explain. The first one shows the total amount of coins in your wallet, pictured is water in the bowl. The second one shows four water capsules in their bowl, each holding their own amount. They don't mix in the second bowl. Now let's have a look with that means. Meet Mary, a friend of Alex. She recently created a various wallet, but does not have any coin share. So Alex wants to send Mary of his coins to get her started. The various wallet will try to use the smallest UTXR values first in order to fill a desired coin transaction size, which in this case means it needs all four UTXRs to get to . When Alex sends the transaction, all four UTXRs will be spent and turns them into STXRs, spent transaction outputs. This is just a background process required and you will never see it. Next, two new transactions are created on the blockchain, for two new UTXRs, one for the various going to Mary, and one for sending the change of the transaction back to Alex. The transactions will go into the latest block in the blockchain so they can be confirmed by all the notes. I will explain how that all works in another animation. So all that is left to do is to send the various coins to Mary's wallet and send the change of the transaction back to Alex's wallet, and this of course is all done automatically.

Verus is a community project that was fair launched May , . No ICO was held, no coins were pre-mined, and there are no founder or developer fees. Varus' history comes by way of Bitcoin, Zikash and Komodo, building some of the foundations of blockchain, privacy, the lead proof of work and smart transactions. Varus' consensus algorithm is called proof of power. Proof of power is a split of % proof of stake and % proof of work using Varus' hash, which is optimized for CPU mining, giving everyone a fighting chance on a level playing field. Varus' proof of power is verifiably % hash attack resistant. Varus' ID is the first self-sovereign decentralized identity system of its kind. Firstly, it takes your standard long and impossible to remember address and allows you to create a friendly name, such as John Smith, that you can use to send and receive funds. Your name can include almost all Unicode characters,

including emoji. If you ever lose access to your private keys or they become compromised, another ID can be used to securely revoke and recover your name and the fund stored on it. Along with these great features of Varus' ID, you can use the ID to create verifiable digital signatures, decentralized attestations, send and receive private messages, and more. Varus' vault gives users the ability to have a custom defined time lock of their assets that they can still stake with or revoke and recover if needed. Another great feature of Varus is the public block chain as a service, or PVAS, giving anyone the ability to launch tokens or full-interoperable block chains with no programming needed. You can set the price for identities on your chain as well as coin supply and emission, and it comes with all of the features that Varus offers, making it easy for anyone to start a project on the Varus platform, or to start their own block chain with a few clicks of a button. Half of the fee for block chain creation goes into the fee pool to be paid to miners and stakeholders over time, and the other half goes into the fee pool of the newly created block chain. An exciting feature of PVAS creation is the ability to do crowdfunding launches, where you can set a minimum collection for that currency to launch. Offer pre-launch participation discounts, pre-conversion currency carve-outs, or price-neutral launch pre-allocations, all with the ability to automatically refund if the fund raise in conditions are not met. PVAS chains are fully interoperable, with cross-chain spends and currency conversions, Varus and PVAS chains can also be merged mine together, up to chains at once with all your hash power securing every chain. This helps small projects get hash power to secure the network for a lower cost. The first bridge is a decentralized bridge to Ethereum for ETH and ERC tokens, with other cross-platform capabilities in the future. This opens up the Varus ecosystem to value from other block chains, makes tokens and coins from the Varus network available on other platforms, and allows the storing of assets from other chains in a various vault protected Varus ID. The benefits of having layer 2 on a UTXO blockchain is the ability to have consensus and reward models that are aligned with the economic activity happening on the chain. Varus uses a parallel processing model that solves all 2nd layer transactions in a block at the same time, and with the same pricing for all trade volumes and directions in that block. This, combined with the fee structure, provides minor extracted value resistance, increased security, and conversion pricing that is fair and more stable than other systems. Find out more at Varus.io.

Our internet and economies have become globally centralized. Our data, bought and sold to the highest bidder. Media platform sensorized in the technology engineer to funnel and extract value and commodifying control our behavior. With the emergency blockchain technology, we have a chance to develop new systems in engineer than for more sovereignty, privacy, and freedom. But many blockchain projects fall prey to the same patterns of centralization. Built in silos, they lack scalability and few teams build bridges to work between them. Now, with this understanding, we have a better way forward. The next evolution in decentralized tech is here. Varus is a truly free, open-source blockchain protocol designed for privacy, safety, open participation, and unlimited scalability. Varus is not a business or bank. To ensure integrity and impartiality, the secure public network is supported by a worldwide community of contributors, including a nonprofit foundation. Varus combines the latest in

cryptography and zero-knowledge privacy to enable its vision of public blockchains as a service. Decentralized financial and communications tech that can scale to a new internet of value and data exchange. With Varus, any individual or organization can create recoverable self-sauver identities, own their data and build and run apps and business software on any PC or mobile platform. In coming soon on Varus, individuals and organizations can tokenize assets and create currencies, create a shared economy by tokenizing a house for example, or bootstrapping crowd-funded project with fully liquid algorithmic conversion across multiple blockchains and crypto currencies. Individuals and communities can create their own micro economies and connect to a worldwide network about their public blockchains, organizations, and economies, and they don't have to pay anyone to do these things or give away their rights to their data. Now is the time to build. These are the tools we need to do that. Build with us. Varus. Truth and privacy for all. Learn more and join the conversation on Discord.

etwork Upgrade

This is the first release candidate (RC1) for the next major upgrade to the Verus network. It is considered fully ready for testing on both VRSCTEST testnet and VRSC mainnet. While this release is likely to work fine on mainnet through the coming activation, it is not yet an official release that you should expect to support mainnet activation. If you install this version, please make sure to upgrade to an official release before network activation, which will happen at block 800200, expected to be mined or staked on December 15th.

This release enables two technology upgrades, one that would be considered big news to most cryptocurrency projects, an improvement to the FPGA-equalizing hash algorithm, and a revolutionary, new decentralized identity technology, Verus IDs, that will disrupt today's centralized systems with the most secure, quantum ready, fully decentralized, self-sovereign digital identity system in the world.

Verus ID

Verus ID includes built-in privacy at the core through integration of zk-SNARKs, and provides a revolutionary improvement to blockchain address security, making it possible to actually recover your money after losing your keys, secure against identity and key theft, prove things about yourself without having to show more details than needed, and transfer assets to heirs as part of your estate. This release also introduces a new smart transaction technology that is unique to the Verus network and replaces its use of Komodo compatible crypto-conditions. The Verus ID system was made possible through the use of Verus Smart Transactions, which will be available for everyone to use on their own blockchains in the upcoming PBaaS network upgrade. Verus Smart Transactions use standard Bitcoin style serialization rather than the ASN.1 used in crypto conditions, which makes support on lite or mobile wallets simpler to implement than the crypto-condition protocol, which is still used on the Verus network for Stake Guard.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Verus ID - A Better Blockchain Identity Technology

If you've been around crypto for even a little while, you'll start to hear things like "not your keys, not your coins", "lose your keys, lose your coins". If you lose your keys, no one will ever be able to recover your money.

In fact, trading off decentralized, censorship resistant public systems for this risk to your cryptocurrency assets is such a fundamental issue in blockchain systems today that it is considered an acceptable price for having the benefit of full control over your own funds and blockchain assets.

Verus ID provides an elegant solution to these previously unsolved, fundamental problems, and also provides quantum-ready friendly identity names and blockchain funds addresses as self-sovereign, revocable, recoverable identities that provide the following advantages over today's blockchain systems:

1. Identity and currency address are one and the same. Your friendly name can be used as both a reference to who you are and an on-chain destination for any funds transfer.
2. Each identity must have a set of addresses (1 or more) and a minimum number of signatures required from those addresses to spend or sign on behalf of an identity. Each identity also refers to two additional identities that have authority over:
 - **Primary** - this is the self identity and may modify any part of the identity except its name, its parent, which is derived from the blockchain on which it is defined, and unless it is also one of the two other authorities, it may not modify any data under their control, including the identity of the other authorities once specified.
 - **Revocation** - this identity may revoke the primary identity, which will also revoke access to all funds and transactions under its control, but it may not spend any funds or sign on behalf of the identity.
 - **Recovery** - this authority may redefine/recover a revoked identity, but it may not change the revocation authority, nor may it modify an unrevoked identity at all.
3. Identities are transferable by defining them with primary addresses with private keys, which are under another party's control. When identities are transferred, the revocation and recovery authorities may or may not be modified as well, as long as the party updating the ID has the authority to do so.
4. If an identity is transferred from one party to another, all funds in transactions that are under the control of that identity will also transfer to the party to which control is being transferred. In the native wallet, these changes are automatically reflected in the wallet balance and visibility of the balance of an identity using `z_getbalance identity@"`.
5. If later, a new address type or signature algorithm, such as one that is quantum secure is added to the Verus blockchain, all transactions sent to an identity that is updated to be under the control of the new address type will be subject to its spend conditions as well. That means that if you have 1000 transactions sent to your identity name/address, an upgrade to a quantum secure address, when they are available, will retroactively secure all 1000 transactions under the control of that address.

Making a new identity

Arguably, the hardest part of the new Verus identity technology is just making your first identity, especially without a GUI. Once you have your own identity, a lot becomes easier, not the least of which is the ability to send money to a friendly name instead of a base58 blockchain address or hash. For example, once you have your identity, either of the following commands will work as well as just about any other CLI command that takes an address:

```
./verus -chain=VRSCTEST z_sendmany bob@ '[{"address":"alice@", "amount":100}]'
```

or

```
./verus -chain=VRSCTEST sendtoaddress alice@ 100
```

Making a Verus ID requires two steps, steps that are designed to prevent potentially selfish miners from replacing your purchase of an ID from the Verus blockchain with their own purchase of the same name instead. To prevent this name “front running”, you must first commit to a name that you wish to have in your identity as its friendly name. Friendly names are unique to a blockchain, so if someone already has a specific name you must have, you will need to either be the first to purchase that name from the blockchain or acquire it from the controlling individual or organization.

Creating a name commitment registers your commitment, but keeps the name completely secret until you register it and reveal your commitment. When you register a name, you must pay 100 VRSC or 100 VRSCTEST, unless you have a referral from someone who already has an identity. With a referral, the price is 80 VRSC or 80 VRSCTEST. When you register an identity with the CLI wallet, the correct amount of money, 100 VRSC without and 80 VRSC with a referral is automatically spent along with your registration. The funds for a referral are distributed as follows:

- 20 VRSC to the referrer’s ID
- 20 VRSC to the referrer’s referrer’s ID
- 20 VRSC to the referrer’s referrer’s referrer’s ID
- 20 VRSC to the miner of the block in which the ID is mined

100% of all funds that are not paid to referrers go to miners or stakers of the blockchain.

To create an identity:

Create your commitment with the following command:

```
./verus -chain=VRSCTEST registernamecommitment Name youraddressorid  
referrerfriendlyname@
```

That will respond with something like:

```
{  
  "txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
  "namereservation": {  
    "name": "Name",  
    "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
    "referral": "iSJKp1hvn51Zg2Y6FBKjzLs9AAy7fomWN",  
    "parent": "",  
    "nameid": "iDXx9FPrAS5k2XCGss6FFmDQQMsts63uUg"  
  }  
}
```

You then wait for the commitment to be mined into a block, then use the information returned above as follows:

```
./verus -chain=VRSCTEST registeridentity
'{"txid":"377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",
"namereservation": {
    "name": "Name", "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",
    "referral": "referrerfriendlyname@"}, "identity": {"name": "Name",
"primaryaddresses": ["RKo5u8N1sStZu81fU8kaxhoDcFcJmNEwSp"], "minimumsignatures": 1,
"privateaddress": "zs14y0aa096em2as6n4fauyumqeywz45rfpze38c39fksgtjsac9vmms7fmstc
ylpaalm7rseu8838"}'}
```

As long as you have enough funds and you are registering a new identity on the blockchain, you will be able to use your new identity in place of an address as soon as the transaction created by the registration command is mined into the blockchain.

You may list the identities in your wallet with the command `listidentities`, and you may display specific identities with the command `getidentity`. If you have the authority, or use...

Assets 8

v0.5.9-24

 [Asherda](#)

[v0.5.9-24](#)

[89d2db0](#)

[v0.5.9-24](#) Pre-release

This release is intended for use in the VRSCTEST network. Mainnet use is still being tested.

Notable Changes

- Add blockchain rule to consider any identities without primary addresses or nonzero minimum number of signatures as invalid.

Verus ID - A Better Blockchain Identity Technology

If you've been around crypto for even a little while, you'll start to hear things like "not your keys, not your coins", "lose your keys, lose your coins". If you lose your keys, no one will ever be able to recover your money.

In fact, trading off decentralized, censorship resistant public systems for this risk to your cryptocurrency assets is such a fundamental issue in blockchain systems today that it is considered an acceptable price for having the benefit of full control over your own funds and blockchain assets.

Verus ID provides an elegant solution to these previously unsolved, fundamental problems, and also provides quantum-ready friendly identity names and blockchain funds addresses as self-sovereign,

revocable, recoverable identities that provide the following advantages over today's blockchain systems:

1. Identity and currency address are one and the same. Your friendly name can be used as both a reference to who you are and an on-chain destination for any funds transfer.
2. Each identity must have a set of addresses (1 or more) and a minimum number of signatures required from those addresses to spend or sign on behalf of an identity. Each identity also refers to two additional identities that have authority over:
 - **Primary** - this is the self identity and may modify any part of the identity except its name, its parent, which is derived from the blockchain on which it is defined, and unless it is also one of the two other authorities, it may not modify any data under their control, including the identity of the other authorities once specified.
 - **Revocation** - this identity may revoke the primary identity, which will also revoke access to all funds and transactions under its control, but it may not spend any funds or sign on behalf of the identity.
 - **Recovery** - this authority may redefine/recover a revoked identity, but it may not change the revocation authority, nor may it modify an unrevoked identity at all.
3. Identities are transferable by defining them with primary addresses with private keys, which are under another party's control. When identities are transferred, the revocation and recovery authorities may or may not be modified as well, as long as the party updating the ID has the authority to do so.
4. If an identity is transferred from one party to another, all funds in transactions that are under the control of that identity will also transfer to the party to which control is being transferred. In the native wallet, these changes are automatically reflected in the wallet balance and visibility of the balance of an identity using "z_getbalance identity@".
5. If later, a new address type or signature algorithm, such as one that is quantum secure is added to the Verus blockchain, all transactions sent to an identity that is updated to be under the control of the new address type will be subject to its spend conditions as well. That means that if you have 1000 transactions sent to your identity name/address, an upgrade to a quantum secure address, when they are available, will retroactively secure all 1000 transactions under the control of that address.

Making a new identity

Arguably, the hardest part of the new Verus identity technology is just making your first identity, especially without a GUI. Once you have your own identity, a lot becomes easier, not the least of which is the ability to send money to a friendly name instead of a base58 blockchain address or hash. For example, once you have your identity, either of the following commands will work as well as just about any other CLI command that takes an address:

```
./verus -chain=VRSCTEST z_sendmany bob@ '[{"address":"alice@", "amount":100}]'
```

or

```
./verus -chain=VRSCTEST sendtoaddress alice@ 100
```

Making a Verus ID requires two steps, steps that are designed to prevent potentially selfish miners from replacing your purchase of an ID from the Verus blockchain with their own purchase of the

same name instead. To prevent this name “front running”, you must first commit to a name that you wish to have in your identity as its friendly name. Friendly names are unique to a blockchain, so if someone already has a specific name you must have, you will need to either be the first to purchase that name from the blockchain or acquire it from the controlling individual or organization.

Creating a name commitment registers your commitment, but keeps the name completely secret until you register it and reveal your commitment. When you register a name, you must pay 100 VRSC or 100 VRSCTEST, unless you have a referral from someone who already has an identity. With a referral, the price is 80 VRSC or 80 VRSCTEST. When you register an identity with the CLI wallet, the correct amount of money, 100 VRSC without and 80 VRSC with a referral is automatically spent along with your registration. The funds for a referral are distributed as follows:

- 20 VRSC to the referrer’s ID
- 20 VRSC to the referrer’s referrer’s ID
- 20 VRSC to the referrer’s referrer’s referrer’s ID
- 20 VRSC to the miner of the block in which the ID is mined

100% of all funds that are not paid to referrers go to miners or stakers of the blockchain.

To create an identity:

Create your commitment with the following command:

```
./verus -chain=VRSCTEST registernamecommitment Name youraddressorid  
referrerfriendlyname@
```

That will respond with something like:

```
{  
  "txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
  "namereservation": {  
    "name": "Name",  
    "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
    "referral": "iSJLKp1hvn51Zg2Y6FBKjzLs9AAy7fomWN",  
    "parent": "",  
    "nameid": "iDXx9FPrAS5k2XCGss6FFmDQQMsts63uUg"  
  }  
}
```

You then wait for the commitment to be mined into a block, then use the information returned above as follows:

```
./verus -chain=VRSCTEST registeridentity  
'{"txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
  "namereservation": {  
    "name": "Name", "salt":  
    "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
    "referral": "referrerfriendlyname@"}, "identity": {"name": "Name",  
    "primaryaddresses": ["RKo5u8N1sStZu81fU8kaxhoDcFcJmNEwSp"],  
    "minimumsignatures": 1,  
    "privateaddress": "zs14y0aa096em2as6n4fauyumqeywz45rfpze38c39fksgtjsac9vmmms7fmstc  
    ylpalm7rseu8838"}}'
```

As long as you have enough funds and you are registering a new identity on the blockchain, you will be able to use your new identity in place of an address as soon as the transaction created by the registration command is mined into the blockchain.

You may list the identities in your wallet with the command `listidentities`, and you may

display specific identities with the command `getidentity`. If you have the authority, or use identities under your control as revocation or recovery authorities for other identities, you may also revoke and recovery identities as well.

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

```
~/ .komodo/VRSCTEST  
~/ .verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST  
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST  
"%APPDATA%\VerusTest
```

Launching the testnet:

```
./verusd -chain=VRSCTEST
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/5cc801f027d86a59d4d70de91697fa86d96ca5034f957c81c0a2f7895a5a5677/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/0a989ab93e059fc778475498459b9de0484557ee8f44feb2fe418c58d976c38a/detection>

Windows:

<https://www.virustotal.com/#/file/2a661915722798ba65ca84990707c4f6cea1f390a79f717adc83c8b50861b581/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.5.9-22



[Asherda](#)

[v0.5.9-22](#)

[4ab9f03](#)

[v0.5.9-22](#) Pre-release

This release is intended for use in the VRSCTEST network. Mainnet use is still being tested.

Verus ID - A Better Blockchain Identity Technology

If you've been around crypto for even a little while, you'll start to hear things like "not your keys, not your coins", "lose your keys, lose your coins". If you lose your keys, no one will ever be able to recover your money.

In fact, trading off decentralized, censorship resistant public systems for this risk to your cryptocurrency assets is such a fundamental issue in blockchain systems today that it is considered an acceptable price for having the benefit of full control over your own funds and blockchain assets.

Verus ID provides an elegant solution to these previously unsolved, fundamental problems, and also provides quantum-ready friendly identity names and blockchain funds addresses as self-sovereign, revocable, recoverable identities that provide the following advantages over today's blockchain systems:

1. Identity and currency address are one and the same. Your friendly name can be used as both a reference to who you are and an on-chain destination for any funds transfer.
2. Each identity must have a set of addresses (1 or more) and a minimum number of signatures required from those addresses to spend or sign on behalf of an identity. Each identity also refers to two additional identities that have authority over:

- **Primary** - this is the self identity and may modify any part of the identity except its name, its parent, which is derived from the blockchain on which it is defined, and unless it is also one of the two other authorities, it may not modify any data under their control, including the identity of the other authorities once specified.
 - **Revocation** - this identity may revoke the primary identity, which will also revoke access to all funds and transactions under its control, but it may not spend any funds or sign on behalf of the identity.
 - **Recovery** - this authority may redefine/recover a revoked identity, but it may not change the revocation authority, nor may it modify an unrevoked identity at all.
3. Identities are transferable by defining them with primary addresses with private keys, which are under another party's control. When identities are transferred, the revocation and recovery authorities may or may not be modified as well, as long as the party updating the ID has the authority to do so.
 4. If an identity is transferred from one party to another, all funds in transactions that are under the control of that identity will also transfer to the party to which control is being transferred. In the native wallet, these changes are automatically reflected in the wallet balance and visibility of the balance of an identity using "z_getbalance identity@".
 5. If later, a new address type or signature algorithm, such as one that is quantum secure is added to the Verus blockchain, all transactions sent to an identity that is updated to be under the control of the new address type will be subject to its spend conditions as well. That means that if you have 1000 transactions sent to your identity name/address, an upgrade to a quantum secure address, when they are available, will retroactively secure all 1000 transactions under the control of that address.

Making a new identity

Arguably, the hardest part of the new Verus identity technology is just making your first identity, especially without a GUI. Once you have your own identity, a lot becomes easier, not the least of which is the ability to send money to a friendly name instead of a base58 blockchain address or hash. For example, once you have your identity, either of the following commands will work as well as just about any other CLI command that takes an address:

```
./verus -chain=VRSCTEST z_sendmany bob@ '[{"address":"alice@", "amount":100}]'
```

or

```
./verus -chain=VRSCTEST sendtoaddress alice@ 100
```

Making a Verus ID requires two steps, steps that are designed to prevent potentially selfish miners from replacing your purchase of an ID from the Verus blockchain with their own purchase of the same name instead. To prevent this name "front running", you must first commit to a name that you wish to have in your identity as its friendly name. Friendly names are unique to a blockchain, so if someone already has a specific name you must have, you will need to either be the first to purchase that name from the blockchain or acquire it from the controlling individual or organization.

Creating a name commitment registers your commitment, but keeps the name completely secret until you register it and reveal your commitment. When you register a name, you must pay 100 VRSC or 100 VRSCTEST, unless you have a referral from someone who already has an identity.

With a referral, the price is 80 VRSC or 80 VRSCTEST. When you register an identity with the CLI wallet, the correct amount of money, 100 VRSC without and 80 VRSC with a referral is automatically spent along with your registration. The funds for a referral are distributed as follows:

- 20 VRSC to the referrer's ID
- 20 VRSC to the referrer's referrer's ID
- 20 VRSC to the referrer's referrer's referrer's ID
- 20 VRSC to the miner of the block in which the ID is mined

100% of all funds that are not paid to referrers go to miners or stakers of the blockchain.

To create an identity:

Create your commitment with the following command:

```
./verus -chain=VRSCTEST registernamecommitment Name youraddressorid
referrerfriendlyname@
That will respond with something like:
{
  "txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",
  "namereservation": {
    "name": "Name",
    "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",
    "referral": "iSJLkp1hvn51Zg2Y6FBKjzLs9AAy7fomWN",
    "parent": "",
    "nameid": "iDXx9FPrAS5k2XCGss6FFmDQQMsts63uUg"
  }
}
```

You then wait for the commitment to be mined into a block, then use the information returned above as follows:

```
./verus -chain=VRSCTEST registeridentity
'{"txid":"377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",
"namereservation": {
  "name": "Name", "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",
  "referral": "referrerfriendlyname@", "identity": {"name": "Name",
  "primaryaddresses": ["RKo5u8N1sStZu81fU8kaxhoDcFcJmNEwSp"], "minimumsignatures": 1,
  "privateaddress": "zs14y0aa096em2as6n4fauyumqeywz45rfpze38c39fksgtjsac9vmmms7fmstc
  ylpalm7rseu8838"}}'
```

As long as you have enough funds and you are registering a new identity on the blockchain, you will be able to use your new identity in place of an address as soon as the transaction created by the registration command is mined into the blockchain.

You may list the identities in your wallet with the command `listidentities`, and you may display specific identities with the command `getidentity`. If you have the authority, or use identities under your control as revocation or recovery authorities for other identities, you may also revoke and recovery identities as well.

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING**

DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:

Linux:

~/.komodo/VRSCTEST
~/.verustest

MacOS

Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest

Windows

"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%\VerusTest

Launching the testnet:

./verusd -chain=VRSCTEST

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/23ae5a14f89a0806f1a6ac143c01a2a46bc3ddbe39136534444824e02b94329c/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/82bf9054feaf39e750a98542086bf5e5217ce18200bd8ff40e509bbb4db00ac3/detection>

Windows:

<https://www.virustotal.com/#/file/e18d521cf928025dc935a6b280909cd4aa06ed77fb60a0af5f43f92ddeb2805/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.5.9-2

 [Asherda](#)

[v0.5.9-2](#)

[ffff5015](#)

[v0.5.9-2](#)

IMPORTANT SECURITY/PRIVACY UPDATE

[Version 0.5.9](#) incorporated a fix for the Zcash protocol issue described in:

<https://z.cash/support/security/announcements/security-announcement-2019-09-24/>

Updating to v0.5.9 or more recent version is considered CRITICAL, not mandatory. We recommended that all users upgrade to this version of the Verus CLI or GUI wallet immediately and discontinue use of previous wallets.

Notable Changes

- Fixed address parsing in getrawtransaction

Testnet Reset

The testnet was deleted and relaunched on the 0.5.9 release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO 0.5.9, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

~/.komodo/VRSCTEST
~/.verustest

MacOS

Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest

Windows

"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%\VerusTest

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

With no costs beyond public blockchain network fees, Verus PBaaS Reserve technology enables any organization of any size to make a fractional reserve currency of its own on an independent, uncluttered blockchain with cross-chain interoperability, and immediately useful for payments and conversion to other currencies from within the Verus wallet. 100% of the 0.01% fees for such conversions go directly to the miners and stakers of the organization's new chain, and those miners and stakers also have the option of merge mining up to 14 other blockchains and coins while mining their own project, improving efficiency, profitability, and shared liquidity for everyone involved. Each blockchain is independent and connected through the Verus Reserve protocol

Launching the testnet:

```
./verus -chain=VRSCTEST
```

Chain definition:

- Maximum of 3 eras
 - Minimum 500 blocks worth of notarizations.
 - Minimum of 0.01 VRSCTEST per block of notarization.
- VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "premine": 50000000000000, "initialcontribution": 500000000000, "conversion": 10000000, "minpreconvert": 4500000000000, "maxpreconvert": 100000000000000, "launchfee": 0, "billingperiod": 14400, "notarizationreward": 100000000000, "eras": [{"reward": 5000000000, "decay": 0, "halving": 0, "eraend": 0, "eraoptions": 1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1

conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in `.verustest/PBAAS/CHAINNAME`, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between

the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN

CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/13dbc2ac7d64f5bf507187b343359c6afc082138bd37d27ae42180b1e5150e49/detection>

Linux:

<https://www.virustotal.com/#/file/d6e6b789457ba789bc98385bc444d16c2d8d56f1fe04beeb6fe2bdfadb36b664/detection>

Windows:

<https://www.virustotal.com/#/file/c0490311446822c8667a202cad64d90539a45ed84da8a90d7d616df4bbcfe995/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC. To find out more about the false p...

Assets 8

v0.5.9-1



[Asherda](#)
[v0.5.9-1](#)
[1afe8b7](#)
[v0.5.9-1](#)

IMPORTANT SECURITY/PRIVACY UPDATE

Version 0.5.9 incorporated a fix for the Zcash protocol issue described in:

<https://z.cash/support/security/announcements/security-announcement-2019-09-24/>

Updating to v0.5.9 or more recent version is considered CRITICAL, not mandatory. We recommended that all users upgrade to this version of the Verus CLI or GUI wallet immediately and discontinue use of previous wallets.

Notable Changes

- Avoid re-indexing whenever possible

Testnet Reset

The testnet was deleted and relaunched on the 0.5.9 release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO 0.5.9, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

```
~/.komodo/VRSCTEST
~/.verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%\VerusTest
```

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

With no costs beyond public blockchain network fees, Verus PBaaS Reserve technology enables any organization of any size to make a fractional reserve currency of its own on an independent, uncluttered blockchain with cross-chain interoperability, and immediately useful for payments and conversion to other currencies from within the Verus wallet. 100% of the 0.01% fees for such conversions go directly to the miners and stakers of the organization's new chain, and those miners and stakers also have the option of merge mining up to 14 other blockchains and coins while mining their own project, improving efficiency, profitability, and shared liquidity for everyone involved. Each blockchain is independent and connected through the Verus Reserve protocol

Launching the testnet:

```
./verus -chain=VRSCTEST
```

Chain definition:

- Maximum of 3 eras
 - Minimum 500 blocks worth of notarizations.
 - Minimum of 0.01 VRSCTEST per block of notarization.
- VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "premine": 5000000000000000, "initialcontribution": 50000000000000, "conversion": 100000000, "minpreconvert": 45000000000000, "maxpreconvert": 1000000000000000, "launchfee": 0, "billingperiod": 14400, "notarizationreward": 100000000000, "eras": [{"reward": 5000000000, "decay": 0, "halving": 0, "eraend": 0, "eraoptions": 1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount": 5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PbaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/341d9d06eed1c8df0057d56f853667e9738e8ec89cd6329a84c81543d38582fe/detection>

Linux:

<https://www.virustotal.com/#/file/eaf521b19ff4c6d8bfc84d25a59ae6ac745003f3927ed3fc71d5a6af883b32c7/detection>

Windows:

<https://www.virustotal.com/#/file/10e6bb5be7b7aa33bb57beb6105a5477e8f863c8f3d2b487f239ede9864dbbfc/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC. To find out more about the false positives...

Assets 8

v0.5.9

 [Asherda](#)

[v0.5.9](#)

[9e4bb76](#)

[v0.5.9](#)

IMPORTANT SECURITY/PRIVACY UPDATE

This update incorporates a fix for the Zcash protocol issue described in:

<https://z.cash/support/security/announcements/security-announcement-2019-09-24/>

In addition, this version also includes updates to Sapling functionality, including the Sprout to Sapling migration APIs added to the Zcash protocol up to and including version 2.0.7-3 of Zcash. The issue fixed by this update does not expose any funds to theft or counterfeit, but it does fix a vulnerability that an attacker may exploit to probe for information regarding a specific node's

control over private keys. Since the issue resolved does not put funds directly at risk, this update is considered CRITICAL, not mandatory. We recommended that all users upgrade to this version of the Verus CLI or GUI wallet immediately and discontinue use of previous wallets.

Notable Changes

- Integrated Zcash updates from Sapling release to 2.0.7-3
- Added Sprout to Sapling upgrade API
- Including support for PBaaS multi-chain test network with permissionless blockchain launches, reserve currency conversion protocols, merge mining, and easy cross-chain transactions.

Testnet Reset

The testnet will be deleted and relaunched for this release. **IF YOU HAVE PREVIOUSLY LAUNCHED VRSCTEST, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

~/.komodo/VRSCTEST
~/.verustest

MacOS

Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest

Windows

%APPDATA%\Komodo\VRSCTEST
%APPDATA%\VerusTest

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

With no costs beyond public blockchain network fees, Verus PBaaS Reserve technology enables any organization of any size to make a fractional reserve currency of its own on an independent, uncluttered blockchain with cross-chain interoperability, and immediately useful for payments and conversion to other currencies from within the Verus wallet. 100% of the 0.01% fees for such conversions go directly to the miners and stakers of the organization's new chain, and those miners and stakers also have the option of merge mining up to 14 other blockchains and coins while mining their own project, improving efficiency, profitability, and shared liquidity for everyone involved. Each blockchain is independent and connected through the Verus Reserve protocol

.

Launching the testnet:

```
./verus -chain=VRSCTEST
```

Chain definition:

- Maximum of 3 eras
- Minimum 500 blocks worth of notarizations.
- Minimum of 0.01 VRSCTEST per block of notarization.

VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name":"RESERVEWITHPREMINE","paymentaddress":"R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM","premine":50000000000000,"initialcontribution":5000000000000,"conversion":100000000,"minpreconvert":4500000000000,"maxpreconvert":100000000000000,"launchfee":0,"billingperiod":14400,"notarizationreward":1000000000000,"eras":[{"reward":5000000000,"decay":0,"halving":0,"eraend":0,"eraoptions":1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate  
startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE....

Assets 8

PaaS Cross-chain Technology Preview September 23, 2019



[v0.6.0.34-PaaS-Cross-chain-Technology-Preview](#)

[b2c0e54](#)

[PaaS Cross-chain Technology Preview September 23, 2019](#) Pre-release

Notable Changes

- Properly distribute initial contributions on non-reserve chains
- Show chain name on failure to notarize

- Continue attempting to notarize after running out of notary rewards
- Fix getdefinedchains currency state

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

With no costs beyond public blockchain network fees, Verus PBaaS Reserve technology enables any organization of any size to make a fractional reserve currency of its own on an independent, uncluttered blockchain with cross-chain interoperability, and immediately useful for payments and conversion to other currencies from within the Verus wallet. 100% of the 0.01% fees for such conversions go directly to the miners and stakers of the organization's new chain, and those miners and stakers also have the option of merge mining up to 14 other blockchains and coins while mining their own project, improving efficiency, profitability, and shared liquidity for everyone involved. Each blockchain is independent and connected through the Verus Reserve protocol

Launching the testnet:

```
./verus -chain=VRSCTEST
```

Chain definition:

- Maximum of 3 eras
 - Minimum 500 blocks worth of notarizations.
 - Minimum of 0.01 VRSCTEST per block of notarization.
- VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name":"RESERVEWITHPREMINE","paymentaddress":"R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM","premine":50000000000000,"initialcontribution":50000000000000,"conversion":100000000,"minpreconvert":45000000000000,"maxpreconvert":100000000000000,"launchfee":0,"billingperiod":14400,"notarizationreward":1000000000000,"eras":[{"reward":5000000000,"decay":0,"halving":0,"eraend":0,"eraoptions":1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}],
```

```
{"networkaddress":"2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMc  
y5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE",  
"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress":  
"R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE",  
"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress":  
"R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Assets 8

v0.5.8

 [Asherda](#)

[v0.5.8](#)

[1a1d05a](#)

[v0.5.8](#)

This release includes the ability to connect to and use the Verus Testnet with easy multi-chain network, cross-chain send, and in-wallet currency conversion capabilities that go beyond any blockchain system or cryptocurrency available today. Don't take our word for it. Just give it a try when you're not using the other mainstream wallet features

Try out cross-chain sends, create and exchange your own reserve currencies and merge mine up to 15 chains at once on the VerusCoin testnet!

Notable Changes

- Added Testnet support for Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

With no costs beyond public blockchain network fees, Verus PBaaS Reserve technology enables any organization of any size to make a fractional reserve currency of its own on an independent, uncluttered blockchain with cross-chain interoperability, and immediately useful for payments and conversion to other currencies from within the Verus wallet. 100% of the 0.01% fees for such conversions go directly to the miners and stakers of the organization's new chain, and those miners and stakers also have the option of merge mining up to 14 other blockchains and coins while mining their own project, improving efficiency, profitability, and shared liquidity for everyone involved. Each blockchain is independent and connected through the Verus Reserve protocol

.

Launching the testnet:

```
./verus -chain=VRSCTEST
```

Chain definition:

- Maximum of 3 eras
- Minimum 500 blocks worth of notarizations.
- Minimum of 0.01 VRSCTEST per block of notarization.

VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name":"RESERVEWITHPREMINE","paymentaddress":"R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM","premine":50000000000000,"initialcontribution":5000000000000,"conversion":100000000,"minpreconvert":4500000000000,"maxpreconvert":100000000000000,"launchfee":0,"billingperiod":14400,"notarizationreward":1000000000000,"eras":[{"reward":5000000000,"decay":0,"halving":0,"eraend":0,"eraoptions":1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate  
... a specific block.  
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate  
startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/462a2014bdd120eb155b964fe1f22910671ade38e26ff4372e64341e9613dff5/detection>

Linux:

<https://www.virustotal.com/#/file/6b1b3b848ab12c471d0469db37e5f8059aee565550d9e1c38fd97a55d493f184/detection>

Windows:

<https://www.virustotal.com/#/file/878ee7b0ed79d20f03994cf6322345be4a9f857aa4326ceb564d4ec51edd53ef/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC. To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview September 14 2019



[Asherda](#)

[v0.6.0.33-PBaaS-Cross-chain-Technology-Preview](#)

[ab04c6c](#)

[Public Blockchains as a Service \(PBaaS\) and Verus Reserve cross-chain currency conversion technology -- Technology Preview September 14 2019](#) Pre-release

Please note that this is a test release intended to use for testing and development with public blockchains as a service (PBaaS). THIS HAS NOT BEEN TESTED FOR USE ON MAINNET.

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

Verus PBaaS Reserve technology enables any organization of any size to make a currency with its own blockchain, support it in world class wallets, develop its own applications, and use it for payments worldwide that can be converted to the reserve currency without ever having to leave the blockchain network. Each blockchain is independent and connected through the Verus Reserve protocol.

Notable Changes

- Minor fix to export transactions

Chain definition:

- Maximum of 3 eras
- Minimum 500 blocks worth of notarizations.
- Minimum of 0.01 VRSCTEST per block of notarization.
VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "premine": 5000000000000000, "initialcontribution": 50000000000000, "conversion": 10000000000000, "minpreconvert": 45000000000000, "maxpreconvert": 1000000000000000, "launchfee": 0, "billingperiod": 14400, "notarizationreward": 1000000000000, "eras": [{"reward": 50000000000, "decay": 0, "halving": 0, "eraend": 0, "eraoptions": 1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount": 5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PbaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Assets 8

Public Blockchains as a Service (PBaaS) and Verus Reserve cross-chain currency conversion technology -- Technology Preview September 13 2019

 [Asherda](#)

[v0.6.0.29-PBaaS-Cross-chain-Technology-Preview](#)

[2accbf4](#)

[Public Blockchains as a Service \(PBaaS\) and Verus Reserve cross-chain currency conversion technology -- Technology Preview September 13 2019](#) Pre-release

Please note that this is a test release intended to use for testing and development with public blockchains as a service (PBaaS). THIS HAS NOT BEEN TESTED FOR USE ON MAINNET.

The Verus public testnet enables anyone to make new blockchains and currencies, each with scale and operational independence, zk-SNARK privacy, automatic wallet support, and 100% liquid conversion between VRSCTEST currency at any volume. Sending currencies cross-chain or converting between currencies is as easy as sending a cryptocurrency in less advanced systems today, except that the new Verus protocol enables you to include another blockchain as part of your destination and convert a fractional reserve currency to and from its reserve, according to its exact price along the way. All conversions are fully decentralized, require a standard protocol fee of 0.01%, which goes 100% to miners and stakers of the network, and calculates one price per block for all buys and sells in that block with zero spread.

Verus PBaaS Reserve technology enables any organization of any size to make a currency with its own blockchain, support it in world class wallets, develop its own applications, and use it for payments worldwide that can be converted to the reserve currency without ever having to leave the blockchain network. Each blockchain is independent and connected through the Verus Reserve protocol.

Notable Changes

- Fixed daemon output export error message on VRSCTEST
- Fixed fees bouncing back from a reserve send to Verus and allow exports to continue if done incorrectly
- Include currency state in chain definitions for both getchaindefinition and getdefinedchains

Chain definition:

- Maximum of 3 eras
- Minimum 500 blocks worth of notarizations.
- Minimum of 0.01 VRSCTEST per block of notarization.

VRSCTEST are needed in wallet to make a chain.

Sample chain definition:

```
./verusd -chain=VRSCTEST definechain
'{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "premine": 500000000000000, "initialcontribution": 500000000000, "conversion": 100000000, "minpreconvert": 45000000000000, "maxpreconvert": 100000000000000, "launchfee": 0, "billingperiod": 14400, "notarizationreward": 100000000000, "eras": [{"reward": 5000000000, "decay": 0, "halving": 0, "eraend": 0, "eraoptions": 1}], "nodes": [{"networkaddress": "ipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}, {"networkaddress": "2ndipaddress:port", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM"}]}'
```

The example above defines a chain that emits a premine to the indicated address and also converts all contributions up to a maximum amount to the native chain currency at launch at a 100%, 1:1 conversion price. All contributions will enter the currency's reserve deposits, and the more contributions on this chain, the higher the starting reserve ratio will be. If the minimum contributions are not met, the chain will fail to start and all contributions can be spent back to the original contributors.

The initial price in reserve at chain launch will start at 1.0, but there will be some fees to be paid in the early blocks that will come in as reserve currency. All fees in reserve that are converted to a fractional reserve currency are effective purchases of the fractional reserve currency, and based on a reversible formula, a currency's price will rise or fall on conversion from or to reserves with a level of volatility that is inversely related to the reserve ratio. A currency with a reserve ratio of 1 will never change its price when converting to or from its reserve. A currency with a reserve of 0.1 or 10% will rise or fall fairly quickly when converted to or from, depending on the amount of conversion relative to the total currency reserves on deposit.

Once a chain is made, anyone running the local VRSCTEST chain can connect to any defined chain with the command:

```
./verusd -chain=SOMECHAIN
```

On first connection, a local config file is created, and VRSCTEST will not need to run to connect. However, it will be needed to merge mine.

The presence of a correct config file for any chain, which are kept in .verustest/PBAAS/CHAINNAME, which is the home directory for a PBaaS chain, enables the chain to load as an independent chain.

Here are a few examples of cross-chain operations using the command line:

Participating in a blockchain launch and converting from reserve before the chain has started (refund address is optional and will default to paymentaddress):

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "preconvert": 1}'
```

Sending unconverted VRSCTEST to an address on the RESERVEWITHPREMINE chain for later conversion to/from the native coin.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending reserve to another chain and converting a coin from reserve to that chain's native currency at the current market price.

```
./verus -chain=VRSCTEST sendreserve '{"name": "RESERVEWITHPREMINE", "paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "refundaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Sending reserve currency to an address on the current PBaaS reserve chain with no cross-chain operation, Reserve currency on a PBaaS chain is equivalent to and convertible between the native VRSCTEST currency, but may also be sent to and used on PBaaS reserve chains. It will arrive as reserve currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000}'
```

Sending native currency to an address on a PBaaS reserve chain and converting it to reserve currency at the same time at the current market price.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "toreserve": 1}'
```

Sending reserve currency to an address on a PBaaS reserve chain and converting it to native currency at the same time at the current price. It will arrive as native currency.

```
./verus -chain=RESERVEWITHPREMINE sendreserve '{"paymentaddress": "R9LkTaMn1XpjeorE4RjM6pMcy5Nxkx9jPM", "amount":5000, "tonative": 1}'
```

Getting the current price in reserve and other currency statistics for the most recent block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate
```

... a specific block.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate blocknum
```

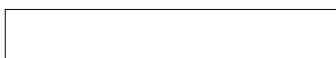
... a range with a specific period between.

```
./verus -chain=RESERVEWITHPREMINE getcurrencystate  
startblocknum, endblocknum, blockstostep
```

Refunding contributions made to a failed PaaS chain launch.

```
verus -chain=VRSCTEST refundfailedlaunch CHAINNAME
```

[Skip to content](#)



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

- -
 -
 - Code
 - Issues 21

- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v0.6.0-7

 [Asherda](#)
[v0.6.0-7](#)
[2491918](#)
[v0.6.0-7](#)

At block 800200, The Verus mainnet protocol was updated to include VerusID. This update is the latest for the upgraded Verus mainnet.

Notable Changes

- Improved staking check

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address

mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/387f4933627c7ad774c19ede52aba9d2ca65b27dca49e6dc6ee5bc471cbebc0/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/fc157e8826f9f547e8ff54eed6cccd1cf8c314104e90a46f39464a9484c9597fb/detection>

Windows:

<https://www.virustotal.com/#/file/ff10726405536584fa9be5c2989c8365aaa5eaa79bb5bd6cd603d9e2f979e8dc/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-6

 [Asherda](#)

[v0.6.0-6](#)

[a682ef2](#)

[v0.6.0-6](#)

At block 800200, The Verus mainnet protocol was updated to include VerusID. This update is the latest for the upgraded Verus mainnet.

This release fixes an issue that does not generally affect the Verus network, but can corrupt individual wallets in the case of a specific type of reorg occurrence that is relatively rare, but can happen. We STRONGLY RECOMMEND that all users upgrade to version 0.6.0-6 asap.

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs

and has met its original goals in keeping FPGA performance for the price under 2x of CPU. VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/c4415da8270a48944cd29507129085a677ea09ba3cf014f93393a9f6fe08391c/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/599b02ca94c9409f4167c15f7c5203b909f38c43693df702bb0dd92d7cdbe098/detection>

Windows:

<https://www.virustotal.com/#/file/75b8aee40a5394f6b99ab0ac815b8973cc475d83bedb6e24cb7e37996553bd7c/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

v0.6.0-5

 [Asherda](#)

[v0.6.0-5](#)

[fd371c1](#)

[v0.6.0-5](#)

THIS UPDATE FIXES AN OVERLY AGGRESSIVE BLOCK REJECTION IN THE VerusID PROTOCOL, AND IS A MANDATORY UPDATE FOR THE UPGRADED VERUS MAINNET

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

AT BLOCK 800200, THE VERUS MAINNET PROTOCOL WAS UPDATED TO INCLUDE VerusID. THIS UPDATE IS THE LATEST UPDATE FOR THE UPGRADED VERUS MAINNET

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery

state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.

- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU. VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/8d9a74da37c9b1be5f416c2f2f92f772b086da01770d8ed6a118778f87a7e568/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/27aa9d8d4a89dadf1a87928664eef7dc70ef92624d62596ee6e699b064f17daa/detection>

Windows:

<https://www.virustotal.com/#/file/e8f9b060d37e7028714d8c078705c8d6ceaf51b0d6f4c121cfbe40ab781b16f/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-4

 [Asherda](#)

[v0.6.0-4](#)

[f7cabaa](#)

[v0.6.0-4](#)

Notable Changes

- Alleviate peer banning behavior
- Staking enhancements

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

AT BLOCK 800200, THE VERUS MAINNET PROTOCOL WAS UPDATED TO INCLUDE VerusID. THIS UPDATE IS THE LATEST UPDATE FOR THE UPGRADED VERUS MAINNET

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of

UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/a3c9e986b441a4973c15a3578fb713578b294b19077feedb2182a78270560d52/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/3075c0ebc2337e36ce4d815a6a0d2a2715dcea2118cf7b70822061c60264c7e3/detection>

Windows:

<https://www.virustotal.com/#/file/594cf99ecc390e580edfbec4f7c73f45c959515030d256afb16e07074ba142e4/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-3

 [Asherda](#)

[v0.6.0-3](#)

[f0eb427](#)

[v0.6.0-3](#)

Notable Changes

- This update fixes an issue which would ban nodes too aggressively, resulting in disconnection from the chain and potential to fork away from the main chain. This should be considered a mandatory update.
- Add cansign and canspend to getidentity

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

**AT BLOCK 800200, THE VERUS MAINNET PROTOCOL HARDFORKED. THIS UPDATE IS MANDATORY TO REMAIN ON THE VERUS MAINNET.

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.

- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/a82b8cfb4207b6a8e05300bc91c0363dbad0de7e3f5a1a57043b641ba4852b2e/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/1246d439115b7578ddd47f8427a93f3ebad9a637d0735ed1db31003a9a3f53be/detection>

Windows:

<https://www.virustotal.com/#/file/2ca4b8bef8ad3e23bb896593be1509cf85619f2a6b87e72f9e59b99b52b63810/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-2



[Asherda](#)

[v0.6.0-2](#)

[a680284](#)

[v0.6.0-2](#)

Notable Changes

- Enable optional VRSC or VRSCTEST chains as parent of an ID. For example, bob@ is equivalent to bob.vrsc@ or, on testnet, bob.vrsctest@
- Update stratum protocol for mining to include solution information for miner, which enables smooth autoswitch and prepares for PBaaS merge mining
- Minor issues, including listidentities issue on Mac and compatibility with pool mining, even on block numbers under 16 to prepare for PBaaS

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

AT BLOCK 800200, THE VERUS MAINNET PROTOCOL WILL HARDFORK, AND THIS UPDATE IS MANDATORY TO REMAIN ON THE VERUS MAINNET. MAKE SURE TO UPGRADE BY DECEMBER 15TH, OR YOU MAY NEED TO RESYNCHRONIZE THE BLOCKCHAIN IF YOU ARE RUNNING IN NATIVE MODE

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which

will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/8a7f7bf3df09fa352979b1b92b3eea99fc802bce41f4349cfdeb3f25792233ce/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/de2961c3d4c0a00ce565989721d25b3abcf1e9e70b511c5cdb4e343005b13cc4/detection>

Windows:

<https://www.virustotal.com/#/file/74bbe68954498821f079a9a611cf6b8b8f8deb10f9a6e2feb594deba5c5ed7b1/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-1

 [Asherda](#)
[v0.6.0-1](#)
[fdc64a5](#)

v0.6.0-1

Thanks to [@hellcatz](#) and his constant development, improvement, and support of LuckPool.net for Verus and other coins they mine, we have fixed an issue that would have allowed the old hash algorithm to be used after the VerusID and VerusHash 2.1 network activation.

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm

AT BLOCK 800200, THE VERUS MAINNET PROTOCOL WILL HARDFORK, AND THIS UPDATE IS MANDATORY TO REMAIN ON THE VERUS MAINNET. MAKE SURE TO UPGRADE BY DECEMBER 15TH, OR YOU MAY NEED TO RESYNCHRONIZE THE BLOCKCHAIN IF YOU ARE RUNNING IN NATIVE MODE

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.

- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/021e59635bb77ccb8de45b017f5e07b1d71ab55e8f508e3a68aca96f7c9c2f5b/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/2c39e4fc74ba9a1874d5d3b9d957543c83ecca50f80f0be8f844b6a599a8b65c/detection>

Windows:

<https://www.virustotal.com/#/file/9f6a11ab3621496a0210d531167351a66c7b7881dfb405872b82db73c9094efe/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0 Verus ID - Decentralized, Friendly, Quantum-ready, Revocable, Recoverable Blockchain Identities for the World!



[Asherda](#)

[v0.6.0](#)

[be0d1c6](#)

[v0.6.0 Verus ID - Decentralized, Friendly, Quantum-ready, Revocable, Recoverable Blockchain Identities for the World!](#)

This release includes activation support for VerusID, and relative adjustment to the CPU and FPGA-equalizing VerusHash 2.1 Algorithm.

AT BLOCK 800200, THE VERUS MAINNET PROTOCOL WILL HARDFORK, AND THIS UPDATE IS MANDATORY TO REMAIN ON THE VERUS MAINNET. MAKE SURE TO UPGRADE BY DECEMBER 15TH, OR YOU MAY NEED TO RESYNCHRONIZE THE BLOCKCHAIN IF YOU ARE RUNNING IN NATIVE MODE

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it

made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction,

including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/bfa8df164e7ac5a1e7ea10e7abec275059df247fd9d62258e32b28777c40baf4/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/644b4e30c681f2268f18e9e258c696a8ab4ae53964bca4b0a2d256e407b464ec/detection>

Windows:

<https://www.virustotal.com/#/file/f9ad88116f33a1cc27ff75a4eb6108873a4ab3820759d0998b7a3a0e69d4d9b3/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.5.9-42 Verus Identity Release Candidate 3

 [Asherda](#)

[v0.5.9-42](#)

[8c1a6f6](#)

[v0.5.9-42 Verus Identity Release Candidate 3](#) Pre-release

Notable Changes

- Set content hashes will be a key/value map instead of an array
- Improve invalid block and invalid transaction handling.

Verus ID and VerusHash 2.1 Network Upgrade

This is the third release candidate (RC3) for the next major upgrade to the Verus network. It is considered fully ready for testing on both VRSCTEST testnet and VRSC mainnet. While this release is likely to work fine on mainnet through the coming activation, it is not yet an official release that you should expect to support mainnet activation. If you install this version, please make sure to upgrade to an official release before network activation, which will happen at block 800200, expected to be mined or staked on December 15th.

This release enables two technology upgrades, one that would be considered big news to most cryptocurrency projects, an improvement to the FPGA-equalizing hash algorithm, and a revolutionary, new decentralized identity technology, Verus IDs, that will disrupt today's centralized systems with the most secure, quantum ready, fully decentralized, self-sovereign digital identity system in the world.

Verus ID

Verus ID includes built-in privacy at the core through integration of zk-SNARKs, and provides a revolutionary improvement to blockchain address security, making it possible to actually recover your money after losing your keys, secure against identity and key theft, prove things about yourself without having to show more details than needed, and transfer assets to heirs as part of your estate. This release also introduces a new smart transaction technology that is unique to the Verus network and replaces its use of Komodo compatible crypto-conditions. The Verus ID system was made possible through the use of Verus Smart Transactions, which will be available for everyone to use on their own blockchains in the upcoming PBaaS network upgrade. Verus Smart Transactions use standard Bitcoin style serialization rather than the ASN.1 used in crypto conditions, which makes support on lite or mobile wallets simpler to implement than the crypto-condition protocol, which is still used on the Verus network for Stake Guard.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Verus ID - A Better Blockchain Identity Technology

If you've been around crypto for even a little while, you'll start to hear things like "not your keys, not your coins", "lose your keys, lose your coins". If you lose your keys, no one will ever be able to recover your money.

In fact, trading off decentralized, censorship resistant public systems for this risk to your cryptocurrency assets is such a fundamental issue in blockchain systems today that it is considered an acceptable price for having the benefit of full control over your own funds and blockchain assets.

Verus ID provides an elegant solution to these previously unsolved, fundamental problems, and also provides quantum-ready friendly identity names and blockchain funds addresses as self-sovereign, revocable, recoverable identities that provide the following advantages over today's blockchain systems:

1. Identity and currency address are one and the same. Your friendly name can be used as both a reference to who you are and an on-chain destination for any funds transfer.
2. Each identity must have a set of addresses (1 or more) and a minimum number of signatures required from those addresses to spend or sign on behalf of an identity. Each identity also refers to two additional identities that have authority over:
 - **Primary** - this is the self identity and may modify any part of the identity except its name, its parent, which is derived from the blockchain on which it is defined, and unless it is also one of the two other authorities, it may not modify any data under their control, including the identity of the other authorities once specified.
 - **Revocation** - this identity may revoke the primary identity, which will also revoke access to all funds and transactions under its control, but it may not spend any funds or sign on behalf of the identity.
 - **Recovery** - this authority may redefine/recover a revoked identity, but it may not change the revocation authority, nor may it modify an unrevoked identity at all.
3. Identities are transferable by defining them with primary addresses with private keys, which are under another party's control. When identities are transferred, the revocation and recovery authorities may or may not be modified as well, as long as the party updating the ID has the authority to do so.
4. If an identity is transferred from one party to another, all funds in transactions that are under the control of that identity will also transfer to the party to which control is being transferred. In the native wallet, these changes are automatically reflected in the wallet balance and visibility of the balance of an identity using `z_getbalance identity@"`.
5. If later, a new address type or signature algorithm, such as one that is quantum secure is added to the Verus blockchain, all transactions sent to an identity that is updated to be under the control of the new address type will be subject to its spend conditions as well. That means that if you have 1000 transactions sent to your identity name/address, an upgrade to a quantum secure address, when they are available, will retroactively secure all 1000 transactions under the control of that address.

Making a new identity

Arguably, the hardest part of the new Verus identity technology is just making your first identity, especially without a GUI. Once you have your own identity, a lot becomes easier, not the least of which is the ability to send money to a friendly name instead of a base58 blockchain address or hash. For example, once you have your identity, either of the following commands will work as well as just about any other CLI command that takes an address:

```
./verus -chain=VRSCTEST z_sendmany bob@ '[{"address":"alice@", "amount":100}]'
```

or

```
./verus -chain=VRSCTEST sendtoaddress alice@ 100
```

Making a Verus ID requires two steps, steps that are designed to prevent potentially selfish miners from replacing your purchase of an ID from the Verus blockchain with their own purchase of the same name instead. To prevent this name “front running”, you must first commit to a name that you wish to have in your identity as its friendly name. Friendly names are unique to a blockchain, so if someone already has a specific name you must have, you will need to either be the first to purchase that name from the blockchain or acquire it from the controlling individual or organization.

Creating a name commitment registers your commitment, but keeps the name completely secret until you register it and reveal your commitment. When you register a name, you must pay 100 VRSC or 100 VRSCTEST, unless you have a referral from someone who already has an identity. With a referral, the price is 80 VRSC or 80 VRSCTEST. When you register an identity with the CLI wallet, the correct amount of money, 100 VRSC without and 80 VRSC with a referral is automatically spent along with your registration. The funds for a referral are distributed as follows:

- 20 VRSC to the referrer’s ID
- 20 VRSC to the referrer’s referrer’s ID
- 20 VRSC to the referrer’s referrer’s referrer’s ID
- 20 VRSC to the miner of the block in which the ID is mined

100% of all funds that are not paid to referrers go to miners or stakers of the blockchain.

To create an identity:

Create your commitment with the following command:

```
./verus -chain=VRSCTEST registernamecommitment Name youraddressorid  
referrerfriendlyname@
```

That will respond with something like:

```
{  
  "txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
  "namereservation": {  
    "name": "Name",  
    "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
    "referral": "iSJLkp1hvn51Zg2Y6FBKjzLs9AAy7fomWN",  
    "parent": "",  
    "nameid": "iDXx9FPrAS5k2XCGss6FFmDQQMsts63uUg"  
  }  
}
```

You then wait for the commitment to be mined into a block, then use the information returned above as follows:

```
./verus -chain=VRSCTEST registeridentity
'{"txid":"377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",
"namereservation": {
    "name": "Name", "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",
    "referral": "referrerfriendlyname@"}, "identity": {"name": "Name",
"primaryaddresses": ["RKo5u8N1sStZu81fU8kaxhoDcFcJmNEwSp"], "minimumsignatures": 1,
"privateaddress": "zs14y0aa096em2as6n4fauyumqeywz45rfpze38c39fksgtjsac9vmms7fmstc
ylpaalm7rseu8838"}'}
```

As long as you have enough funds and you are registering a new identity on the blockchain, you will be able to use your new identity in place of an address as soon as the transaction created by the registration command is mined into the blockchain.

You may list the identities in your w...

Assets 8

v0.5.9-38 Verus Identity Release Candidate 2

 [Asherda](#)

[v0.5.9-38](#)

[f7326ce](#)

[v0.5.9-38 Verus Identity Release Candidate 2](#) Pre-release

Notable Changes

- Improved Identity synchronization
- Cleaned up debug log output
- Set block space limit for Identities

Verus ID and VerusHash 2.1 Network Upgrade

This is the second release candidate (RC2) for the next major upgrade to the Verus network. It is considered fully ready for testing on both VRSCTEST testnet and VRSC mainnet. While this release is likely to work fine on mainnet through the coming activation, it is not yet an official release that you should expect to support mainnet activation. If you install this version, please make sure to upgrade to an official release before network activation, which will happen at block 800200, expected to be mined or staked on December 15th.

This release enables two technology upgrades, one that would be considered big news to most cryptocurrency projects, an improvement to the FPGA-equalizing hash algorithm, and a revolutionary, new decentralized identity technology, Verus IDs, that will disrupt today's centralized systems with the most secure, quantum ready, fully decentralized, self-sovereign digital identity system in the world.

Verus ID

Verus ID includes built-in privacy at the core through integration of zk-SNARKs, and provides a revolutionary improvement to blockchain address security, making it possible to actually recover your money after losing your keys, secure against identity and key theft, prove things about yourself without having to show more details than needed, and transfer assets to heirs as part of your estate. This release also introduces a new smart transaction technology that is unique to the Verus network and replaces its use of Komodo compatible crypto-conditions. The Verus ID system was made possible through the use of Verus Smart Transactions, which will be available for everyone to use on their own blockchains in the upcoming PBaaS network upgrade. Verus Smart Transactions use standard Bitcoin style serialization rather than the ASN.1 used in crypto conditions, which makes support on lite or mobile wallets simpler to implement than the crypto-condition protocol, which is still used on the Verus network for Stake Guard.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Verus ID - A Better Blockchain Identity Technology

If you've been around crypto for even a little while, you'll start to hear things like "not your keys, not your coins", "lose your keys, lose your coins". If you lose your keys, no one will ever be able to recover your money.

In fact, trading off decentralized, censorship resistant public systems for this risk to your cryptocurrency assets is such a fundamental issue in blockchain systems today that it is considered an acceptable price for having the benefit of full control over your own funds and blockchain assets.

Verus ID provides an elegant solution to these previously unsolved, fundamental problems, and also provides quantum-ready friendly identity names and blockchain funds addresses as self-sovereign, revocable, recoverable identities that provide the following advantages over today's blockchain systems:

1. Identity and currency address are one and the same. Your friendly name can be used as both a reference to who you are and an on-chain destination for any funds transfer.

2. Each identity must have a set of addresses (1 or more) and a minimum number of signatures required from those addresses to spend or sign on behalf of an identity. Each identity also refers to two additional identities that have authority over:
 - **Primary** - this is the self identity and may modify any part of the identity except its name, its parent, which is derived from the blockchain on which it is defined, and unless it is also one of the two other authorities, it may not modify any data under their control, including the identity of the other authorities once specified.
 - **Revocation** - this identity may revoke the primary identity, which will also revoke access to all funds and transactions under its control, but it may not spend any funds or sign on behalf of the identity.
 - **Recovery** - this authority may redefine/recover a revoked identity, but it may not change the revocation authority, nor may it modify an unrevoked identity at all.
3. Identities are transferable by defining them with primary addresses with private keys, which are under another party's control. When identities are transferred, the revocation and recovery authorities may or may not be modified as well, as long as the party updating the ID has the authority to do so.
4. If an identity is transferred from one party to another, all funds in transactions that are under the control of that identity will also transfer to the party to which control is being transferred. In the native wallet, these changes are automatically reflected in the wallet balance and visibility of the balance of an identity using `z_getbalance identity@`.
5. If later, a new address type or signature algorithm, such as one that is quantum secure is added to the Verus blockchain, all transactions sent to an identity that is updated to be under the control of the new address type will be subject to its spend conditions as well. That means that if you have 1000 transactions sent to your identity name/address, an upgrade to a quantum secure address, when they are available, will retroactively secure all 1000 transactions under the control of that address.

Making a new identity

Arguably, the hardest part of the new Verus identity technology is just making your first identity, especially without a GUI. Once you have your own identity, a lot becomes easier, not the least of which is the ability to send money to a friendly name instead of a base58 blockchain address or hash. For example, once you have your identity, either of the following commands will work as well as just about any other CLI command that takes an address:

```
./verus -chain=VRSCTEST z_sendmany bob@ '[{"address":"alice@", "amount":100}]'
```

or

```
./verus -chain=VRSCTEST sendtoaddress alice@ 100
```

Making a Verus ID requires two steps, steps that are designed to prevent potentially selfish miners from replacing your purchase of an ID from the Verus blockchain with their own purchase of the same name instead. To prevent this name “front running”, you must first commit to a name that you wish to have in your identity as its friendly name. Friendly names are unique to a blockchain, so if someone already has a specific name you must have, you will need to either be the first to purchase that name from the blockchain or acquire it from the controlling individual or organization.

Creating a name commitment registers your commitment, but keeps the name completely secret until you register it and reveal your commitment. When you register a name, you must pay 100 VRSC or 100 VRSCTEST, unless you have a referral from someone who already has an identity. With a referral, the price is 80 VRSC or 80 VRSCTEST. When you register an identity with the CLI wallet, the correct amount of money, 100 VRSC without and 80 VRSC with a referral is automatically spent along with your registration. The funds for a referral are distributed as follows:

- 20 VRSC to the referrer's ID
- 20 VRSC to the referrer's referrer's ID
- 20 VRSC to the referrer's referrer's referrer's ID
- 20 VRSC to the miner of the block in which the ID is mined

100% of all funds that are not paid to referrers go to miners or stakers of the blockchain.

To create an identity:

Create your commitment with the following command:

```
./verus -chain=VRSCTEST registernamecommitment Name youraddressorid  
referrerfriendlyname@
```

That will respond with something like:

```
{  
  "txid": "377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
  "namereservation": {  
    "name": "Name",  
    "salt": "7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
    "referral": "iSJLKp1hvn51Zg2Y6FBKjzLs9AAy7fomWN",  
    "parent": "",  
    "nameid": "iDXx9FPrAS5k2XCGss6FFmDQQMsts63uUg"  
  }  
}
```

You then wait for the commitment to be mined into a block, then use the information returned above as follows:

```
./verus -chain=VRSCTEST registeridentity  
'{"txid":"377576f8ada1acc2aeb013ddf7a9ad86756cb990d4c5fc70b5a9a0fca43d727e",  
"namereservation": {  
  "name": "Name", "salt":  
"7b981f0a1ff2593167ef078150d0116335a7f329f9403b0abaeca30a42d8876a",  
"referral":"referrerfriendlyname@"}, "identity": {"name": "Name",  
"primaryaddresses": ["RKo5u8N1sStZu81fU8kaxhoDcFcJmNEwSp"],  
"minimumsignatures": 1,  
"privateaddress": "zs14y0aa096em2as6n4fauyumqeywz45rfpze38c39fksgtjsac9vmmms7fmstc  
ylpaalm7rseu8838"}'
```

As long as you have enough funds and you are registering a new identity on the blockchain, you will be able to use your new identity in place of an address as soon as the transaction created by the registration command is mined into the blockchain.

You may list the identities in your wallet with the c...

Assets 8

[Skip to content](#)

-
-

- Pricing



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

- [Code](#)
 - [Issues 21](#)
 - [Pull requests](#)
 - [Actions](#)
 - [Projects](#)
 - [Wiki](#)
 - [Security](#)
 - [Insights](#)

Releases: VerusCoin/VerusCoin

Releases Tags



v0.6.5-2

 [Asherda](#)

[v0.6.5-2](#)

[ea97d52](#)

[v0.6.5-2](#)

This is a mainnet release with fix to prevent any recurrence of the recent minor fork at block 1002417 and integrated support for the new PBaaS testnet. This release is approved for use on mainnet nodes, and it is being released on Github prior to full GUI wallet release v0.6.5-2 to enable exchanges, pools, and infrastructure nodes to update immediately, as needed.

Notable Changes:

- Added `getsaplingtree` API
- Added support for Verus Testnet with PBaaS Multi-currency Tokens
- Resolved multi-block reorg issue on mainnet that can prevent successful reorg in rare, otherwise valid cases

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

`~/.komodo/VRSCTEST`
`~/.verustest`

MacOS

`Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST`
`Users/{YourUserName}/Library/Application Support/VerusTest`

Windows

`"%APPDATA%\Komodo\VRSCTEST`
`"%APPDATA%\VerusTest`

Launching the testnet:

`./verusd -chain=VRSCTEST`

Verus PBaaS Multi-currency Tokens (TestNet only)

The testnet supports a completely new capability of token definitions and token launches (Kickstarter/Gofundme style, ICO, ITO, IPO, etc.). These tokens can be sent through the Verus protocol using the new 'sendcurrency' API, which works for all currencies in the Verus network. Tokens can be used as reserve currencies for other currencies as well. Token launches can accept multiple other coins or tokens for conversion to the new token with payment to the token ID. Each token can have a different conversion price for pre-launch participation as well as minimums and maximums of participation. If minimums are not met by the currency's startblock, all participation will be refunded, less a minimal transfer/network fee, which was already allocated to miners/stakers.

Tokens can be controlled by the blockchain and used for many purposes, including payment models, tickets, point systems, etc., or they can be partially blockchain controlled, centralized tokens, with currencies that are mintable at any time by the ID behind them. The controlling ID can also receive aggregated outgoing transactions to external systems that both burn coins and pass account data in the process. This will allow applications to control currency supplies on the public network, whether these applications are external, decentralized networks or centralized custodial systems, in many typical use cases of tokens, including club coins, game tokens, token launches, etc., no programming is required.

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions":
[0.01],"minpreconversion":[1000],"preallocation":[{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the

block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"mike@","convertto":"gold","preconvert":1,"amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,  
"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions":[0.01,  
1.0],"minpreconversion":[1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"  
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/9221506334ae6562d31f441202a5cbc019e6790ebec06c61ddaedebd9425db92/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/2e03e9442d34fe118cbeb586a5cff0567e98cdb4c6bda7cedf110f7456f7c62c/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/ac5d54de262512ff02405da2d6182fe4741d3929d8218a333024e2649e42123c/detection>

Windows:

<https://www.virustotal.com/#/file/f5223182d7dc042bf0fcfc383586d34327d3cd3c0955fa0db92812d41146f2298/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted text file.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation@"
AfeNDwABQR+hVp1D3jZymlkW4NrwdJbsh4jQDxxx44WNf6QZoXY/UiU1WFy9RM+/pkCC1IQfCXF6I9ey3B/3DofrvzKKAsV /Downloads/Verus-CLI-Linux-v0.6.5-2-amd64/Verus-CLI-Linux-v0.6.5-2-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation@"
AfeNDwABQR+hVp1D3jZymlkW4NrwdJbsh4jQDxxx44WNf6QZoXY/UiU1WFy9RM+/pkCC1IQfCXF6I9ey3B/3DofrvzKKAsV
99d7451472bc1fb34359d17e524d7755defbd279b0c2569a67231f8b7a37f9ff
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.5-1

 [Asherda](#)

[v0.6.5-1](#)

[c3c4e12](#)

[v0.6.5-1](#) Pre-release

This release is intended for use in the VRSCTEST only. Mainnet approved testnet release is not yet available

Notable Changes:

- Check for creation of invalid staking transactions

Verus Testnet PBaaS Multi-currency Tokens

The testnet supports a completely new capability of token definitions and token launches (Kickstarter/Gofundme style, ICO, ITO, IPO, etc.). These tokens can be sent through the Verus protocol using the new 'sendcurrency' API, which works for all currencies in the Verus network. Tokens can be used as reserve currencies for other currencies as well. Token launches can accept multiple other coins or tokens for conversion to the new token with payment to the token ID. Each token can have a different conversion price for pre-launch participation as well as minimums and maximums of participation. If minimums are not met by the currency's startblock, all participation will be refunded, less a minimal transfer/network fee, which was already allocated to miners/stakers.

Tokens can be controlled by the blockchain and used for many purposes, including payment models, tickets, point systems, etc., or they can be partially blockchain controlled, centralized tokens, with currencies that are mintable at any time by the ID behind them. The controlling ID can also receive aggregated outgoing transactions to external systems that both burn coins and pass account data in the process. This will allow applications to control currency supplies on the public network, whether these applications are external, decentralized networks or centralized custodial systems, in many typical use cases of tokens, including club coins, game tokens, token launches, etc., no programming is required.

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions": [0.01],"minpreconversion": [1000],"preallocation": [{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn

the 0.025% conversion fees of participation in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin", "options":96,  
"proofprotocol":2, "currencies":["vrsctest", "gold"], "conversions": [0.01,  
1.0], "minpreconversion": [1000, 1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"  
'[{"address":"mike@", "currency":"mycoin", "mintnew":1, "amount":10000}]'
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/64f4f431c73cc8d28da48ee5adaa8fddb07ca20339706c1aa953abe8819cbbd4/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/bfa5cdbbf35495f9e6e4c8a71da7a05c41795c844aecf0d05817f7e5ee32f2aa/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/3c71f41cf03ea85fb3d7c46b6fae18b9458bdd66d06f9dd16eea3a4dc0b3e05/detection>

Windows:

<https://www.virustotal.com/#/file/c671dff603e3a0219c9bd8403b8ac1efddb0485621dfce9b802e975b4d7652cc>

/detection

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Testing@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted textfile.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Testing@"
AcBbDwABQSAhp083zww8L/K3dZTImWLcF8D/u93GyeQFCtKRoiZbb10cvNCMVDN/13S74mZUGCekQQV9
t4fzChTy1mJ/DKKk /Downloads/Verus-CLI-Linux-v0.6.5-amd64/Verus-CLI-Linux-v0.6.5-
amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Testing@"
AcBbDwABQSAhp083zww8L/K3dZTImWLcF8D/u93GyeQFCtKRoiZbb10cvNCMVDN/13S74mZUGCekQQV9
t4fzChTy1mJ/DKKk
b589a80b79a022f24524e113ccd5b7841d137fb198c5e64618c7a3ed867feb62
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.5 Verus Testnet PBaaS Multi-currency Tokens

 [Asherda](#)

[v0.6.5](#)

[980d305](#)

[v0.6.5 Verus Testnet PBaaS Multi-currency Tokens](#) Pre-release

This release is intended for use in the VRSCTEST only. Mainnet approved testnet release is not yet available

Verus Testnet PBaaS Multi-currency Tokens

The testnet supports a completely new capability of token definitions and token launches (Kickstarter/Gofundme style, ICO, ITO, IPO, etc.). These tokens can be sent through the Verus protocol using the new 'sendcurrency' API, which works for all currencies in the Verus network. Tokens can be used as reserve currencies for other currencies as well. Token launches can accept multiple other coins or tokens for conversion to the new token with payment to the token ID. Each token can have a different conversion price for pre-launch participation as well as minimums and maximums of participation. If minimums are not met by the currency's startblock, all participation will be refunded, less a minimal transfer/network fee, which was already allocated to miners/stakers.

Tokens can be controlled by the blockchain and used for many purposes, including payment models, tickets, point systems, etc., or they can be partially blockchain controlled, centralized tokens, with currencies that are mintable at any time by the ID behind them. The controlling ID can also receive aggregated outgoing transactions to external systems that both burn coins and pass account data in the process. This will allow applications to control currency supplies on the public network, whether these applications are external, decentralized networks or centralized custodial systems, in many typical use cases of tokens, including club coins, game tokens, token launches, etc., no programming is required.

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions": [0.01],"minpreconversion": [1000],"preallocation": [{"mike@":50000000.0000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin", "options":96,  
"proofprotocol":2, "currencies":["vrsctest", "gold"], "conversions": [0.01,  
1.0], "minpreconversion": [1000, 1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"  
'[{"address":"mike@", "currency":"mycoin", "mintnew":1, "amount":10000}]'
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/f0cf8bcd39b69e23481d265fb8789d27ae6e655ee168acb06573268ded74185/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/eb259e23a629cac31789621cf4e917905ac43ba722e47f82f8dc0360d4f6ba72/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/ea0b8244bb8c6ddea4eb0eac51e50aa9dfd57ebb0c17696f44c3e21b3d99320c/detection>

Windows:

<https://www.virustotal.com/#/file/80bc1d1c574a91708cbb4f13be9ba110cccc0c4fd8555bc4378737660eef6d06>

/detection

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted textfile.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AT9MDwABQR+giVSMbRiq3p60hCUVfEdm/yPLZYjX1Eh+ufvtntpDPhyjvjUGQRClss+mQohhXPxfWY5oq
IzfioqJsnHHlo2XH /Downloads/Verus-CLI-Linux-v0.6.5-amd64/Verus-CLI-Linux-v0.6.5-
amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AT9MDwABQR+giVSMbRiq3p60hCUVfEdm/yPLZYjX1Eh+ufvtntpDPhyjvjUGQRClss+mQohhXPxfWY5oq
IzfioqJsnHHlo2XH
60020bfebbb2ced560c5d1e1333c43d9f80c4db4797812213ae04ad39df99c22
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.4-3

 [Asherda](#)
[v0.6.4-3](#)

[391c403](#)

[v0.6.4-3](#)

New in v0.6.4-3

- Added `z_viewtransaction` api: Gets detailed shielded information about an in-wallet transaction

Verus CLI v0.6.4-3

Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

Verus ID

Verus IDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. Verus IDs provide plenty of opportunity for identity applications. Specifically, Verus ID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

Verus IDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which

will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/c93d7f3ee35502a2dd9eaf87d76b253c6ee8e1b62c256103a267082daf3fa7f9/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/4346c377e8a6d95aae7abb289703a32ee1288786586a01fd0da6595c34d33ac9/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/131ec86c7a4024648d33c80992cfbd1708743fd4d1b9087234df6410d34ecb91/detection>

Windows:

<https://www.virustotal.com/#/file/6d57ec0805dc0f56604be718c83e61fe57fea9e640fcc3fb28e2bc8784619864/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted textfile.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
ATPPDgABQR/xBtY9wU+RmBJqX5Y6AwP1w9nJShSAFpyGzu50ufFR3pTC1s1Wyk96zGTCZqMUzaurrbt
u4ysrKZNr7FUq/0/ /Downloads/Verus-CLI-Linux-v0.6.4-3-amd64/Verus-CLI-Linux-
v0.6.4-3-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
ATPPDgABQR/xBtY9wU+RmBJqX5Y6AwP1w9nJShSAFpyGzu50ufFR3pTC1s1Wyk96zGTCZqMUzaurrbt
u4ysrKZNr7FUq/0/
3e5cc7bea46c6869545a0a886d66fdbf21fb2800761e3dca6f626046681ee26e
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.4-1

 [Asherda](#)
[v0.6.4-1](#)
[ad586ce](#)
[v0.6.4-1](#)

New in v0.6.4-1

- Added convertpassphrase api, allowing wif key exports of electrum wallet passphrases

Verus CLI v0.6.4-1

Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most

notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

Verus ID

Verus IDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. Verus IDs provide plenty of opportunity for identity applications. Specifically, Verus ID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

Verus IDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once

mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.

- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS

BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/21dff79a0060325787c96b4aaaf9becd161fe958cb4428adbc82d9bd1edb07cc/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/c4d339cc8a85b3beb240b3ea4692a7630124441305cf07cf7890724277df67f7/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/f8e92c5acf1fe5e0f317d4d0bbc8288c03b2ce4c21791a33ff5f2de60c385ca8/detection>

Windows:

<https://www.virustotal.com/#/file/d4c883ff9aa87e4e0356e66d44952a50e7bbff6fd8120a93b799510e20644a5a/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AcFjDQABQSDvcPZH+EBJHnQR491hRjINjwGtaVf3S8wFwNAY0KwYF1Qj3z0kndcLIJKqDCVKWj5RarHj
CApwUGbjYK4x29Uc /Downloads/Verus-CLI-Linux-v0.6.4-amd64/Verus-CLI-Linux-v0.6.4-
amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AcFjDQABQSDvcPZH+EBJHnQR491hRjINjwGtaVf3S8wFwNAY0KwYF1Qj3z0kndcLIJKqDCVKWj5RarHj
CApwUGbjYK4x29Uc
0cbd162064918fb63a8914c0cf963e0eb310fcdaa7433a9066c6806491e1
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.4

 [Asherda](#)

[v0.6.4](#)

[31f26c9](#)

[v0.6.4](#)

New in v0.6.4

- Estimated active staking supply displayed in output from `getmininginfo`
- Average block fees for last 100 blocks displayed in `getmininginfo` output
- Immature coins, including time locked coins displayed in `coinsupply` output
- Eligible staking balance, with consideration for all disqualifying factors displayed in `getwalletinfo` output
- `z_exportwallet` as wallet backup now accepts boolean parameter to remove all addresses with no UTXOs or Identities in the backup
- Staking performance is improved
- Fixed rare condition of improper ID wallet synchronization of transferred or revoked IDs

Verus CLI v0.6.4

Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed

already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

Verus ID

Verus IDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. Verus IDs provide plenty of opportunity for identity applications. Specifically, Verus ID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

Verus IDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/c90815f194641646e069824711fca2dc1f7a9cd6ba5bd5016bdcdbff8a1f190/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/5ea61dee317688aee255f6f5255d62d0307517196098ace354881a49ee36a3c3/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/3e9e463afed0eaa737a6fbd51438d5e828fccd4ca3d4be74751e79fe3eecc119/detection>

Windows:

<https://www.virustotal.com/#/file/a3b8980fa5044177976e53e30bb3142c9ed78d982fede9d4cf015417dfa35f8a/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation@".

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AYpdDQABQR9QSLDca3cCXdpZhNtUyVL3GdFM4248ka0Nhpp0d+/yAGmH7pmBXfcy0bnInU5cxGjgbe5n
2wLZRKmllS/zPM6q /Downloads/Verus-CLI-Linux-v0.6.4-amd64/Verus-CLI-Linux-v0.6.4-
amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AYpdDQABQR9QSLDca3cCXdpZhNtUyVL3GdFM4248ka0Nhpp0d+/yAGmH7pmBXfcy0bnInU5cxGjgbe5n
2wLZRKmllS/zPM6q
60df4018e329cc6748379369d02943b1ff4e7b21363beedb2624ea2770fba1b9
```

The result will be true if the signature is valid.

true

Assets 6

v0.6.2-1

 [Asherda](#)
[v0.6.2-1](#)
[beb316e](#)
[v0.6.2-1](#)

Notable Changes

- Improved coinsupply API speed and added accurate immature funds count

Verus CLI v0.6.2-1

Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

Verus ID

Verus IDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. Verus IDs provide plenty of opportunity for identity applications. Specifically, Verus ID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

Verus IDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs

are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/4cf0e9684029faaa29b87277cbe19f85b40a059d6a04ae9861d94beb6d95c56a/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/6bb4ebb2c4d5e0b59a52fb060d4b06596843955b3814dbf1c858a3e5d6be2ea9/detection>

Windows:

<https://www.virustotal.com/#/file/bd2fd47bb22dbe81bb34d74c04182d03e7e72f83a0cb254acb0b869d8ee07091/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation@".

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation@"
AZPmDAABQR/rW5EvSr1wxFw4FP+FBgTbNoZJKR0preBeH8dGQx4YJFcjz40LDSBqWfuffFFEFmaZGWNNW
Y3QVIp3jwU7qyLfu /Downloads/Verus-CLI-Linux-v0.6.2-1-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation@"
AZPmDAABQR/rW5EvSr1wxFw4FP+FBgTbNoZJKR0preBeH8dGQx4YJFcjz40LDSBqWfufFFEFmaZGWN
Y3QVIp3jwU7qyLfu
6bb4ebb2c4d5e0b59a52fb060d4b06596843955b3814dbf1c858a3e5d6be2ea9
```

The result will be true if the signature is valid.

true

Assets 8

v0.6.2

 [Asherda](#)
[v0.6.2](#)
[68b2bea](#)
[v0.6.2](#)

Verus CLI v0.6.2

Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!

Verus ID

Verus IDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. Verus IDs provide plenty of opportunity for identity applications. Specifically, Verus ID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

Verus IDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/32742e2b62aa446f6772ae68706626a953652c1766ac118cfb612ac6707bc48d/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/0eb57c65fc05daa34a55863f12987d4127cd8dd8c8fdce36df206e900598c9fe/detection>

Windows:

<https://www.virustotal.com/#/file/9aab61dc575c25e71df5c1c6d8c1eeb56d0b818e6a5d18378a412a7e929c810a/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation@".

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation@"
AfLZDAABQR9H2eDMg7vKCrf7KmJt/4+8Vi4FMJSrer93IUNCMhk63JHnCXrIv1GysI3nhkY3qoCXGpW
QiIBlk6pzA/+Ztg8 /Downloads/Verus-CLI-Linux-v0.6.2-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation@"
AfLZDAABQR9H2eDMg7vKCrf7KmJt/4+8Vi4FMJSrer93IUNCMhk63JHnCXrIv1GysI3nhkY3qoCXGpW
QiIBlk6pzA/+Ztg8
0eb57c65fc05daa34a55863f12987d4127cd8dd8c8fdce36df206e900598c9fe
```

The result will be true if the signature is valid.

true

Assets 8

v0.6.1



[v0.6.1](#)
[d0b3c87](#)
[v0.6.1](#)

Notable Changes

- z_importviewingkey and z_exportviewingkey for Sapling to enable confidential, verifiable, and transparent voting

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU.

VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which

will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/e1884e04a3d5932b99e3bfcb8a34cd3ed4034340367a5fade0b831eda6a0daa2/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/b514dbcf8ae804efe16ea5080751e3c0e6702a6db59b38eb7994c9a5fdc81c4c/detection>

Windows:

<https://www.virustotal.com/#/file/b5fb64cad0bb29ab76b0215b7050beee2919bcb1b6eef6bd52436d665fa3a051/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

v0.6.0-8

 [Asherda](#)
[v0.6.0-8](#)
[5dc785a](#)

v0.6.0-8

At block 800200, The Verus mainnet protocol was updated to include VerusID. This update is the latest for the upgraded Verus mainnet.

Notable Changes

- Fix accretion of coinbases in mempool

Verus ID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications. Specifically, VerusID provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. VerusIDs are:

- **Revocable** -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.
- **Recoverable** -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.
- **Private** - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to

selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU. VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/6dc2eb44ee294829c6f14e38349cccd1f8ba2bf683b93c0cda328c9bbdd588bab/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/ba3014f6ae1298fea11258474eb053b8841b1be04db911e4a3e2f80990ac42c3/detection>

Windows:

<https://www.virustotal.com/#/file/596ef34389d9aa801936da73f39f7ac6c85878f9b927321537e850deae4f614f/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Assets 8

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
 - [Privacy](#)
 - [Security](#)
 - [Status](#)
 - [Docs](#)
 - Contact GitHub
 - [Pricing](#)
 - [API](#)
 - [Training](#)
 - [Blog](#)
 - [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)

- • • • • • • • • •

-
-
-
- [Pricing](#)

[Sign in](#)

[Sign up](#)

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

-
-
-
-
- [Code](#)
- [Issues 21](#)
- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)

v0.7.1-5

 [Asherda](#)
[v0.7.1-5](#)
[8e50ee2](#)
[v0.7.1-5](#)

New in v0.7.1-5

This non-mandatory release is compatible with mainnet with minimal changes from v0.7.1-1. It contains fixes necessary to continue participating in the testnet launched in v0.7.1-1, without reset of your wallet or holdings, if you started using the VRSCTEST multi-currency testnet in v0.7.1-1 or later. You should be able to synchronize easily, but if you are unable to synchronize VRSCTEST shortly after connecting, try deleting the data in your VRSCTEST folder, and retain the wallet.dat file.

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1-5 has no improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencynname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencynname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

Enhancements to TestNet Protocols for Increased Scale, Currency Burn Functions, and Centralized Fractional Currencies

Version v0.7.1-5 is not yet considered hardened for mainnet, but has increased scale over previous versions in addition to the ability to apply both fractional and centralized attributes to a currency. This allows price neutral minting by the currency controller through automatic reductions of the reserve ratio of all currencies. This version also enables a new parameter to the "sendcurrency" API called "burn". If "burn" is set to 1, a currency sent, which must be a token or fractional currency is officially removed from that currency's supply. In the case of a fractional currency, this will have the effect of increasing the on-chain price for all holders, depending on the amount burned.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts, pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
' [{"currency":"vrsctest", "address":"alice@", "amount":10}] '
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency """  
' [{"currency":"vrsctest", "address":"alice@", "amount":10},  
 {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency """ ' [{"address":"bob@", "amount":10,  
 "convertto":"TESTCOIN"}]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency """ ' [{"address":"alice@", "amount":10,  
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency  
' {"name":"gold", "options":96, "currencies":["vrsctest"], "conversions":  
[0.01], "minpreconversion":1000, "preallocations":  
[{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency """  
' [{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions":[0.01,1.0],"minpreconversion":[1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

```
~/komodo/VRSCTEST
~/verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%"\\VerusTest
```

Launching the testnet:

```
./verusd -chain=VRSCTEST
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do...

Assets 6

v0.7.1-4

 [Asherda](#)
[v0.7.1-4](#)
[67d1355](#)
[v0.7.1-4](#)

New in v0.7.1-4

This non-mandatory release is compatible with mainnet with minimal changes from v0.7.1-1. It contains fixes necessary to continue participating in the testnet launched in v0.7.1-1, without reset of your wallet or holdings, if you started using the VRSCTEST multi-currency testnet in v0.7.1-1 or later. You should be able to synchronize easily, but if you are unable to synchronize VRSCTEST shortly after connecting, try deleting the data in your VRSCTEST folder, and retain the wallet.dat file.

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1-4 has no improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencynname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencynname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

Enhancements to TestNet Protocols for Increased Scale, Currency Burn Functions, and Centralized Fractional Currencies

Version v0.7.1-4 is not yet considered hardened for mainnet, but has increased scale over previous versions in addition to the ability to apply both fractional and centralized attributes to a currency. This allows price neutral minting by the currency controller through automatic reductions of the reserve ratio of all currencies. This version also enables a new parameter to the "sendcurrency" API called "burn". If "burn" is set to 1, a currency sent, which must be a token or fractional currency is officially removed from that currency's supply. In the case of a fractional currency, this will have the effect of increasing the on-chain price for all holders, depending on the amount burned.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts, pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
' [{"currency":"vrsctest", "address":"alice@", "amount":10}] '
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency":"vrsctest", "address":"alice@", "amount":10}, {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"TESTCOIN"}]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold", "options":96, "currencies":["vrsctest"], "conversions": [0.01], "minpreconversion": [1000], "preallocation": [{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*" '[{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions":[0.01,1.0],"minpreconversion":[1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

```
~/komodo/VRSCTEST
~/verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%"\\VerusTest
```

Launching the testnet:

```
./verusd -chain=VRSCTEST
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do ...

Assets 6

v0.7.1-3

 [Asherda](#)
[v0.7.1-3](#)
[7e0042a](#)
[v0.7.1-3](#)

New in v0.7.1-3

- This non-mandatory release is compatible with mainnet with minimal changes from v0.7.1-1. It contains fixes necessary to continue participating in the testnet launched in v0.7.1-1 and 0.7.1-2, without reset of your wallet or holdings, if you started using the VRSCTEST multi-currency testnet in v0.7.1-1 or "-2". You should be able to synchronize easily, but if you are unable to synchronize VRSCTEST shortly after connecting, try deleting the data in your VRSCTEST folder, and retain the wallet.dat file.
- Improved staking supply calculation

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1-3 has minor improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencynname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencynname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

Enhancements to TestNet Protocols for Increased Scale, Currency Burn Functions, and Centralized Fractional Currencies

Version v0.7.1-3 is not yet considered hardened for mainnet, but has increased scale over previous versions in addition to the ability to apply both fractional and centralized attributes to a currency. This allows price neutral minting by the currency controller through automatic reductions of the reserve ratio of all currencies. This version also enables a new parameter to the "sendcurrency" API called "burn". If "burn" is set to 1, a currency sent, which must be a token or fractional currency is officially removed from that currency's supply. In the case of a fractional currency, this will have the effect of increasing the on-chain price for all holders, depending on the amount burned.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are

simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts, pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest","address":"alice@","amount":10}]'
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency":"vrsctest", "address":"alice@", "amount":10}, {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"TESTCOIN"}]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@, that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency '{"name":"gold", "options":96, "currencies":[{"vrsctest"}, {"conversions": [0.01], "minpreconversion": [1000], "preallocation": [{"mike@":50000000.0000000}]}]}
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*" '[{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions":[0.01,1.0],"minpreconversion":[1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:**

Linux:

```
~/komodo/VRSCTEST
~/verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%"\\VerusTest
```

Launching the testnet:

```
./verusd -chain=VRSCTEST
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to p...

Assets 6

v0.7.1-2

 [Asherda](#)
[v0.7.1-2](#)
[176ee5c](#)
[v0.7.1-2](#)

New in v0.7.1-2

- This non-mandatory release is compatible with mainnet with no changes from v0.7.1-1. It contains fixes necessary to continue participating in the testnet launched in v0.7.1-1, without reset of your wallet or holdings, if you started using the VRSCTEST multi-currency testnet in v0.7.1-1. You should be able to synchronize easily, but if you are unable to synchronize VRSCTEST shortly after connecting, try deleting the data in your VRSCTEST folder, and retain the wallet.dat file

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1-2 has minor improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencynname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencynname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts,

pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
' [{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency "*"
' [{"currency":"vrsctest", "address":"alice@", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency "*" '[ {"address":"bob@", "amount":10,
"convertto":"TESTCOIN"} ]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency "*" '[ {"address":"alice@", "amount":10,
"convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"} ]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions":
[0.01],"minpreconversion":[1000],"preallocation": [{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"mike@", "convertto":"gold", "preconvert":1, "amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,
"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions": [0.01,
1.0],"minpreconversion": [1000, 1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@", "currency":"mycoin", "mintnew":1, "amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING**

DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:

Linux:

```
~/.komodo/VRSCTEST  
~/.verustest
```

MacOS

```
Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST  
Users/{YourUserName}/Library/Application Support/VerusTest
```

Windows

```
"%APPDATA%\Komodo\VRSCTEST  
"%APPDATA%\VerusTest
```

Launching the testnet:

```
./verusd -chain=VRSCTEST
```

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/1f9cb1fe4ba5aef4a0859aebd2b89001c72e1482fd27a99e1ee67f8de11011a7/detection>

Linux-AMD64: <https://www.virustotal.com/#/file/ccc03d8a038bf21af8...>

Assets 6

v0.7.1-1

 [Asherda](#)

[v0.7.1-1](#)
[535988a](#)
[v0.7.1-1](#)

New in v0.7.1-1

- Re-launched VRSCTEST with fixes. Make sure you delete your VRSCTEST data before connecting to the testnet.

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1-1 has minor improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencyname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencyname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts,

pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
' [{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency "*"
' [{"currency":"vrsctest", "address":"alice@", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency "*" '[ {"address":"bob@", "amount":10,
"convertto":"TESTCOIN"} ]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency "*" '[ {"address":"alice@", "amount":10,
"convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"} ]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions":
[0.01],"minpreconversion":[1000],"preallocation": [{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"mike@","convertto":"gold","preconvert":1,"amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,
"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions": [0.01,
1.0],"minpreconversion": [1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING**

DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:

Linux:

~/.komodo/VRSCTEST
~/.verustest

MacOS

Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest

Windows

"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%\VerusTest

Launching the testnet:

./verusd -chain=VRSCTEST

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/e94319a3ad2b81825b0ce231deb1117ac28ae897ecee1dfb0fe424c198db7d06/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/a9eb0f6a0bec1feb8e156410e0ccdbb7139e16a4b10bdea5e939b30ad834f50f/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/f17a5af4d09123d3f11f3682929fdc92f46f474463134fbdc1c2da4e56bdf8f6/detection>

Windows:

<https://www.virustotal.com/#/file/aafa51f1d6068d65a80715db668e7e7e39b9d9437005de957d08f7300094a20c/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or ...

Assets 6

v0.7.1



[Asherda](#)

[v0.7.1](#)

[c35f5a7](#)

[v0.7.1](#)

Verus Testnet with Multi-currency Fractional Reserves and Currency Launch Protocol

Verus multi-currency fractional reserve baskets are UTXO-based, fractional reserve currencies with anti-front running, simultaneously solved, provably fair liquidity system for all currencies and tokens in or imported into the PBaaS system.

While Verus daemon v0.7.1 has minor improvements for mainnet operation, it offers full support for the new Verus Testnet multi-currency technology preview. On testnet, the new "sendcurrency" API, which is already available on Verus mainnet in its simplified form, is enhanced with new multi-currency parameters. The first parameter, "currency":"currencyname" - where the default on testnet for currency name is "vrsctest", specifies the source currency of the send. The next, "convertto":"currencyname", allows conversion from one currency to either a fractional currency where it is a reserve or a reserve of the fractional currency of the source.

An end to front running

Conversion will always be at the same rate as all conversions processed in one group of transactions. There is also no spread between the conversion to and from a currency. All transactions in a aggregation of import blocks, which will be detailed in later documentation, are simultaneously solved in all currency conversion directions and all converted at the same rate, all getting the same price in each direction as any person who may be exchanging in the other.

There is a fee, 0.025% for conversions, 100% of which goes to miners and stakers. While it is not yet implemented, we also intend to add an implied volatility fee, which will be charged variably on imbalanced blocks of transactions, making blocks that change the price of a currency significantly pay more of an implied volatility fee, and ensuring that relatively volatile currency baskets offset any potential impermanent loss to liquidity providers risk with earnings from implied volatility fees. At the same time, currency baskets that are less volatile relative to the constituent currencies will typically have lower implied volatility fees and lower earnings for liquidity providers to offset a lower risk of impermanent loss.

The Best Way to Launch a Currency

Currency launches with Kickstarter-like minimums to activate or, if not met, automatically refund, dynamic currency launch pricing, based on participation, pre-launch participation price discounts, pre-conversion reserve currency carveouts, and price-neutral launch pre-allocations, all specified in easy to understand commands and parameters, no programming necessary!

Currencies can be launched to accept a range of other currencies and with or without fractional reserve capabilities. Verus import/export protocol was designed to make integration with other block chains provable and easy, and there are already external bridges in progress that expect to enable automatic send/receive of any Verus network token between Verus and ETH/ERC20 tokens, where currencies can be sent between Verus and ETH, expressed as Verus currencies on the Verus network, and exposed automatically on ETH as an ERC20 token.

Preparing for Multi-chain, Dynamic Merge Mining

While this 0.7.1 testnet enables on-chain token currency definition, the same advances will soon be available on the Verus merge mining and multi-chain technology which was running on testnet last year. When enabled, this will make it possible for Verus network chains to operate fully independently, yet be dynamically merge mineable along with Verus and up to 20 other Verus network blockchains on a single hash, earning rewards and powering all chains simultaneously. All network blockchains will provide the same fractional currency capabilities, both for on-chain tokens as well as native zk-SNARK supporting currencies as well.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest","address":"alice@","amount":10}]'
```

Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"currency":"vrsctest","address":"alice@","amount":10},
 {"currency":"VRSCTEST","address":"bob@","amount":10}]'
```

Converting VRSCTEST to another currency, TESTCOIN:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@","amount":10,
 "convertto":"TESTCOIN"}]'
```

Preconverting to new currency, NEWCOIN, before it is active:

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@","amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@,

that I somehow launch in a way that maps it in a way that can be widely trusted to a specific, auditable store of gold.

I also would like to launch a token called mycoin@, which is something like a Kickstarter, where a business, "my", offers to attribute the coins some utility or product value if the purchase exceeds a certain level.

First, I could define the currency "gold" as follows:

```
./verus -chain=VRSCTEST definecurrency
'{"name":"gold","options":96,"currencies":["vrsctest"],"conversions":
[0.01],"minpreconversion":[1000],"preallocation": [{"mike@":50000000.00000000}]}'
```

of course, since this is a test currency, I send myself some to start. The identity of the currency must be funded with at least 10 VRSCTEST before sending the transaction returned from this command to

initiate a currency launch that will start at 50 blocks from when it was made (default), and that must have 1000 VRSCTEST preconverted at 0.01 VRSCTEST per GOLD in order to launch.

all of this happens as part of the mining process, since mining the blocks that launch a currency earn the 0.025% conversion fees of participation

in the launch, converting VRSCTEST to GOLD. I could send the following command before the block where GOLD token launches.

After it launches, the only way at present to create new tokens is with a centralized issuance option. To convert VRSCTEST to GOLD, you could issue the command:

```
./verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"mike@","convertto":"gold","preconvert":1,"amount":100}]'
```

that would effectively park my conversion until the token launches, at which point, I will either find 0.975 GOLD in my wallet, or I will have my VRSCTEST back.

Assuming it launches, and I later want to create mycoin, which can be converted to with either GOLD or VRSCTEST, I can create mycoin with:

```
./verus -chain=VRSCTEST definecurrency '{"name":"mycoin","options":96,
"proofprotocol":2,"currencies":["vrsctest", "gold"],"conversions": [0.01,
1.0],"minpreconversion": [1000,1000]}'
```

In "mycoin", I set proofprotocol to 2, which is PROOF_CHAINID. That means that the controller of the chain ID can mint new coins as follows:

```
./verus -chain=VRSCTEST sendcurrency "mycoin@"
'[{"address":"mike@","currency":"mycoin","mintnew":1,"amount":10000}]'
```

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE DELETE THE FOLLOWING**

DIRECTORIES BEFORE RUNNING THIS NEW UPDATE:

Linux:

~/.komodo/VRSCTEST
~/.verustest

MacOS

Users/{YourUserName}/Library/Application Support/Komodo/VRSCTEST
Users/{YourUserName}/Library/Application Support/VerusTest

Windows

"%APPDATA%\Komodo\VRSCTEST
"%APPDATA%\VerusTest

Launching the testnet:

./verusd -chain=VRSCTEST

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/7a46a8f579b45a08eaf9a20f138ff75c80590740959803c8c93a5f54810af3ad/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/26cb5bd36f7dcd4c8db49baaf427afea0aa87dc518dfac3bac93950d204944f9/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/2b07c7505f0c67d8957420677f85bb2a1947029494b01c2707fdb3d0c2644aaa/detection>

Windows:

<https://www.virustotal.com/#/file/1bf02b7b5022f3b4e9b072e604a6a94656a9dc9188d1625e557fe304bdf04067/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To fin...

Assets 6

v0.7.0-4



[Asherda](#)

[v0.7.0-4](#)

[ab82cc9](#)

[v0.7.0-4](#)

At block 1053660, the Verus Mainnet Protocol upgraded to the new Komodo Notary Nodes for Komodo dPoW notarization, season 4, and a number of new capabilities, as well as a change to the Verus Proof of Power staking and mining hash that is now live on Verus mainnet.

The network upgrade activated the following features:

- Added support for season 4 Komodo notary node operators
- Coinbase shielding will no longer be required on any coinbases from the past that have not yet been shielded.
- Staking will now work on all normal ID balances, enabling full use of IDs for storing and staking funds.
- A new “sendcurrency” API for the command line provides more control when sending from and to multiple addresses or identities.
- “updateidentity` will now be able to change the case of characters in the name of a VerusID in the global locale.
- Revocation identity will now lose all control over an ID after revocation, including the ability to still change the revocation identity itself, which it used to retain. After revocation now, the recovery identity will have full control over the revoked ID.
- Staking and hashing consensus protocol updates with VerusHash 2.2

New in v0.7.0-4

- Validate ID destination on `sendfrom`, `sendtoaddress`, `sendmany`, and `z_sendmany`

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/97fec5a44e617452d48e13f0c00736430d904a72d85fc61731fd01452de77880/detection>

Linux-AMD64:

https://www.virustotal.com/#/file/3c8134361c965ab87751f817a097c1825ca61e2a521bfd1ed9382fb_c44638920/detection

Linux-ARM64:

<https://www.virustotal.com/#/file/c5c70239f6cb94cd1ab8cb109597f609a198f1bce644fa3434f50426f9fa8724/detection>

Windows:

<https://www.virustotal.com/#/file/418d4da1a4113b29f7c7ce490f8aa527d6217f2d85fd10ff639b33110ab5eba0/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted text file.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AR1XEAABQR8XbEix6eohYN9m6BbLjcdUkCZH/NYw1/mfBq/miT6SewPg4PvaJHsUI08FLB8kQDpa9eKg
lWJN9DTfvpWlMH9Z /Downloads/Verus-CLI-Linux-v0.7.0-4-amd64/Verus-CLI-Linux-
v0.7.0-4-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AR1XEAABQR8XbEix6eohYN9m6BbLjcdUkCZH/NYw1/mfBq/miT6SewPg4PvaJHsUI08FLB8kQDpa9eKg
lWJN9DTfvpWlMH9Z
3c8134361c965ab87751f817a097c1825ca61e2a521bfd1ed9382fbc44638920
```

The result will be true if the signature is valid.

true

Assets 6

SECURITY UPDATE - v0.7.0-3 - MANDATORY

 [Asherda](#)

[v0.7.0-3](#)

[5611c0a](#)

SECURITY UPDATE - v0.7.0-3 - MANDATORY

At block 1053660, the Verus Mainnet Protocol upgraded to the new Komodo Notary Nodes for Komodo dPoW notarization, season 4, and a number of new capabilities, as well as a change to the Verus Proof of Power staking and mining hash that is now live on Verus mainnet.

Although the network is running smoothly after the upgrade, an issue was identified and resolved in this release. In addition, a couple of user features, including restoring "-pubkey=hexpubkey" as a parameter for redirecting staking and mining rewards are included.

The network upgrade activated the following features:

- Added support for season 4 Komodo notary node operators
- Coinbase shielding will no longer be required on any coinbases from the past that have not yet been shielded.
- Staking will now work on all normal ID balances, enabling full use of IDs for storing and staking funds.
- A new “sendcurrency” API for the command line provides more control when sending from and to multiple addresses or identities.
- “updateidentity` will now be able to change the case of characters in the name of a VerusID in the global locale.

- Revocation identity will now lose all control over an ID after revocation, including the ability to still change the revocation identity itself, which it used to retain. After revocation now, the recovery identity will have full control over the revoked ID.
- Staking and hashing consensus protocol updates with VerusHash 2.2

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/08254d35e8693eb304b35d1cee55f03761b78ad76dc5e25d6e42b300fcdd4e50/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/aa4842153c54510626b171128265059da30ae2bbfca974ceecc4348abb67e544/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/8a23406c8726ccf2fc09a20f298590d334ea00dbbca02eedfae3eb8349d95/detection>

Windows:

<https://www.virustotal.com/#/file/d274514424e2334a7695e74c20569e9267158f22303021539060da6d82ab482c/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted text file.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AfQaEAABQR9cFt4qNU1KD14h1vC1LhfICrm9LWuzcyW8Wh+zsU/B0Qji9pUp0PRBnEEH4aQWlfRgeS4j
2sByVDix3VMwj2aM /Downloads/Verus-CLI-Linux-v0.7.0-3-amd64/Verus-CLI-Linux-
v0.7.0-3-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AfQaEAABQR9cFt4qNU1KD14h1vC1LhfICrm9LWuzcyW8Wh+zsU/B0Qji9pUp0PRBnEEH4aQWlfRgeS4j
2sByVDix3VMwj2aM
d509506edcf48a3ba64d814916f98e46105a0f6b437635cf5d2f2247e1574d7a
```

The result will be true if the signature is valid.

true

Assets 6

v0.7.0-1

 [Asherda](#)
[v0.7.0-1](#)
[9d4787b](#)
[v0.7.0-1](#)

**MAKE SURE TO UPGRADE BY June 14TH, OR YOU MAY NEED TO
RESYNCHRONIZE THE BLOCKCHAIN IF YOU ARE RUNNING IN NATIVE MODE.**

At block 1053660, the Verus Mainnet Protocol will upgrade to the new Komodo Notary Nodes for Komodo dPoW notarization, season 4, and a number of new capabilities, as well as a change to the Verus Proof of Power staking and mining hash will go live on the Verus mainnet as well.

The network upgrade will activate the following features:

- Added support for season 4 Komodo notary node operators
- Coinbase shielding will no longer be required on any coinbases from the past that have not yet been shielded.
- Staking will now work on all normal ID balances, enabling full use of IDs for storing and staking funds.

- A new “sendcurrency” API for the command line provides more control when sending from and to multiple addresses or identities.
- “updateidentity` will now be able to change the case of characters in the name of a VerusID in the global locale.
- Revocation identity will now lose all control over an ID after revocation, including the ability to still change the revocation identity itself, which it used to retain. After revocation now, the recovery identity will have full control over the revoked ID.
- Staking and hashing consensus protocol updates with VerusHash 2.2

Notable Changes:

- Fix Komodo S4 notary count
- `sendcurrency` does not require coinbase funds to be shielded.

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/74ccab3336fc62efb8fb4cf1a37f3c50eba59030c906bf5475227e9276d29e4b/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/3cc1b3fd86368bb6a6c69f385e3411495974f4920dda96d9e8e6d48ef3aa0143/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/300379f4530e3a7f9379a55ffb3bb88ff7c1cb7806421bbfa8f3c1bb588bece7/detection>

Windows:

<https://www.virustotal.com/#/file/56dfa302af9a6185406c3db7102b75c77e67187d8d674b3c47374f8c28b90056/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted text file.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
AfcIEAABQSBXNoiaxzhPSEo99oIkVNEWvQP1ebMA5l4TN0lsUzi+Djo09AcsQjAG9gF/
HuVJ1S5gAgaQN28Pwj6ExYWoY/dm /Downloads/Verus-CLI-Linux-v0.7.0-1-amd64/Verus-
CLI-Linux-v0.7.0-1-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
AfcIEAABQSBXNoiaxzhPSEo99oIkVNEWvQP1ebMA5l4TN0lsUzi+Djo09AcsQjAG9gF/
HuVJ1S5gAgaQN28Pwj6ExYWoY/dm
0587c3da3d0f64015be98e097353a24d0581b9fbc1f3a788adf31a408c643023
```

The result will be true if the signature is valid.

true

Assets 6

v0.7.0 Verus Network Upgrade - MANDATORY UPDATE FOR MAINNET BY JUNE 14TH

 [Asherda](#)

[v0.7.0](#)

[c1057a5](#)

[v0.7.0 Verus Network Upgrade - MANDATORY UPDATE FOR MAINNET BY JUNE 14TH](#)

MAKE SURE TO UPGRADE BY June 14TH, OR YOU MAY NEED TO RESYNCHRONIZE THE BLOCKCHAIN IF YOU ARE RUNNING IN NATIVE MODE.

At block 1053660, the Verus Mainnet Protocol will upgrade to the new Komodo Notary Nodes for Komodo dPoW notarization, season 4, and a number of new capabilities, as well as a change to the Verus Proof of Power staking and mining hash will go live on the Verus mainnet as well.

The network upgrade will activate the following features:

- Added support for season 4 Komodo notary node operators
- Coinbase shielding will no longer be required on any coinbases from the past that have not yet been shielded.
- Staking will now work on all normal ID balances, enabling full use of IDs for storing and staking funds.
- A new “sendcurrency” API for the command line provides more control when sending from and to multiple addresses or identities.
- “updateidentity” will now be able to change the case of characters in the name of a VerusID in the global locale.
- Revocation identity will now lose all control over an ID after revocation, including the ability to still change the revocation identity itself, which it used to retain. After revocation now, the recovery identity will have full control over the revoked ID.
- Staking and hashing consensus protocol updates with VerusHash 2.2

Disclaimer

This is experimental and unfinished software. Use at your own risk! No warranty for any kind of damage!

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The enclosed copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MacOS:

<https://www.virustotal.com/#/file/67f38974426e598b60f3cc97ca4a0fd80f8291e24c6738ecea59ddf0055172e1/detection>

Linux-AMD64:

<https://www.virustotal.com/#/file/29919bf5117ec7aa88b86eae07e83d203909cd92a8ccb0a02911633e761cdff0/detection>

Linux-ARM64:

<https://www.virustotal.com/#/file/a9d5f4cff70ba5b7be4d236ae692affa846336a44f6ba1a398742d87fe3ec352/detection>

Windows:

<https://www.virustotal.com/#/file/7a7634d3bcc29b9a10190fd58a37be53ef686f7cb80293a0c4feb6aed9a056b4/detection>

Avast and Kaspersky may flag the software as "not-a-virus" or "PUP". These are warnings that you are installing mining software, which may be installed by a third party to exploit your PC.

To find out more about the false positives, review the following resources:

<https://blog.malwarebytes.com/detections/pup-optional-bitcoinminer/>

<https://www.kaspersky.com/blog/not-a-virus/18015/>

Verifying Downloads

A txt file containing the signer, standard sha256 file checksum, and signature, is included for each download. These packages have been signed with the identity "Verus Coin Foundation Releases@".

1. Extract downloaded archive
2. Verify signature for the extracted archive using the extracted text file.
3. Extract archive to desired directory

CLI examples

Verifying file directly

```
./verus verifyfile "Verus Coin Foundation Releases@"
Afn5DwABQSBdcb9HEXsztSaPQLtxhwA1U31x7ahYamF+DqxIn3j+pk/WQEwYd5pDgjp5AJ/
cpKWIfCVi468j1PbZ5WeJ /Downloads/Verus-CLI-Linux-v0.7.0-amd64/Verus-CLI-Linux-
v0.7.0-amd64.tar.gz
```

Verifying using a checksum

```
./verus verifyhash "Verus Coin Foundation Releases@"
Afn5DwABQSBdcb9HEXsztSaPQLtxhwA1U31x7ahYamF+DqxIn3j+pk/WQEwYd5pDgjp5AJ/
cpKWIfCVi468j1PbZ5WeJ
f769349015e0f4b8a6da9b071e3342f2043bd6eec35a69a97de7c7b4fd910148
```

The result will be true if the signature is valid.

true

Assets 6

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
 - [Privacy](#)
 - [Security](#)
 - [Status](#)
 - [Docs](#)
 - Contact GitHub
 - [Pricing](#)
 - [API](#)
 - [Training](#)
 - [Blog](#)
 - [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)



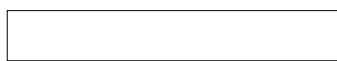
Sign in

Sign up

-
-
-
- [Code](#)
- [Issues 21](#)
- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v0.7.3-3



[v0.7.3-3](#)
[01e44d4](#)
[v0.7.3-3](#)

Mainnet Updates

v0.7.3-3 is a CRITICAL update with fixes for potential, targeted DOS attacks that do not affect overall network security, but should be considered mandatory for exchanges, pools, and businesses using the Verus network. Previous 0.7.3-x releases should no longer be used and should be replaced by this version.

While this release includes testnet functionality for Verus PBaaS, described below. This is a mainnet focused release that does not fix any issues on testnet. Testnet issues will be addressed in upcoming optional releases.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains

with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the

same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{ "currency": "vrsctest", "address": "alice@", "amount": 10 }]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{ "currency": "vrsctest", "address": "alice@:private", "amount": 10 },
{ "currency": "VRSCTEST", "address": "bob@:private", "amount": 10 }]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address": "bob@", "amount": 10,
"convertto": "VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10,
"convertto": "BTC", "via": "VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "alice@", "amount": 10,
"convertto": "NEWCOIN", "preconvert": true}]'
```

Assets 6

v0.7.2-8

 [Asherda](#)

[v0.7.2-8](#)

[d3cfffbe](#)

[v0.7.2-8](#)

Announcing MANDATORY CLI upgrade v0.7.2-8 - THIS UPDATE IS CONSIDERED MANDATORY - ALL NODE OPERATORS SHOULD UPDATE AS SOON AS POSSIBLE
This release includes the following improvements and updates:

- Critical updates
- Fixed fetch-bootstrap temp file cleanup on Windows.
- Fixed fetch-bootstrap checksum verification on MacOS

The Verus DeFi Testnet -- THERE IS NO TESTNET RESET

All new 'sendcurrency' features described above now work on both mainnet and testnet. In addition, testnet can source funds for DeFi features from private z-addresses as well.

This is the most groundbreaking testnet the Verus project has ever released, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a

list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.
- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.

- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"currency":"vrsctest", "address":"alice@", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I h...

Assets 6

v0.7.2-7

 [Asherda](#)

[v0.7.2-7](#)

[3fde9bf](#)

[v0.7.2-7](#)

Announcing NON-MANDATORY CLI upgrade v0.7.2-7, with fixes necessary to continue participating in the current testnet. Mainnet features are effectively unchanged from v0.7.2-6.

The Verus DeFi Testnet -- THERE IS NO TESTNET RESET

All new 'sendcurrency' features described above now work on both mainnet and testnet. In addition, testnet can source funds for DeFi features from private z-addresses as well.

This is the most groundbreaking testnet the Verus project has ever released, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Using it is simple. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts,

protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.

- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide

time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}, {"currency": "VRSCTEST", "address": "bob@", "amount": 10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10, "convertto": "VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once.

So, let's hypothetically assume I have 3 IDs, one named gold@, one named mycoin@, and one named mike@. I would like to have one currency, gold@, that I somehow launch in a way that maps i...

Assets 6

v0.7.2-6

 [Asherda](#)

[v0.7.2-6](#)

[5b13e56](#)

[v0.7.2-6](#)

Announcing NON-MANDATORY CLI upgrade v0.7.2-6, with primarily testnet improvements. Mainnet features are effectively unchanged from v0.7.2-4, with only "z*" added as a new, supported wildcard to the `z_getbalance` API.

This release includes the following improvements and updates for testnet:

- Add `getcurrencybalance` API to get multi-currency balances on transparent/private addresses and identities.
- Optimize multi-currency operators and UTXO selection
- Display unlocked balance in `getwalletinfo`
- Add support for "z*" wildcard in `z_getbalance`

The Verus DeFi Testnet -- THERE IS NO TESTNET RESET

All new 'sendcurrency' features described above now work on both mainnet and testnet. In addition, testnet can source funds for DeFi features from private z-addresses as well.

This is the most groundbreaking testnet the Verus project has ever released, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide

availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Using it is simple. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.
- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.

- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"currency":"vrsctest", "address":"alice@", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Curren...

Assets 6

v0.7.2-5

 [Asherda](#)

[v0.7.2-5](#)

[1de9f7c](#)

v0.7.2-5

Announcing NON-MANDATORY CLI upgrade v0.7.2-5, with fixes that only apply to testnet. Mainnet features are unchanged from v0.7.2-4.

This release includes the following improvements and updates:

- Fix testnet transaction funding issue for multi-currency `sendcurrency` commands
- Fix testnet currency conversions

The Verus DeFi Testnet -- THERE IS NO TESTNET RESET

All new 'sendcurrency' features described above now work on both mainnet and testnet. In addition, testnet can source funds for DeFi features from private z-addresses as well.

This is the most groundbreaking testnet the Verus project has ever released, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Using it is simple. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus

Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.

- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that

we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}, {"currency": "VRSCTEST", "address": "bob@", "amount": 10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10, "convertto": "VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an ID of the same name. The controller of this ID is the only one who can create a currency of that name, and they can only do so once. So, let's hypothetically assume I have 3 IDs, one...

Assets 6

v0.7.2-4

 [Asherda](#)

[v0.7.2-4](#)

[bcd146e](#)

[v0.7.2-4](#)

v0.7.2-4 WITH THE WORLD'S FIRST FRIENDLY-NAME, zk-SNARK ADDRESSES FOR MAINNET AND TESTNET

Announcing NON-MANDATORY CLI upgrade v0.7.2-4 with the world's first friendly-name private z-address support for sending and receiving funds and messages, including the following improvements and updates:

- `sendcurrency` can now use sapling addresses as native currency funding sources and destinations for any transaction.
- An identity's z-address can now be referenced as a funding source or destination by appending `:private` to an ID (eg. `"idname@:private"`). This form of z-address will also be able to be used with the `z_getbalance` command.
- New wildcards can also be used as funds sources in the '`sendcurrency`' command and parameters to `z_getbalance` that include:
 - All transparent addresses in the wallet (both R-addresses and IDs)
`"*"`
 - All R-addresses in the wallet
`"R*"`
 - All IDs controlled by the wallet
`"i*"`
- `sendcurrency` will now be an asynchronous API that will return an operation ID, as `z_sendmany` does, and complete in the background. It will also have its status available through `z_getoperationstatus`.

NOTE: Due to the zero knowledge nature of z-addresses, a z-address is an endpoint in your VerusID which can be changed if you revoke and recover an ID, funds held on a private address or endpoint are not able to be redirected to the newly recovered ID unless you still have control over the z-address itself. Transparent funds controlled by an ID are recovered when the ID is revoked and recovered, even if you have lost the keys. This is a fundamental limitation of zero-knowledge privacy, not an error and not something to be fixed. Please consider this when using friendly-name private addresses as funds and not just message endpoints.

The Verus DeFi Testnet -- THERE IS NO TESTNET RESET

All new 'sendcurrency' features described above now work on both mainnet and testnet. In addition, testnet can source funds for DeFi features from private z-addresses as well.

This is the most groundbreaking testnet the Verus project has ever released, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus

Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.

- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that

we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

**Warning: All testnet coin...

Assets 6

v0.7.2-3

 [Asherda](#)
[v0.7.2-3](#)
[7d66ad8](#)
[v0.7.2-3](#)

Announcing NON-MANDATORY upgrade v0.7.2-3 with the following improvements and fixes:

- Fixes a rare issue in wallet synchronization for IDs that have been transferred between wallets in a specific combination of transfers and spends of prior outputs. In the cases seen, the wallet maintained some outputs as unspent, when in fact they were spent on the blockchain. We have seen a very small number of people hit this issue after transferring IDs between owners and/or separate wallets and having spent prior outputs on transactions that still have valid UTXOs to the ID. If you have encountered such an issue, this version will

properly synchronize the wallet with a full rescan. If you have not, use of this version will prevent the issue from ever occurring.

- Stricter updateidentity parameter checks to ensure that revocation and recovery IDs are already registered, valid IDs on the blockchain before accepting a change

Testnet Reset

The testnet was deleted and relaunched on v0.7.2-2. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO v0.7.2-2, PLEASE FOLLOW THE INSTRUCTIONS AT THE BOTTOM OF THE RELEASE NOTES BEFORE RUNNING THIS UPDATE**

The New Verus DeFi Testnet

This is the most groundbreaking testnet the Verus project has ever announced, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Using it is simple. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts,

protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.

- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide

time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}, {"currency": "VRSCTEST", "address": "bob@", "amount": 10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10, "convertto": "VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional curr...

v0.7.2-2

 [Asherda](#)

[v0.7.2-2](#)

[9a9d036](#)

[v0.7.2-2](#)

Verus v0.7.2-2 is a mandatory for pools and notaries, but not generally mandatory Verus node and wallet upgrade, with support for the new Verus testnet (VRSCTEST) with fully decentralized finance, advanced ID capabilities, and DeFi liquidity system integrated into the new network protocol. While most of the changes in this release are focused on testnet, the mainnet support does include some improvements and a fix for the getblocktemplate API used by pools

Testnet Reset

The testnet was deleted and relaunched on v0.7.2-2. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO v0.7.2-2, PLEASE FOLLOW THE INSTRUCTIONS AT THE BOTTOM OF THE RELEASE NOTES BEFORE RUNNING THIS UPDATE**

The New Verus DeFi Testnet

This is the most groundbreaking testnet the Verus project has ever announced, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any

full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.

- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.
- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again

under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency":"vrsctest", "address":"alice@", "amount":10}, {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency

To create a currency of a specific name, you need an...

Assets 6

v0.7.2-1

 [Asherda](#)
[v0.7.2-1](#)
[8b5a602](#)
[v0.7.2-1](#)

This update is required to continue participating on the testnet launched in v0.7.2

Verus v0.7.2-1 is a non-mandatory but highly recommended Verus node and wallet upgrade, with fully decentralized finance and liquidity system activated on the new Verus testnet. While most of the new capabilities in this release are focused on testnet, the mainnet support does include some performance improvements and also fixes an issue that may occur in an edge case of revocation and recovery of IDs, which can result in a revoked ID being irrecoverable from an earlier version than v0.7.2.

Testnet Reset

The testnet was deleted and relaunched on v0.7.2. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO v0.7.2, PLEASE FOLLOW THE INSTRUCTIONS AT THE BOTTOM OF THE RELEASE NOTES BEFORE RUNNING THIS UPDATE**

The New Verus DeFi Testnet

This is the most groundbreaking testnet the Verus project has ever announced, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API, `getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.
- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.
- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to

enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}, {"currency": "VRSCTEST", "address": "bob@", "amount": 10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10, "convertto": "VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "bob@", "amount": 10, "convertto": "BTC", "via": "VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address": "alice@", "amount": 10, "convertto": "NEWCOIN", "preconvert": ...}]'
```

Verus v0.7.2 with Verus DeFi Platform and Theft-resistant VerusIDs on Testnet

 [Asherda](#)

[v0.7.2](#)

[6b02f06](#)

[Verus v0.7.2 with Verus DeFi Platform and Theft-resistant VerusIDs on Testnet](#)

We are excited to announce Verus v0.7.2, non-mandatory but highly recommended Verus node and wallet upgrade, with fully decentralized finance and liquidity system activated on the new Verus testnet. While most of the new capabilities in this release are focused on testnet, the mainnet support does include some performance improvements and also fixes an issue that may occur in an edge case of revocation and recovery of IDs, which can result in a revoked ID being irrecoverable from an earlier version than v0.7.2.

Testnet Reset

The testnet was deleted and relaunched on this release. **IF YOU HAVE LAST LAUNCHED VRSCTEST FROM A VERSION PRIOR TO THIS, PLEASE FOLLOW THE INSTRUCTIONS AT THE BOTTOM OF THE RELEASE NOTES BEFORE RUNNING THIS UPDATE**

The New Verus DeFi Testnet

This is the most groundbreaking testnet the Verus project has ever announced, and we believe it holds the potential to improve and reshape the DeFi industry. There are truly so many new capabilities and blockchain firsts working now on testnet that it's hard to adequately cover them all. Below is a list of new capabilities active on the v0.7.2 testnet that will be on mainnet in the near future when considered fully hardened, tested, and ready for worldwide, public scale. Until then, Verus testnet is a fully decentralized blockchain network with 50% proof of work, 50% proof of stake, worldwide availability, and is running all of the new Verus protocols that are being tested and hardened for mainnet. All currencies and assets on the Verus testnet have absolutely no value and may be reset at any time in the interest of testing for mainnet. If you would like a representative basket of Verus testnet currencies that you can use to try out all of the new protocols from anywhere in the world, simply visit the Verus Discord and ask in the #pbaas-development channel. You can use this link as an invite to the Verus Discord: <https://discord.gg/gecZsCf>

- **DeFi and cross-currency liquidity integrated with blockchain processing, fees, VerusID, and Verus PoP, 51% hash-attack resistant consensus algorithm** - This is a massive set of platform capabilities that take DeFi further than any project in crypto today. We will list it here and describe it below. Using it is simple. Understanding the implications of what such a protocol can do when rent free and integrated in a truly decentralized public blockchain is a process of realizations that is likely to be similar to layers of an onion. We will do our best to explain below.
- **System-wide queries for fractional basket currencies for arbitrage, investment discovery, or conversion bargain hunting** - The new RPC API,

`getcurrencyconverters` (final name at mainnet, TBD), enables instant queries of any full node for all registered currencies above a fixed liquidity threshold that contain one or more reserve currencies. This new API enables wallets or applications to easily query the entire blockchain network of fractional currencies, determine which provides the best conversion rate at acceptable liquidity, and either convert at the best possible rates worldwide, or select only better-than-market rates and arbitrage to earn.

- **Enhanced VerusIDs with Verus Vault technology** - This capability extends the already groundbreaking VerusID revocable, recoverable, provable, friendly name blockchain addresses and identities to include an advanced time lock capability that does not prevent staking on locked funds, enabling easy implementation of things like vesting, trusts, protocols for theft-proofing funds, simplifying estate planning, and much more. With Verus Vault technology, your funds, identity, and estate planning can be done in a fully self-sovereign manner, with the peace of mind in knowing you are protected from lost or stolen private keys.
- **Verus Fee Pooling technology** A sustainable mining and staking fee and revenue model while strengthening, rather than creating risk of weakening blockchain security, by preserving immediate incentive to maximize per-block fees, while smoothing fee distribution to miners and stakers over time, fully addressing the legitimate concerns expressed and still not addressed by Vitalik Buterin with respect to ETH and ETH 2.0 (<https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/>). In the fee pooling model, which is an extremely powerful, yet simple solution to enabling a mining and staking economy, all fees are gathered from transactions in a block as usual, then the fees, also calculated as usual, are added to a common, persistent pool of collected fees, and the current block miner or staker takes up to 1% of the aggregate fee pool.

Verus Vault - Advanced Time Locking of Identities

On the new Verus testnet, we have introduced a capability we call Verus Vault, which both provides options for a theft-resistant workflow of identity management, but also enables many use cases from trusts to vesting schedules, to escrow alternatives, estate planning and more. The concept is simple, but the ramifications when combined with the already unique revocation and recovery capabilities offered by VerusID on mainnet are immense.

The Verus Vault use case, which was originally described in the conception of the feature, is the ability to timelock an ID, which would prevent any spending of any funds controlled by that ID, while still allowing the ID and all its funds to be used for staking. If the timelock was set for one day when locked, the unlock operation itself will require a waiting period of one day, which can only be circumvented by a revocation followed by recovery operation. In practice, this means that even if someone compromises your private keys while an ID is locked, they cannot spend any of the funds (though they could stake them and delegate rewards to another address, which you could immediately see) unless they unlock the ID first and wait one day. If you notice that your ID has been unlocked by someone other than you, you can revoke and recover before any funds are lost at all. If you wanted to get extremely secure, to the point of covering a case where someone may steal your private keys without you knowing and then lay in wait until you unlock your identity, so they can pounce on your funds before you lock it again, you can actually modify the controlling private

keys of an identity before unlocking it, which ensures that the unlocked funds become usable again under the control of freshly made, uncompromised keys. While we believe this unique Verus technology actually provides for fully theft-proof workflows and protection of all assets on the network under the control of an ID, we will generally refer to the technology as "theft-resistant" and disclaim all guarantees of fitness for any particular purpose as part of the "experimental software" disclaimer.

In addition to use cases related to theft resistance and delayed lock opening, Verus Vault can be used to provide more standard forms of time-locking by simply time locking and unlocking in one operation, leaving the unlock period immediately counting down. This capability could be used to provide time locked funds to team members, children, employees, or structured in a way to provide time locked deal and payment terms. The uses for this technology are certainly beyond those that we have already thought of, and we look forward to seeing how people leverage this powerful new protocol capability.

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all wallet funds to two recipients (alice@ and bob@):**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"currency":"vrsctest", "address":"alice@", "amount":10}, {"currency":"VRSCTEST", "address":"bob@", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10, "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10, "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

Defining a Currency...

Assets 6

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
- [Privacy](#)
- [Security](#)
- [Status](#)
- [Docs](#)
- Contact GitHub
- [Pricing](#)
- [API](#)
- [Training](#)
- [Blog](#)
- [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)

-
-
-



Sign in

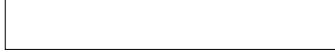
Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

- [Code](#)
 - [Issues 21](#)
 - [Pull requests](#)
 - [Actions](#)
 - [Projects](#)
 - [Wiki](#)
 - [Security](#)
 - [Insights](#)

Releases: VerusCoin/VerusCoin

Releases Tags



v0.8.0-1

 [Asherda](#)

[v0.8.0-1](#)

[108ad28](#)

[v0.8.0-1](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY v0.8.0-1 UPDATE

This release includes a testnet fix for the Ethereum bridge and is required for those who want to continue using the current testnet.

v0.8.0 and v0.8.0-1 Updates include:

1. Testnet reset with fixes for all reported PBaaS, DeFi, and advanced VerusID user issues. If you had reported issues, please verify that the issues you reported are addressed in this release.
2. Support for the bidirectional Ethereum gateway/bridge, which has been testing on private networks and which we hope to launch within a day or two after first vetting it on the release testing network with community test volunteers.
3. Support for the new `getvdxfid` RPC call, used along with your ID and published names to generate VDXF (Verus Data eXchange Format) keys, which community members used to create the world's first self-sovereign, completely decentralized, rent-free, non-cancelable social media profiles as can be seen here: <https://luckpool.net/profile/identity/mike.vrsc> . Work is underway to document the VDXF keys defined and used for profiles as well as the process of setting up your own. While the capability is already extremely powerful, we should remind everyone that Verus is a platform, not a social network. The core technology is exciting, but what will be even more exciting is when businesses and entrepreneurs leverage it to enable new, self-sovereign user experiences, applications, and accompanying businesses to enable the future, truly decentralized web.
4. Fix for the Electron certificate issue in the GUI wallet, which has been affecting prices and preventing BTC fee calculation.
5. Support for some additional, popular ERC20 currencies, in anticipation of more usage after release of the ETH bridge.
6. Fix in GUI for calculated balances of specific addresses sometimes showing lower than actual, even though wallet balance displayed correctly.
7. Additional hardening, fixes and improvement focused on mainnet and the Ethereum bridge.

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"'
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":100,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@
'[{"address":"<destination_address>","amount":<transaction_amount>}]''
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can

now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular

liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the p...

Assets 6

v0.8.0

 [Asherda](#)
[v0.8.0](#)

[7274b43](#)

[v0.8.0](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY, RECOMMENDED v0.8.0 UPDATE.

This release is mandatory for those who want to use the testnet, which will be reset as of v0.8.0.

Updates include:

1. Testnet reset with fixes for all reported PBaaS, DeFi, and advanced VerusID user issues. If you had reported issues, please verify that the issues you reported are addressed in this release.
2. Support for the bidirectional Ethereum gateway/bridge, which has been testing on private networks and which we hope to launch within a day or two after first vetting it on the release testing network with community test volunteers.
3. Support for the new `getvdxfid` RPC call, used along with your ID and published names to generate VDXF (Verus Data eXchange Format) keys, which community members used to create the world's first self-sovereign, completely decentralized, rent-free, non-cancelable social media profiles as can be seen here: <https://luckpool.net/profile/identity/mike.vrsc> . Work is underway to document the VDXF keys defined and used for profiles as well as the process of setting up your own. While the capability is already extremely powerful, we should remind everyone that Verus is a platform, not a social network. The core technology is exciting, but what will be even more exciting is when businesses and entrepreneurs leverage it to enable new, self-sovereign user experiences, applications, and accompanying businesses to enable the future, truly decentralized web.
4. Fix for the Electron certificate issue in the GUI wallet, which has been affecting prices and preventing BTC fee calculation.
5. Support for some additional, popular ERC20 currencies, in anticipation of more usage after release of the ETH bridge.
6. Fix in GUI for calculated balances of specific addresses sometimes showing lower than actual, even though wallet balance displayed correctly.
7. Additional hardening, fixes and improvement focused on mainnet and the Ethereum bridge.

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is ...

Assets 6

v0.7.4

 [Asherda](#)

[v0.7.4](#)

[77424d8](#)

[v0.7.4](#)

ANNOUNCING HIGHLY RECOMMENDED NON-MANDATORY, CRITICAL FOR NOTARIES AND POOLS UPDATE, v0.7.4.

Version v0.7.4 can improve a node's synchronization and this version also upgrades and resets the Verus testnet, preparing it for the testnet launch of the Ethereum/Rinkeby bridge and bringing Verus PBaaS multi-chain, multi-currency, Verus Vault ID timelocking, Verus MEV-resistant DeFi, and the Ethereum bridge much closer to a mainnet ready state.

This release fixes all known functionality issues reported from the previous testnet including some related to currency and blockchain launches, as well as being incorrectly able to reduce the timelock period on a Verus Vault locked ID. It also includes the ability to export an ID from one chain to another, using the “`exportid`”:`true` parameter in `sendcurrency`, enabling cross-chain use of VerusIDs. With this release, we will be resetting the Verus testnet. To use the new testnet, you must make sure to delete the `vrsctest` and `.verustest/pbaas` or `VerusTest\pbaas` folders beforehand.

In addition to being prepared for the Ethereum bridge launch, this release adds some new testnet capabilities, including the ability to send to IDs cross chain and export IDs from chain to chain using the “`exportid`”:`true` parameter with `sendcurrency`. IDs only need to be exported once and can then be used and managed directly as a forked ID on the destination chain.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@", "exportto":"PBaaSChainName", "exportid":"true", "amount":100, "currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>", "amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for

your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts

for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API re...

Assets 6

2 people reacted

v0.7.3-10

 [Asherda](#)
[v0.7.3-10](#)
[c3b2554](#)

[v0.7.3-10](#)

Announcing NON-MANDATORY, RECOMMENDED UPDATE FOCUSED ON NETWORK CONNECTIVITY AND ROBUSTNESS IMPROVEMENTS v 0.7.3-10.

v0.7.3-10 has very minimal updates targeted at networking robustness - faster response to notary inconsistencies and a check for detecting a corrupt index that if present, might otherwise prevent a node from staying synced.

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@" '[{"currency": "vrsctest", "address": "alice@", "amount": 10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z" '[{"currency": "vrsctest", "address": "alice@:private", "amount": 10}, {"currency": "VRSCTEST", "address": "bob@:private", "amount": 10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address": "bob..."}]
```

Assets 6

v0.7.3-9

[v0.7.3-9](#)
[f86f251](#)
[v0.7.3-9](#)

Announcing immediate availability of NON-MANDATORY, RECOMMENDED ESPECIALLY FOR NOTARIES, HIGH PEER COUNT PUBLIC NODES, AND POOLS, v0.7.3-9 Verus CLI.

In addition to more immediate and effective bad node banning behavior, v0.7.3-9 includes a fix for signrawtransaction that could sometimes result in failure of countersigning a multisig, ID-based raw transaction.

While this release does include network robustness and a multisig fix, we were also able to upgrade Verus multisig capabilities to an ease of use and level of capability beyond other networks we are aware of today. In this release, it is now possible, on both mainnet and testnet, to issue any “sendcurrency” command with the source being a multisig ID, for which you do not have all the keys present in your wallet. If you do, there will be a return error in operation status, from both GUI or CLI, and in that error will be the partially signed transaction, which can be used by applications or with copy and paste to send to other signers with partial authority over the source ID. On mainnet, destinations for such a transaction can include IDs, private addresses with memos to a z-address or ID:private, and transparent addresses. On testnet, such transactions can include all that is possible on mainnet as well as multiple currencies, currency conversions, and cross-chain sends.

To our knowledge, this release raises the industry standard for multisig capability and its support on public blockchain protocols to a new level, even before considering the multisig revocation and recovery capabilities.

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address": "<destination_address>", "amount": <transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts

for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that hav...

Assets 6

v0.7.3-8

 [Asherda](#)

[v0.7.3-8](#)

[942b01b](#)

[v0.7.3-8](#)

Announcing MANDATORY UPDATE v0.7.3-8, which will re-enable full Verus Proof of Power, including staking, at block 1576200, occurring on Thursday at approximately 5:00PM UTC time. YOU MUST UPDATE BEFORE THEN TO FOLLOW THE MAIN CHAIN.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised

a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{"currency":"vrsctest", "address":"alice@:private", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@:private", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

- **Sending VRSCTEST cross-chain to PBaaSChain:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"RXLYm4J6qi7yam9zXtkEkRwbvCrnWKGZuv", "amount":10,
 "exportto":"Bridge.PBaaSChain"}]'
```

- **Converting VRSCTEST cross-chain to PBaaSChain:**

```
verus -...
```

v0.7.3-7

 [Asherda](#)

[v0.7.3-7](#)

[2fcb26a](#)

[v0.7.3-7](#)

Announcing MANDATORY UPGRADE v0.7.3-7. This release fixes the bug that was exploited by over the last day which stalled the network. You must upgrade to this release. If you shut down prior to block 15679999 you should be able to run this release without any additional steps. If your wallet continued to run after block 15679999 (or is still running) you will need to bootstrap or otherwise re-sync.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee

conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.

- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{"currency":"vrsctest", "address":"alice@:private", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@:private", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

- **Sending VRSCTEST cross-chain to PBaaSChain:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":...]
```

v0.7.3-6

 [Asherda](#)

[v0.7.3-6](#)

[7a740d2](#)

[v0.7.3-6](#)

Announcing MANDATORY UPGRADE v0.7.3-6, REQUIRED FOR KOMODO NOTARY SEASON 5. Notaries, pools, and exchanges should upgrade to this version as soon as possible, before notary changes take effect on block 1562500. Failing to update will not prevent connection to the network, but updating in a timely manner will ensure a stronger, notarized connection to the Verus blockchain.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee

conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.

- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{"currency":"vrsctest", "address":"alice@:private", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@:private", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address":"bob@", "amount":10,
"convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10,
"convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10,
"convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

- **Sending VRSCTEST cross-chain to PBaaSChain:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"RXLYm4J6qi7yam9zXtkEkRwbv...']
```

v0.7.3-5

 [alexenglish](#)

[v0.7.3-5](#)

[ef42f6e](#)

[v0.7.3-5](#)

Announcing NON-MANDATORY GUI upgrade v0.7.3-5, required to continue participating in the PBaaS enabled testnet. v0.7.3-5 resolves an ID wallet sync issue for mainnet that could occur when recovering a revoked ID, causing the wallet to double count some amounts. This issue could not result in either loss of funds nor a security issue. This release also resolves all known testnet issues on the current testnet and does not require a testnet reset.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee

conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.

- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{"currency":"vrsctest", "address":"alice@:private", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@:private", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

- **Sending VRSCTEST cross-chain to PBaaSChain:**

```
ver...
```

v0.7.3-4

 [Asherda](#)

[v0.7.3-4](#)

[72c69c8](#)

[v0.7.3-4](#)

Announcing NON-MANDATORY CLI upgrade v0.7.3-4, required for to continue participating in the PBaaS enabled testnet. Mainnet features are effectively unchanged from v0.7.3-3

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an

unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 2, "timelock": <Unlock block delay>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- Revoking an identity will clear its locked status, regardless of time delay or unlock height.
- A locked identity cannot revoke itself.

Conversion Queries

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves

- **E.g. BTC ETH:**

```
verus -chain=VRSCTEST getcurrencyconverters btc eth
```

Will return all currencies that have btc/eth markets at or above the liquidity threshold.

Sending and Converting Currency

Warning: All testnet coins/currencies have no value and will disappear whenever VRSCTEST is reset

The `sendcurrency` API can be used to send and convert funds.

- **Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):**

```
verus -chain=VRSCTEST sendcurrency "bob@"
'[{"currency":"vrsctest", "address":"alice@", "amount":10}]'
```

- **Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):**

```
verus -chain=VRSCTEST sendcurrency "*Z"
'[{"currency":"vrsctest", "address":"alice@:private", "amount":10},
 {"currency":"VRSCTEST", "address":"bob@:private", "amount":10}]'
```

- **Converting VRSCTEST to a fractional basket currency, VRSC-BTC using IDs as a funding source:**

```
verus -chain=VRSCTEST sendcurrency "*i" '[{"address":"bob@", "amount":10,
 "convertto":"VRSC-BTC"}]'
```

- **Converting VRSCTEST to another reserve, BTC through a fractional currency, VRSC-BTC:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"bob@", "amount":10,
 "convertto":"BTC", "via":"VRSC-BTC"}]'
```

- **Preconverting to new currency, NEWCOIN, before it is active:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"alice@", "amount":10,
 "convertto":"NEWCOIN", "preconvert":true, "refundto":"alice@"}]'
```

- **Sending VRSCTEST cross-chain to PBaaSChain:**

```
verus -chain=VRSCTEST sendcurrency "*"
'[{"address":"RXLYm4J6qi7yam9zXtkEkRwbvCrnWKGZuv", "amount":10,
 "exportto":"Bridge.PBaaSChain"}]'
```

- **Converting VRSCTEST cross-chain to PBaaSChain:**

```
verus -chain=VRSCTEST sendcurrency "*" '[{"address":"RXLYm4J6qi7..."]'
```

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
 - [Privacy](#)
 - [Security](#)
 - [Status](#)
 - [Docs](#)
 - Contact GitHub
 - [Pricing](#)
 - [API](#)
 - [Training](#)
 - [Blog](#)
 - [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

-
-
-
-
- [Code](#)
- [Issues 21](#)
- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v0.9.1-1

 [Asherda](#)
[v0.9.1-1](#)
[81dac44](#)
[v0.9.1-1](#)

ANNOUNCING MAINNET AND PBAAS TESTNET UPGRADE v0.9.1-1

**CRITICAL FOR NOTARIES, POOLS, EXCHANGES,
MANDATORY FOR TESTNET USERS, HIGHLY
RECOMMENDED FOR ALL MAINNET USERS**

v0.9.1-1 includes testnet fixes for the Ethereum bridge launch with advanced features, addresses all known issues on testnet, and includes sync improvements for testnet and mainnet.

Testnet Featuring Verus PBaaS

The most powerful and interoperable currency, blockchain, and ERC20 launch, scale, and MEV-resistant DeFi platform to exist, all with no programming required!

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.1, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

Verus PBaaS Features Live on v0.9.1-1 Testnet

- Provable or pseudonymous, identity-based currency, liquidity pool, NFT and multi-blockchain, fully decentralized network.
- On-chain 100% decentralized, launches with fair launch and/or crowdfunding options, on-chain auto-refund option for missed targets, automatic Ethereum ERC20 contract deployment, auto-created, MEV-resistant, liquidity baskets and even auto-bridged, merge mineable, fully independent rent-free PBaaS blockchain launches.
- All DeFi AMM conversions are verified via mining and staking as part of the L1 consensus protocol. All conversions in a single liquidity basket are calculated simultaneously for all transactions in any given block, meaning all participants get the same price in all directions of conversion, with a minimum conversion fee of 0.025% and a maximum of 0.05%.
- All currencies and identities are primitives at L1 and are validated and verified on UTXO transactions, which check all inputs and outputs just as single currency L1, such as Ethereum or Bitcoin check the native currency inputs and outputs of transactions.
- All currencies on all independent connected chains, once launched can be sent back and forth to other multi-currency capable networks (currently other independent PBaaS chains and Ethereum as ERC20s).
- Currencies on Verus can also be defined as "mapped currencies", which map 1:1 to an existing currency on Ethereum (eg. DAI), and can then be sent from Ethereum with simple transactions and received and used as the new "mapped currency", or sent back to Ethereum and used there as the original currency.

- Currency launches can raise funds in 3 ways. Each of these fundraising options creates a currency that is not 100% backed, and has a price that responds to market forces. 100% backed currencies do not change their relative price to the underlying basket reserve when they are converted to or from reserves. Fractionally backed currencies, which result from fundraising or endowing launch grants/pre-allocations to DAOs or entities such as foundations, whether time-locked or not, take their funding from a percentage of the reserve backing. Except in the case of a new blockchain launch, which creates new native currency for operating the proof of stake of the blockchain, any currency that uses fundraising options will be a fractionally backed currency which will appreciate when more people convert to it and depreciate when more convert from it. If a launch is refunded due to minimum participation options that are not met, no fundraising is received:
 - Pre-launch discount - all participants contributing to the launch before it goes live on its start block get a discount to the initial on-chain price of the currency. This is usually accompanied by a maximum cap on pre-conversion participation to ensure that the currency would still be in demand when it goes live.
- Pre-launch carve-out - this enables a percentage of each of the initial participation currencies to be carved out of the launch and total reserve ratio in a price neutral manner and sent directly to the launching identity upon successful launch.
- Pre-allocation - this enables some of the newly launched currency to be pre-allocated to one or more identities. If the currency is fractional, this happens in a price neutral manner, and again is subtracted from the reserve ratio. Recipients can optionally be time locked IDs for vesting or unlock periods.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"'
'[{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 100, "currency": "vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@
'[{"address": "<destination_address>", "amount": <transaction_amount>}]''
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited...

Assets 6

v0.9.1

 [Asherda](#)
[v0.9.1](#)
[623c69f](#)
[v0.9.1](#)

ANNOUNCING NON-MANDATORY, RECOMMENDED FOR MAINNET, MANDATORY FOR TESTNET PBAAS UPGRADE v0.9.1

Testnet Featuring Verus PBaaS -- The most powerful and interoperable currency, blockchain, and ERC20 launch, scale, and MEV-resistant DeFi platform to exist, all with no programming required!

v0.9.1 improves mainnet bootstrap and GUI backup/restore user experience but does not upgrade mainnet protocols and is not a required update for mainnet-only users. It is required to use the public testnet. IF YOU HAVE ALREADY USED TESTNET WITH PRIOR VERSIONS, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

Verus PBaaS Features Live on v0.9.1 Testnet

- Provable or pseudonymous, identity-based currency, liquidity pool, NFT and multi-blockchain, fully decentralized network.
- On-chain 100% decentralized, launches with fair launch and/or crowdfunding options, on-chain auto-refund option for missed targets, automatic Ethereum ERC20 contract deployment, auto-created, MEV-resistant, liquidity baskets and even auto-bridged, merge mineable, fully independent rent-free PBaaS blockchain launches.
- All DeFi AMM conversions are verified via mining and staking as part of the L1 consensus protocol. All conversions in a single liquidity basket are calculated simultaneously for all transactions in any given block, meaning all participants get the same price in all directions of conversion, with a minimum conversion fee of 0.025% and a maximum of 0.05%.
- All currencies and identities are primitives at L1 and are validated and verified on UTXO transactions, which check all inputs and outputs just as single currency L1, such as Ethereum or Bitcoin check the native currency inputs and outputs of transactions.
- All currencies on all independent connected chains, once launched can be sent back and forth to other multi-currency capable networks (currently other independent PBaaS chains and Ethereum as ERC20s).
- Currencies on Verus can also be defined as "mapped currencies", which map 1:1 to an existing currency on Ethereum (eg. DAI), and can then be sent from Ethereum with simple transactions and received and used as the new "mapped currency", or sent back to Ethereum and used there as the original currency.
- Currency launches can raise funds in 3 ways. Each of these fundraising options creates a currency that is not 100% backed, and has a price that responds to market forces. 100% backed currencies do not change their relative price to the underlying basket reserve when

they are converted to or from reserves. Fractionally backed currencies, which result from fundraising or endowing launch grants/pre-allocations to DAOs or entities such as foundations, whether time-locked or not, take their funding from a percentage of the reserve backing. Except in the case of a new blockchain launch, which creates new native currency for operating the proof of stake of the blockchain, any currency that uses fundraising options will be a fractionally backed currency which will appreciate when more people convert to it and depreciate when more convert from it. If a launch is refunded due to minimum participation options that are not met, no fundraising is received:

- Pre-launch discount - all participants contributing to the launch before it goes live on its start block get a discount to the initial on-chain price of the currency. This is usually accompanied by a maximum cap on pre-conversion participation to ensure that the currency would still be in demand when it goes live.
- Pre-launch carve-out - this enables a percentage of each of the initial participation currencies to be carved out of the launch and total reserve ratio in a price neutral manner and sent directly to the launching identity upon successful launch.
- Pre-allocation - this enables some of the newly launched currency to be pre-allocated to one or more identities. If the currency is fractional, this happens in a price neutral manner, and again is subtracted from the reserve ratio. Recipients can optionally be time locked IDs for vesting or unlock periods.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-

knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":100,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The...

Assets 6

v0.9.0-3

 [Asherda](#)

[v0.9.0-3](#)

[a2c4774](#)

[v0.9.0-3](#)

Announcing NON-MANDATORY, RECOMMENDED Verus v0.9.0-3. This release is considered non-mandatory but IS RECOMMENDED for mainnet users, Komodo notaries, and is recommended for testnet users to ensure the best experience with testnet.

This version fixes the following issues with v0.9.0-2

- Adds GUI support for seeing, making, and taking offers for currency or identity trades on the VerusID tab and when searching for identities.
- Fixes an issue where offers to buy or sell IDs were prematurely closed, even when expiry was set far in the future
- Fixes an issue with setidentitytimelock that caused it to return an error in most cases
- Ensures full rescanning of identities when using z_importwallet

New Command and API for Verus Vault - setidentitytimelock

v0.9.0-2 introduced a new command and RPC API, "setidentitytimelock". setidentitytimelock may be used to set a time unlock delay, set a lock until a specific block height, or unlock a delayed timelock on a VerusID under your control.

You can use setidentitytimelock in one of two ways:

```
verus setidentitytimelock '{"unlockatblock":1796500}' (returntx)
```

or

```
verus setidentitytimelock '{"setunlockdelay":1440}' (returntx)
```

or

```
verus setidentitytimelock '{"unlockatblock":0}' (returntx)
```

The first example will lock the ID until the block specified. If the block specified is in the past, the ID will not be locked.

The second example will set a 1440 block unlock delay, which will be needed to wait before spending funds controlled by the ID after unlocking the ID (about one day unlock time).

The third example will unlock an ID that has been locked with an unlock delay, and the unlock time will be set to the block height at which the ID was unlocked + the originally set unlock delay.

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`

- Windows 10: %AppData%\Roaming\Komodo\vrscatest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrscatest\, %AppData%\Roaming\VerusTest

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"'
'[{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 10
0, "currency": "vrscatest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@
'[{"address": "<destination_address>", "amount": <transaction_amount>}]''
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers...

v0.9.0-2

 [Asherda](#)

[v0.9.0-2](#)

[0c30a49](#)

[v0.9.0-2](#)

Announcing NON-MANDATORY, RECOMMENDED Verus v0.9.0-2. This release is considered non-mandatory but IS RECOMMENDED for mainnet users, Komodo notaries, and is mandatory for testnet users to ensure reliable connection to the current testnet.

TO REMAIN CONNECTED TO THE MAINNET VERUS BLOCKCHAIN, YOU MUST UPGRADE FULL NODES TO v0.9.0 OR LATER. IF YOU HAVE NOT UPDATED AND ATTEMPTED TO STAY CONNECT TO MAINNET, YOU MAY NEED TO BOOTSTRAP OR RESYNC YOUR NATIVE WALLET OR NODE AFTER UPGRADING.

New Command and API for Verus Vault - setidentitytimelock

v0.9.0-2 introduces a new command and RPC API, "setidentitytimelock". setidentitytimelock may be used to set a time unlock delay, set a lock until a specific block height, or unlock a delayed timelock on a VerusID under your control.

You can use setidentitytimelock in one of two ways:

```
verus setidentitytimelock '{"unlockatblock":1796500}' (returntx)
```

or

```
verus setidentitytimelock '{"setunlockdelay":1440}' (returntx)
```

or

```
verus setidentitytimelock '{"unlockatblock":0}' (returntx)
```

The first example will lock the ID until the block specified. If the block specified is in the past, the ID will not be locked.

The second example will set a 1440 block unlock delay, which will be needed to wait before spending funds controlled by the ID after unlocking the ID (about one day unlock time).

The third example will unlock an ID that has been locked with an unlock delay, and the unlock time will be set to the block height at which the ID was unlocked + the originally set unlock delay.

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new

age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidit...

Assets 6

v0.9.0-1



[Asherda](#)

[v0.9.0-1](#)

[d0aecf8](#)

[v0.9.0-1](#)

Announcing NON-MANDATORY, RECOMMENDED Verus v0.9.0-1. This release is considered non-mandatory but IS RECOMMENDED for mainnet users, Komodo notaries, and is mandatory for testnet users to ensure reliable connection to the current testnet.

TO REMAIN CONNECTED TO THE MAINNET VERUS BLOCKCHAIN, YOU MUST UPGRADE FULL NODES TO v0.9.0 OR LATER BY BLOCK 1796400, which is expected to occur on or around Monday, November 29th at 6:15PM UTC.

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute

transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, pr...

Assets 6

v0.9.0

 [Asherda](#)

[v0.9.0](#)

[4c7a0a6](#)

[v0.9.0](#)

Announcing MANDATORY Verus v0.9.0 Mainnet Release of Verus Vault and the on-chain NFT and ID Marketplace

TO REMAIN CONNECTED TO THE MAINNET VERUS BLOCKCHAIN, YOU MUST UPGRADE FULL NODES TO v0.9.0 OR LATER BY BLOCK 1796400, which is expected to occur on or around Monday, November 29th at 6:15PM UTC.

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your

funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@", "exportto":"PBaaSChainName", "exportid":true, "amount":100, "currency":vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":<destination_address>, "amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-

capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-...

Assets 6

v0.8.0-5

 [Asherda](#)
[v0.8.0-5](#)
[4864da8](#)
[v0.8.0-5](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY v0.8.0-5 UPDATE - ONLY REQUIRED FOR THOSE CONTINUING TO HELP GET TESTNET READY FOR MAINNET

We want to thank the Verus Community for bearing with the number of updates we've had to get the bridge to Ethereum working smoothly. This release both fixes the remaining known bridge issues and adds security to ensure that you don't send to an unspendable address on any network by using the wrong format for a destination.

The good news is that throughout the whole process of this first bridge release on testnet, we've been able to update through the issues we've seen without a reset and have not lost a single satoshi of test funds (which is not to say it is fully hardened quite yet). At this point, we hope and believe that this is the last update in this round of testnet, which will allow us to get back to focusing on the mainnet release and not asking those testing to update until we are very close to a mainnet release.

Once more, we will ask that you shut down the old version now, and we will restart the notarizations and transaction flow between Ethereum and Verus at 5 AM UTC time.

Thank you to everyone who helped with all the testing and exposing of issues that enabled us to make maybe even inconveniently quick progress.

If you miss the fork for this version at the above time you can get back on the correct chain by stopping Verus and deleting everything other than your wallet.dat file in your data directory at:

- Linux: `~/.Komodo/vrsctest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest` or `%AppData%\Komodo\vrsctest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 100, "currency": "vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address": "<destination_address>", "amount": <transaction_amount>}]'
```

```
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an

unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized "bridge" converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

Locking and Unlocking IDs

- **Time Lock:**

The timelock parameter defines the unlock height of the identity.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags": 0, "timelock": <Unlock block height>, "minimumsignatures": 1, "primaryaddresses": ["t-address"]}'
```

- **Time Delay:**

The timelock parameter defines how many blocks to delay an ID's unlock when the flags are set back to an unlocked state.

```
verus -chain=VRSCTEST updateidentity '{"name": "ID@", "flags...}
```

Assets 6

1 person reacted

v0.8.0-4

 [Asherda](#)

[v0.8.0-4](#)

[21d7af5](#)

[v0.8.0-4](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY v0.8.0-4 UPDATE.

This release has no mainnet changes, resolves issues that people using testnet have reported with the Ethereum/Rinkeby bridge, and is mandatory for those who wish to continue using testnet after transactions between the Verus and Rinkeby testnets resume flowing.

If you have pending transactions on the bridge, some requested conversions may come back as refunded, and you should receive all amounts expected, less fees. We will not reset the testnet, but within an hour after this availability, the network will resume transaction imports/exports to and from Rinkeby, and anyone who continues running the old version may need to resync to testnet from scratch to continue. If you have upgraded to v0.8.0-4 or do so before connecting again to testnet, you should only see improvement without disruption.

I want to thank everyone who participated already in using this testnet and bridge release. I hope we can keep that level of participation or more up, as it has been extremely helpful. Because of all that's happened on this testnet to date, we were able to make a lot of progress in finding and resolving the following testnet issues in this release:

1. Enable support for much larger proofs than was previously possible from systems that either have large proofs required or significantly variable length proofs. Having this capability in the daemon will pave the way for more types of cross-chain proving in the future as well.
2. Fix failed refunds of conversion transactions attempted after a fractional currency launches, but before it's preconversion phase is complete. If you sent transactions during this time, you may have stuck refunds that should flow after this upgrade.
3. Ensure that currencies are marked as "launch complete" immediately after all preconversions have been processed, enabling normal conversions in all directions from that point forward.
4. Fix an issue that left some import notarizations incomplete and caused cross-chain transfers to wait longer than expected after some imports.
5. Improve and validate hardening.

If you miss the fork for this version at the above time you can get back on the correct chain by stopping Verus and deleting everything other than your wallet.dat file in your data directory at:

- Linux: `~/.Komodo/vrsctest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`

- Windows 10: %AppData%\Roaming\Komodo\vrscatest or %AppData%\Komodo\vrscatest

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"
' [{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 10
0, "currency": "vrscatest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@
' [{"address": "<destination_address>", "amount": <transaction_amount>}]''
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the ...

Assets 6

v0.8.0-3

 [Asherda](#)
[v0.8.0-3](#)
[d6c4d90](#)
[v0.8.0-3](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY v0.8.0-3 UPDATE.

THIS UPDATE IS ONLY NEEDED FOR THOSE COMMUNITY MEMBERS USING TESTNET. THERE ARE NO MAINNET CHANGES IN THIS VERSION.

You will need this update to stay connected to testnet once we restart bridge traffic in about an hour. If you do stay connected on the old version and fork away from testnet, you should still be able to start with a reindex and rescan after upgrade.

v0.8.0 Updates include:

1. Testnet reset with fixes for all reported PBaaS, DeFi, and advanced VerusID user issues. If you had reported issues, please verify that the issues you reported are addressed in this release.
2. Support for the bidirectional Ethereum gateway/bridge, which has been testing on private networks and which we hope to launch within a day or two after first vetting it on the release testing network with community test volunteers.
3. Support for the new `getvdxfid` RPC call, used along with your ID and published names to generate VDXF (Verus Data eXchange Format) keys, which community members used to create the world's first self-sovereign, completely decentralized, rent-free, non-cancelable social media profiles as can be seen here: <https://luckpool.net/profile/identity/mike.vrsc> . Work is underway to document the VDXF keys defined and used for profiles as well as the process of setting up your own. While the capability is already extremely powerful, we should remind everyone that Verus is a platform, not a social network. The core technology is exciting, but what will be even more exciting is when businesses and entrepreneurs leverage it to enable new, self-sovereign user experiences, applications, and accompanying businesses to enable the future, truly decentralized web.
4. Fix for the Electron certificate issue in the GUI wallet, which has been affecting prices and preventing BTC fee calculation.
5. Support for some additional, popular ERC20 currencies, in anticipation of more usage after release of the ETH bridge.
6. Fix in GUI for calculated balances of specific addresses sometimes showing lower than actual, even though wallet balance displayed correctly.
7. Additional hardening, fixes and improvement focused on mainnet and the Ethereum bridge.

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and sta...

Assets 6

v0.8.0-2

 [Asherda](#)
[v0.8.0-2](#)
[292c723](#)
[v0.8.0-2](#)

Announcing immediate availability of GUI and CLI for all platforms of the NON-MANDATORY v0.8.0-2 UPDATE

This daemon release includes a fix that simplifies some bridge transactions, and is the version you should run on testnet when the Ethereum bridge does launch for a good experience.

v0.8.0 Updates include:

1. Testnet reset with fixes for all reported PBaaS, DeFi, and advanced VerusID user issues. If you had reported issues, please verify that the issues you reported are addressed in this release.
2. Support for the bidirectional Ethereum gateway/bridge, which has been testing on private networks and which we hope to launch within a day or two after first vetting it on the release testing network with community test volunteers.
3. Support for the new `getvdxfid` RPC call, used along with your ID and published names to generate VDXF (Verus Data eXchange Format) keys, which community members used to create the world's first self-sovereign, completely decentralized, rent-free, non-cancelable social media profiles as can be seen here: <https://luckpool.net/profile/identity/mike.vrsc> . Work is underway to document the VDXF keys defined and used for profiles as well as the process of setting up your own. While the capability is already extremely powerful, we should remind everyone that Verus is a platform, not a social network. The core technology is exciting, but what will be even more exciting is when businesses and entrepreneurs leverage it to enable new, self-sovereign user experiences, applications, and accompanying businesses to enable the future, truly decentralized web.
4. Fix for the Electron certificate issue in the GUI wallet, which has been affecting prices and preventing BTC fee calculation.
5. Support for some additional, popular ERC20 currencies, in anticipation of more usage after release of the ETH bridge.
6. Fix in GUI for calculated balances of specific addresses sometimes showing lower than actual, even though wallet balance displayed correctly.
7. Additional hardening, fixes and improvement focused on mainnet and the Ethereum bridge.

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\VRSC\`, `%AppData%\Roaming\VerusTest`

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and

fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with a...

Assets 6

Footer

© 2023 GitHub, Inc.

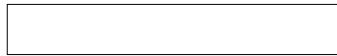
Footer navigation

- [Terms](#)
- [Privacy](#)

- [Security](#)
 - [Status](#)
 - [Docs](#)
 - Contact GitHub
 - [Pricing](#)
 - [API](#)
 - [Training](#)
 - [Blog](#)
 - [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

-

-
- [Code](#)
- [Issues 21](#)
- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v0.9.4-3



[v0.9.4-3](#)
[2521515](#)
[v0.9.4-3](#)

Announcing v0.9.4-3, UNCHANGED FOR MAINNET -- MANDATORY FOR CONTINUED TESTNET USE

This release has no material mainnet changes.

This release upgrades the testnet protocol to fix an issue with the Ethereum bridge and VRSCTEST mining and staking. Please update to v0.9.4-3 to continue mining and staking on testnet and the Ethereum bridge. This version also fixes an issue discovered when defining Ethereum NFT mapped tokens, which are not yet supported in the released Ethereum contracts, but are updated no in the daemon and ready to use when the contracts are upgraded, hopefully in the next few days.

NFT mapped tokens will allow you to define an Ethereum NFT as the token that controls revoke and recover for an ID. Similar to how you can define a token for tokenized ID control, an NFT mapped token, when defined, refers by contract and tokenid to a specific NFT on the Ethereum blockchain. Once defined as the currency for a specific ID or sub-ID and linked to the Ethereum NFT, that NFT will be able to be sent over from the Ethereum blockchain as a satoshi of that currency, sent to any address or ID on the Verus blockchain, exchanged trustlessly on the Verus on-chain P2P marketplace, and used to revoke and recover the ID which defined the token. Once the contracts are updated, you will be able to mint NFTs on Ethereum's Goerli testnet, and endow your Ethereum NFTs with all of the power of your specially named VerusID. This new technology allows you to protect your NFTs with VerusID's revoke and recover technology, bind your NFT to the pseudonymous ID you use to log into applications, use it to extend and publish much more than just the NFT, and generally take your Ethereum NFTs to the next level!

The rest of these release notes are the same as the prior release. Drop by the #pbaas-development channel to help us prepare for mainnet and start building your vision today!

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc`, but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt":
```

```
"b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral":  
""}, "identity": {"primaryaddresses":  
[ "RDnf7mH7RQki9b7PqdBD2Er6Wxv3DTawGr"], "minimumsignatures": 1, "name": "ID", "parent":  
": "vrsc-btc@"}}' false 0  
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":100,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an ...

v0.9.4-2

 [Asherda](#)

[v0.9.4-2](#)

[1541b4c](#)

[v0.9.4-2](#)

Announcing v0.9.4-2, UNCHANGED FOR MAINNET -- MANDATORY FOR TESTNET USE

This release has no material mainnet changes.

THE NEW ETHEREUM BRIDGE IS EXPECTED TO BE DEFINED WITHIN THE NEXT 24 HOURS AND FOR THE BRIDGE LIQUIDITY BASKET TO LAUNCH SHORTLY AFTER DEFINITION. IF YOU LEAVE A PRIOR VERSION NODE RUNNING UNTIL AFTER THE VETH BRIDGE IS DEFINED, YOU MAY FORK AWAY DUE TO CHANGES IN THIS RELEASE AND THEN NEED TO RESYNC TO TESTNET FROM SCRATCH TO CONTINUE.

For testnet, this is RC4 of the Verus PBaaS protocol, which includes some Ethereum bridge related fixes and a merge mining API change. With the API change, all merge mining nodes must upgrade to the new protocol to be compatible with each other and participate in cross-chain notarization rewards.

After the Ethereum bridge has been defined, you will be able to send currencies over to Verus testnet from Ethereum's Goerli network and also participate in the launch of the Ethereum bridge liquidity basket with VETH, USDC, or VRSCTEST by using `sendcurrency` with `"preconvert": true` on the command line or through the GUI. We recommend that you get some Goerli ETH using this facet or others (<https://goerli-faucet.mudit.blog/>), which will make your experience of all Verus PBaaS can offer a bit more well rounded when the Ethereum bridge is available once again.

Once the Goerli Ethereum liquidity basket (Bridge.vETH) activates on testnet, which is a 100% backed basket composed of 1/3rd each of Verus, Ethereum, and USDC (all testnet tokens), you can use it from Metamask on Goerli, from the Verus CLI or GUI on the vrsctest chain, or from CLI/GUI on other PBaaS chains to provide liquidity while holding Bridge.vETH, a 100% backed basket of three currencies of balanced relative value, or to pass through and convert currencies between Verus, Ethereum, USDC, and Bridge.vETH.

The Brige.vETH currency also coordinates with the vETH@ gateway and enables registering `*.veth@` IDs for \$5 worth of Bridge.vETH, based on its internal USDC conversion price at the time of registration. When you register `*.veth@` IDs, 0.02 VRSCTEST goes to miners and stakers, and \$5 worth of Bridge.vETH is burned, reducing the total supply of Bridge.vETH for everyone without reducing reserves, and contributing to the relative value, "real yield" of Bridge.vETH, relative to its three reserve currencies. This is the way that all decentralized, fractional currencies which enable ID registration accept registration fees, equally benefiting all fractional currency holders/liquidity providers when IDs are registered, while providing Verus miners and stakers the identity import fee of 0.02 VRSC.

* .veth@ IDs have the ability, along with root vrsctest IDs, to make 1:1 ERC20 mapped currencies on Ethereum. They can also be used to create Tokenized ID control tokens (see notes for v0.9.4-1), as all types of IDs can, but if they are used to do that, they will have used their currency creation ability for that rather than an ERC20 mapped currency.

The rest of these release notes are the same as the prior release. Drop by #pbaas-development to start building your vision on Verus and help prepare PBaaS for mainnet!

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc`, but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or `z-address`

```
# verus -chain=vrsc test registeridentity '{ID registration with name
commitment}' returntx feoffer sourceoffunds

verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "name": "name", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": "", "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6Wxv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they

can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your busi...

Assets 6

v0.9.4-1

 [Asherda](#)

[v0.9.4-1](#)

[7f145d5](#)

[v0.9.4-1](#)

Announcing v0.9.4-1 with the world's first self-sovereign, tokenized ID control technology,
RECOMMENDED FOR MAINNET -- MANDATORY FOR TESTNET USE

This release has no material mainnet changes.

For testnet, this is RC3 of the PBaaS protocol and is required to follow the main testnet chain. RC3 has some significant and important API additions, including the ability for any ID at all, even sub-IDs, to generate a single token as its currency. That token then enables whoever holds it to have another level of revoke/recover power over the ID, in addition to the normal revocation and recovery authorities, which also still work. This technology is now available in the pre-existing APIs, updateidentity (for changing revocation / recovery authorities), revokeidentity, and recoveryidentity. Please see help for details.

In addition, we have continued to harden the protocol and fixed issues with cross-chain currency definition import/export and the new ability to burn reserves into a liquidity basket that were reported in the current testnet.

The rest of these release notes are the same as the prior release. Drop by pbaas-development to start building your vision on Verus while helping get PBaaS ready for mainnet!

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID", "options":2080, "preallocations":
[{"ControlTokenRecipient@":0.00000001}], "maxpreconversion": [0]}'
```

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc`, but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsc test registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsc test registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsc test registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@{}"}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again

unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":100,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains

with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while loc...

Assets 6

1 person reacted

v0.9.4

 [Asherda](#)

[v0.9.4](#)

[e2a52aa](#)

[v0.9.4](#)

Announcing v0.9.4, HIGHLY RECOMMENDED FOR MAINNET -- MANDATORY FOR TESTNET USE

This release fixes a mainnet issue that may, in rare cases, cause a node to deadlock.

It also adds the following new features to mainnet:

- Auto TLS encrypted connections between nodes on mainnet and testnet, no action needed, but you can require that all connections be encrypted by starting the daemon with “-tlsencforcement”.
- A new “-idindex” flag that enables 3 new APIs to lookup identities by their primary addresses or revocation and recovery addresses, `getidentitywithaddress`, `getidentitywithrevocation`, and `getidentitywithrecovery`. Generally improved multithreading within the daemon.
- These improvements allow anyone to now write applications to easily graph or display public relationships between identities, transparent addresses, and transactions without compromising the privacy of private addresses, messages, or transactions between identities using name@:private endpoints, z-addresses or communicating via off-chain channels.

For testnet, this is RC2 of the PBaaS protocol and has some significant and important advances and changes. In addition to a great deal of progress hardening for mainnet, v0.9.4 introduces the following new features to the PBaaS testnet:

- All features above for mainnet
- NFT mapped tokens and tokenized ID control - In v0.9.4, as with v0.9.3, you can use root identities to create any kind of currencies, *.veth IDs to create Ethereum mapped currencies, and what is new in v0.9.4 is that you can now use any type of ID or sub-ID to create a new kind of currency that will always have a 1 Satoshi supply and enable anyone holding it to have power over the ID that issued it. This type of currency is called tokenized ID control and can be treated like a super NFT over all copies of the source ID on all chains that were exported to other chains after the currency was created. Once the Ethereum contracts are upgraded with this capability, *.veth or IDs with root names on the Verus blockchain can be used to create Ethereum NFT mapped currencies, which enable a Cryptopunk, Ethrock, or whatever other NFT you’d like to not only be sent to Verus, held on a revocable/recoverable ID, and also permanently upleveled with ID powers, being cryptographically bound as an ID’s permanent control token that can be sent to any chain or to Ethereum and back.

This unique Verus-only technology and VerusIDs enable new models of complex, atomic and fully decentralized transactions. By using exported IDs across any number of networks and an ID control token, ownership of an unlimited array of assets across an unlimited number of networks can for the first time ever, have ownership transferred in one transaction.

- While NFT mapped tokens are supported on the RC2 PBaaS protocol, they have not yet been implemented in the Ethereum contracts. You will not be able to send NFTs from Ethereum to Verus or PBaaS chains until that work is complete, which we do not expect to take a great deal of time.

- Liquidity currencies now support new methods to burn currency with the following results:
 - **burnweight** - available to centralized currencies only and enables burning the primary currency to raise the reserve ratio weights of all reserves.
 - **burn** for reserve currencies - in prior releases, burn was only used for primary currencies and would reduce the supply of a currency relative to its reserves, increasing the on-chain value of the primary currency. In v0.9.4, you can now also “burn” reserve currencies in a liquidity basket currency, which simply donates that amount of reserve into the currency’s reserves without taking any new supply in exchange. This is available to decentralized or centralized currencies.
- Bridgekeeper is now available for permissionless bridge notarization and selectable in the GUI on the mining page. For CLI, we will make a Node application available for testing in the next few days.
- RC1 cross-chain witness and notarization protocol enhanced in RC2 as follows:
 - In RC2, any notarization to be confirmed must be first posted publicly to allow for potential to prove invalidity, and then later confirmed. In the prior protocol, a witnessed notarization could be posted to a chain and accepted when first seen. Upon analysis and discussion with [@allbits](#) and others, we realized that the attack vector of even stolen notary/witness keys combined with an in-secret 51% combined hash/stake attack could be prevented if we changed the protocol to first require posting the ready to finalize data and allow notaries who recognize that there is a signature on chain for something they don’t agree with (likely stolen key), to simply revoke their identity, which will prevent that notarization from ever being confirmed. This same security feature is supported between all PBaaS chains and has been implemented in the Ethereum contracts for witnesses as well. This new protocol also provides for other types of evidence to prevent finalization, such as proof that there exists a more powerful (most power rule) tip than one witnessed. Enabling decentralized, cross-chain most power rules are an advance in cross-chain protocols that over time will strongly protect censorship resistance qualities of any multichain protocol.
 - The above technology required changes to generalize evidence types between chains, enabling cryptographic enforcement that all notarizations are ultimately determined by miners and stakers and finalized by witnesses who must follow most power (stake+work) rules based on the protocol. Witnesses cannot finalize a chain tip unless alternating miners and stakers have posted 3 consecutive, agreeing, progressing cross-chain notarizations, without intervening disagreement, trading off more certain cross-chain transaction latency for on that depends on miners and stakers, meaning maximum decentralization and a high cost to DoS attack cross-chain, approaching impossible over any significant length of time. This approach makes the incentivized merge mining and bridge mining opportunities, as well as miner/staker participation, automatically part of and also critical to the cross-chain protocol. In v0.9.4, all users on the network may mine the Ethereum bridge via the bridgekeeper tool on the mining page in the GUI.
 - With this new protocol, all PBaaS chains on testnet will consider witnessed / finalized tips final and will not unwind past those witness points or follow a chain

that disagrees with a known finalized tip. When this protocol goes live on mainnet, and for any PBaaS chain whose community opts to use the Komodo notary service, Komodo notarizations will continue to ensure a stable chain tip, but no matter what other notarizations are present, PBaaS chains will always follow their proof of power protocol and lock to their witnessed tips as well.

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

The rest of these release notes are the same as the prior release. Drop by pbaas-development to start building your vision on Verus while helping get PBaaS ready for mainnet!

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc`, but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feoffer sourceoffunds

verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Curren...

Contributors



allbits
Assets 6

v0.9.3

 [Asherda](#)
[v0.9.3](#)
[9e8227b](#)
[v0.9.3](#)

Announcing v0.9.3, NON-MANDATORY FOR MAINNET -- MANDATORY FOR TESTNET USE

This release fixes one issue on mainnet where a node using `-connect="networkaddress"` may still attempt to connect to other nodes.

This update includes the first protocol release candidate for mainnet PBaaS, VerusID protocol upgrades, and Verus DeFi. In addition to some functional fixes and full web site support for mapped currencies and simple commands for mapping Verus currencies to any ERC20 currencies on Ethereum with provable, auditable 1:1 mapping and cross chain send/receive. In addition this new protocol upgrade changes some parameters of the Verus Proof of Power consensus algorithm, enabling smoother starts, restarts, and general operation of PBaaS chains, regardless of their native

currency supply or emission schedule. As these consensus parameter changes would require either a testnet fork, which would not be relevant for the mainnet upgrade or a reset, accessing the v0.9.3 testnet requires that you reset your local testnet by deleting the vrsctest and pbaas data folders from the locations described below.

As part of this release, we will be launching the new Ethereum bridge and enable full use of Ethereum mapped currencies on the Goerli Ethereum testnet (note that we were previously using Rinkeby). This makes it easy for any currency definition to become a fully decentralized, auditable stable coin, simply by mapping to DAI, USDT, USDC, or another stable coin on Ethereum. In addition, the new website for Metamask support and use from Ethereum now displays all mapped currency options, even those added when someone exports a currency definition to Ethereum, either as a mapped currency or for using any Verus or PBaaS currency as an ERC20.

This release also resolves an issue with setting max preconversion amounts on currency launches and addresses a number of minor protocol issues found as a result of the active community testing support we are seeing on testnet.

v0.9.3 RESETS TESTNET AND ADDRESSES ALL KNOWN ISSUES FOUND DURING TESTING ON THE NETWORK. FROM HERE, WE HOPE ALL COMMUNITY MEMBERS WILL JOIN IN TRYING OUT ALL FEATURES OF TESTNET TO HELP ENSURE THAT THE FINAL MAINNET RELEASE, WHICH WE ARE WORKING TOWARDS AS SOON AS IT IS CONSIDERED READY AND HARDENED, WILL BE BOTH SMOOTH AND WORLD-CHANGING.

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

Platform Support

- Debian 9 (Stretch) is no longer supported

PROTOCOL RELEASE CANDIDATE RC1

With v0.9.3, The Verus PBaaS protocol, including networking, public blockchains as a service (PBaaS), VerusID extensions, and Verus DeFi, is considered release candidate RC1 for the coming mainnet upgrade. The core developer focus now remains on protocol hardening and enabling usage first, followed by user experience upgrades and supporting infrastructure such as identity and login support for lite mode, mobile and web extension wallets as well as merge mining for pool servers.

NEW PBAAS CAPABILITIES

All PBaaS capabilities, including for application development, are available now on the new Verus testnet. Now would be a good time for development capable projects hoping to get a head start on using Verus's transformative technologies to do a deep dive on the Verus testnet. These new capabilities in v0.9.3, as well as all those we are familiar with, available nowhere else, will be on mainnet SOON^(TM):

All fractional and centralized {"proofprotocol":2} currencies can now issue as a decentralized operation, or with centralized currencies, sell, IDs using their own currency as the payment currency, with options for price, referrals, permission from the currency identity required, which means the transaction must be created or co-signed by the currency identity, or even referrals required, which means that an identity registration must have a referral and co-signature of a referring ID that is either the currency ID or another ID from that currency namespace.

For fully decentralized fractional currencies, ID registration is paid for by burning the specific fractional currency, reducing the supply, and raising the on-chain value of the primary currency relative to its reserves by reducing the current on-chain supply for everyone.

The minimum price of ID registration for fractional currencies can be set in either the fractional currency itself, or in one of its reserves, for example USDC or DAI.

More than the minimum price can be paid for ID registration, which may also include VDXF key and contract bindings to allow for multiple types of content or IP licensing in a single ID namespace.

Each currency, either centralized and/or fractional, can have multiple levels of referral support (limit of 5 levels), meaning that the ID registration fee will be reduced when referred and part of it will be divided evenly among past referrers. The remaining amount of the ID registration fee will be burned for decentralized currencies, or be sent to the currency ID for centralized currencies.

While Native IDs, those registered directly from a blockchain as the name parent, will be able to launch currencies on that blockchain, and if on Verus, actual independent blockchains, IDs issued by fractional or centralized currencies cannot launch currencies, but will still be valuable as revocation and recovery IDs for all other VerusIDs as well as for personalized on-chain and cross-chain addresses with all identity and NFT features, for proofs, for brands, contract binding, on-chain marketplace trading, or login, and will have the ability to cross-prove common ownership with other IDs as well.

The exception to non-native IDs creating currencies, is that *.veth gateway IDs will be available for decentralized registration from the Ethereum bridge converter and will be able to create Ethereum mapped currencies. Registration of .veth IDs will be priced at 100 USD worth of ETH to prevent misuse and support bridge liquidity, with all fees besides the 1 VRSC import fee, which goes into the mining and staking fee pool going to the bridge converter liquidity basket.

All of these features are currently enabled on the v0.9.3 new testnet. Stay connected on the #pbaas-development channel and participate in the launch at no actual cost as a testnet liquidity provider and get some nice testnet IDs, launch your own currencies, or start your exclusive ID-restricted club, governance group, or business.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`

- Windows 10: %AppData%\Roaming\Komodo\vrsc\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsc\, %AppData%\Roaming\VerusTest

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes "" to leave fields blank, the example below specifies a parent currency, vrsc-btc , but no referrer. We're now able to use z_addresses to fund the name commitment and identity registration

```
# verus -chain=vrsc test registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsc test registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsc test registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzL", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referrer": ""}, "identity": {"primaryaddresses": ["RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute

transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next ...

Assets 6

v0.9.2-3

 [Asherda](#)
[v0.9.2-3](#)
[afc6987](#)
[v0.9.2-3](#)

Announcing v0.9.2-3, NON-MANDATORY, HIGHLY RECOMMENDED FOR MAINNET -- CRITICAL FOR TESTNET USE

This update enables full use of Ethereum mapped currencies (Rinkeby when the new bridge launches), which provides the ability for any currency to become a fully decentralized, auditible stable coin, simply by mapping to DAI, USDT, USDC, or another on Ethereum. In addition, the new website for Metamask support and use from Ethereum now displays all mapped currency options, even those added when someone exports a currency definition to Ethereum, either as a mapped currency or for using any Verus or PBaaS currency as an ERC20.

This release also resolves two PBaaS chain launch issues, one that prevented no-reward chains with initial emission from being accepted and one that could allow a chain launch to go forward, even when a co-launched liquidity basket failed to get enough participation. Thanks to increased participation on testnet use, we are getting continued excellent coverage and resolving issues, which are increasingly edge conditions as we go. discovered on both mainnet and testnet and on testnet fixes an error creating transactions for sub identity registrations on fractional reserve currencies. If you already went through the testnet reset process on any version since v0.9.2, you do not need to reset testnet. The rest of these release notes are the same as the prior version.

v0.9.2 RESETS TESTNET AND ADDS A COMPLETE SET OF NEW CAPABILITIES PREVIOUSLY CONSIDERED STRETCH GOALS THAT ENABLE BRANDED SUB-ID ISSUANCE WITH BOTH DECENTRALIZED AND CENTRALIZED OPTIONS AS WELL AS THE FIRST DISRUPTIVE TECHNOLOGY TO SOLVE FOR BRANDED LOYALTY IDENTITIES WITH NOVEL SOLUTIONS FOR ROYALTY AND IP LICENSING BUSINESS MODELS.

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

FOR MAINNET (AND TESTNET), v0.9.2 PROVIDES ENHANCED PRIVACY FEATURES FOR IDENTITY CREATION

PROTOCOL FEATURE COMPLETE

With v0.9.2, The Verus network, public blockchains as a service (PBaaS), VerusID extensions, and Verus DeFi protocols are considered feature complete for the coming mainnet PBaaS upgrade. The core developer focus now is protocol hardening and enabling usage first, followed by user experience upgrades and supporting infrastructure such as identity and login support for lite mode, mobile and web extension wallets as well as merge mining for pool servers.

NEW 3D CROSS-CHAIN LIQUIDITY NETWORK VISUALIZER (GUI FOR TESTNET)

In v0.9.2 Verus Desktop, there is a new 3-dimensional cross-chain network visualizer, which shows all currency conversion and fractional network connections both on chain and cross-chain, with the ability to move around and zoom in and out in 3 dimensions. To access the visualizer, click on “Visualize Network” on the Metaverse tab of Verus Desktop. If you’re interested in seeing the visualizer now, here’s a link directly to the visualizer as part of a recent video members of the community helped make (https://youtu.be/CkgUVjx_g1M?t=43).

NEW PBAAS CAPABILITIES

All PBaaS capabilities, including for application development, are available now on the new Verus testnet. Our goal for interoperability on this new testnet is to launch the Ethereum/Rinkeby bridge in the next couple days. Now would be a good time for development capable projects hoping to get a head start on using Verus’s transformative technologies to do a deep dive on the Verus testnet. These new capabilities in v0.9.2, as well as all those we are familiar with, available nowhere else, will be on mainnet SOON(™):

All fractional and centralized {"proofprotocol":2} currencies can now issue as a decentralized operation, or with centralized currencies, sell, IDs using their own currency as the payment currency, with options for price, referrals, permission from the currency identity required, which means the transaction must be created or co-signed by the currency identity, or even referrals required, which means that an identity registration must have a referral and co-signature of a referring ID that is either the currency ID or another ID from that currency namespace.

For fully decentralized fractional currencies, ID registration is paid for by burning the specific fractional currency, reducing the supply, and raising the on-chain value of the primary currency relative to its reserves by reducing the current on-chain supply for everyone.

The minimum price of ID registration for fractional currencies can be set in either the fractional currency itself, or in one of its reserves, for example USDC or DAI.

More than the minimum price can be paid for ID registration, which may also include VDXF key and contract bindings to allow for multiple types of content or IP licensing in a single ID namespace.

Each currency, either centralized and/or fractional, can have multiple levels of referral support (limit of 5 levels), meaning that the ID registration fee will be reduced when referred and part of it

will be divided evenly among past referrers. The remaining amount of the ID registration fee will be burned for decentralized currencies, or be sent to the currency ID for centralized currencies.

While Native IDs, those registered directly from a blockchain as the name parent, will be able to launch currencies on that blockchain, and if on Verus, actual independent blockchains, IDs issued by fractional or centralized currencies cannot launch currencies, but will still be valuable as revocation and recovery IDs for all other VerusIDs as well as for personalized on-chain and cross-chain addresses with all identity and NFT features, for proofs, for brands, contract binding, on-chain marketplace trading, or login, and will have the ability to cross-prove common ownership with other IDs as well.

The exception to non-native IDs creating currencies, is that *.veth gateway IDs will be available for decentralized registration from the Ethereum bridge converter and will be able to create Ethereum mapped currencies. Registration of .veth IDs will be priced at a level to prevent misuse and support bridge liquidity, with all fees besides the 1 VRSC import fee, which goes into the mining and staking fee pool going to the bridge converter liquidity basket.

All of these features are currently enabled on the v0.9.2 new testnet. Stay connected on the #pbaas-development channel and participate in the launch at no actual cost as a testnet liquidity provider and get some nice testnet IDs, launch your own currencies, or start your exclusive ID-restricted club, governance group, or business.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc` , but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsc test registeridentity '{ID registration with name commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": ["RDnf7mH7RQki9b7PqdBD2Er6Wxv3DTawGr"], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value a...

Assets 6

v0.9.2-2

 [Asherda](#)
[v0.9.2-2](#)
[79d6171](#)
[v0.9.2-2](#)

Announcing v0.9.2-2, NON-MANDATORY, HIGHLY RECOMMENDED FOR MAINNET -- CRITICAL FOR TESTNET USE

This update improves sync speed in some edge cases on both mainnet and testnet and on testnet fixes an error creating transactions for sub identity registrations on fractional reserve currencies. If you already went through the testnet reset process on any version since v0.9.2, you do not need to reset testnet. The rest of these release notes are the same as the prior version.

v0.9.2 RESETS TESTNET AND ADDS A COMPLETE SET OF NEW CAPABILITIES PREVIOUSLY CONSIDERED STRETCH GOALS THAT ENABLE BRANDED SUB-ID ISSUANCE WITH BOTH DECENTRALIZED AND CENTRALIZED OPTIONS AS WELL AS THE FIRST DISRUPTIVE TECHNOLOGY TO SOLVE FOR BRANDED LOYALTY IDENTITIES WITH NOVEL SOLUTIONS FOR ROYALTY AND IP LICENSING BUSINESS MODELS.

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

FOR MAINNET (AND TESTNET), v0.9.2 PROVIDES ENHANCED PRIVACY FEATURES FOR IDENTITY CREATION

PROTOCOL FEATURE COMPLETE

With v0.9.2, The Verus network, public blockchains as a service (PBaaS), VerusID extensions, and Verus DeFi protocols are considered feature complete for the coming mainnet PBaaS upgrade. The core developer focus now is protocol hardening and enabling usage first, followed by user experience upgrades and supporting infrastructure such as identity and login support for lite mode, mobile and web extension wallets as well as merge mining for pool servers.

NEW 3D CROSS-CHAIN LIQUIDITY NETWORK VISUALIZER (GUI FOR TESTNET)

In v0.9.2 Verus Desktop, there is a new 3-dimensional cross-chain network visualizer, which shows all currency conversion and fractional network connections both on chain and cross-chain, with the ability to move around and zoom in and out in 3 dimensions. To access the visualizer, click on “Visualize Network” on the Metaverse tab of Verus Desktop. If you’re interested in seeing the visualizer now, here’s a link directly to the visualizer as part of a recent video members of the community helped make (https://youtu.be/CkgUVjx_g1M?t=43).

NEW PBAAS CAPABILITIES

All PBaaS capabilities, including for application development, are available now on the new Verus testnet. Our goal for interoperability on this new testnet is to launch the Ethereum/Rinkeby bridge in the next couple days. Now would be a good time for development capable projects hoping to get a

head start on using Verus's transformative technologies to do a deep dive on the Verus testnet. These new capabilities in v0.9.2, as well as all those we are familiar with, available nowhere else, will be on mainnet SOON(™):

All fractional and centralized {"proofprotocol":2} currencies can now issue as a decentralized operation, or with centralized currencies, sell, IDs using their own currency as the payment currency, with options for price, referrals, permission from the currency identity required, which means the transaction must be created or co-signed by the currency identity, or even referrals required, which means that an identity registration must have a referral and co-signature of a referring ID that is either the currency ID or another ID from that currency namespace.

For fully decentralized fractional currencies, ID registration is paid for by burning the specific fractional currency, reducing the supply, and raising the on-chain value of the primary currency relative to its reserves by reducing the current on-chain supply for everyone.

The minimum price of ID registration for fractional currencies can be set in either the fractional currency itself, or in one of its reserves, for example USDC or DAI.

More than the minimum price can be paid for ID registration, which may also include VDXF key and contract bindings to allow for multiple types of content or IP licensing in a single ID namespace.

Each currency, either centralized and/or fractional, can have multiple levels of referral support (limit of 5 levels), meaning that the ID registration fee will be reduced when referred and part of it will be divided evenly among past referrers. The remaining amount of the ID registration fee will be burned for decentralized currencies, or be sent to the currency ID for centralized currencies.

While Native IDs, those registered directly from a blockchain as the name parent, will be able to launch currencies on that blockchain, and if on Verus, actual independent blockchains, IDs issued by fractional or centralized currencies cannot launch currencies, but will still be valuable as revocation and recovery IDs for all other VerusIDs as well as for personalized on-chain and cross-chain addresses with all identity and NFT features, for proofs, for brands, contract binding, on-chain marketplace trading, or login, and will have the ability to cross-prove common ownership with other IDs as well.

The exception to non-native IDs creating currencies, is that *.veth gateway IDs will be available for decentralized registration from the Ethereum bridge converter and will be able to create Ethereum mapped currencies. Registration of .veth IDs will be priced at a level to prevent misuse and support bridge liquidity, with all fees besides the 1 VRSC import fee, which goes into the mining and staking fee pool going to the bridge converter liquidity basket.

All of these features are currently enabled on the v0.9.2 new testnet. Stay connected on the #pbaas-development channel and participate in the launch at no actual cost as a testnet liquidity provider and get some nice testnet IDs, launch your own currencies, or start your exclusive ID-restricted club, governance group, or business.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: ~/Komodo/vrsctest, ~/verustest

- Mac OS: ~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest
- Windows 10: %AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes "" to leave fields blank, the example below specifies a parent currency, vrsc-btc , but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@{}"}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which ...

Assets 6

v0.9.2-1

 [Asherda](#)

[v0.9.2-1](#)

[34c2fb5](#)

[v0.9.2-1](#)

Announcing v0.9.2-1, NO FUNCTIONAL CHANGES TO MAINNET FROM v0.9.2 -- CRITICAL FOR TESTNET USE

This update fixes an error in creating transactions for sub identity registrations on fractional reserve currencies and it fixes a synchronization issue that could prevent some clients from fully syncing testnet. The rest of these release notes are the same as the prior version.

v0.9.2 RESETS TESTNET AND ADDS A COMPLETE SET OF NEW CAPABILITIES PREVIOUSLY CONSIDERED STRETCH GOALS THAT ENABLE BRANDED SUB-ID ISSUANCE WITH BOTH DECENTRALIZED AND CENTRALIZED OPTIONS AS WELL AS THE FIRST DISRUPTIVE TECHNOLOGY TO SOLVE FOR BRANDED LOYALTY

IDENTITIES WITH NOVEL SOLUTIONS FOR ROYALTY AND IP LICENSING BUSINESS MODELS.

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

FOR MAINNET (AND TESTNET), v0.9.2 PROVIDES ENHANCED PRIVACY FEATURES FOR IDENTITY CREATION

PROTOCOL FEATURE COMPLETE

With v0.9.2, The Verus network, public blockchains as a service (PBaaS), VerusID extensions, and Verus DeFi protocols are considered feature complete for the coming mainnet PBaaS upgrade. The core developer focus now is protocol hardening and enabling usage first, followed by user experience upgrades and supporting infrastructure such as identity and login support for lite mode, mobile and web extension wallets as well as merge mining for pool servers.

NEW 3D CROSS-CHAIN LIQUIDITY NETWORK VISUALIZER (GUI FOR TESTNET)

In v0.9.2 Verus Desktop, there is a new 3-dimensional cross-chain network visualizer, which shows all currency conversion and fractional network connections both on chain and cross-chain, with the ability to move around and zoom in and out in 3 dimensions. To access the visualizer, click on “Visualize Network” on the Metaverse tab of Verus Desktop. If you’re interested in seeing the visualizer now, here’s a link directly to the visualizer as part of a recent video members of the community helped make (https://youtu.be/CkgUVjx_g1M?t=43).

NEW PBAAS CAPABILITIES

All PBaaS capabilities, including for application development, are available now on the new Verus testnet. Our goal for interoperability on this new testnet is to launch the Ethereum/Rinkeby bridge in the next couple days. Now would be a good time for development capable projects hoping to get a head start on using Verus’s transformative technologies to do a deep dive on the Verus testnet. These new capabilities in v0.9.2, as well as all those we are familiar with, available nowhere else, will be on mainnet SOON(™):

All fractional and centralized {"proofprotocol":2} currencies can now issue as a decentralized operation, or with centralized currencies, sell, IDs using their own currency as the payment currency, with options for price, referrals, permission from the currency identity required, which means the transaction must be created or co-signed by the currency identity, or even referrals required, which means that an identity registration must have a referral and co-signature of a referring ID that is either the currency ID or another ID from that currency namespace.

For fully decentralized fractional currencies, ID registration is paid for by burning the specific fractional currency, reducing the supply, and raising the on-chain value of the primary currency relative to its reserves by reducing the current on-chain supply for everyone.

The minimum price of ID registration for fractional currencies can be set in either the fractional currency itself, or in one of its reserves, for example USDC or DAI.

More than the minimum price can be paid for ID registration, which may also include VDXF key and contract bindings to allow for multiple types of content or IP licensing in a single ID namespace.

Each currency, either centralized and/or fractional, can have multiple levels of referral support (limit of 5 levels), meaning that the ID registration fee will be reduced when referred and part of it will be divided evenly among past referrers. The remaining amount of the ID registration fee will be burned for decentralized currencies, or be sent to the currency ID for centralized currencies.

While Native IDs, those registered directly from a blockchain as the name parent, will be able to launch currencies on that blockchain, and if on Verus, actual independent blockchains, IDs issued by fractional or centralized currencies cannot launch currencies, but will still be valuable as revocation and recovery IDs for all other VerusIDs as well as for personalized on-chain and cross-chain addresses with all identity and NFT features, for proofs, for brands, contract binding, on-chain marketplace trading, or login, and will have the ability to cross-prove common ownership with other IDs as well.

The exception to non-native IDs creating currencies, is that *.veth gateway IDs will be available for decentralized registration from the Ethereum bridge converter and will be able to create Ethereum mapped currencies. Registration of .veth IDs will be priced at a level to prevent misuse and support bridge liquidity, with all fees besides the 1 VRSC import fee, which goes into the mining and staking fee pool going to the bridge converter liquidity basket.

All of these features are currently enabled on the v0.9.2 new testnet. Stay connected on the #pbaas-development channel and participate in the launch at no actual cost as a testnet liquidity provider and get some nice testnet IDs, launch your own currencies, or start your exclusive ID-restricted club, governance group, or business.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a

parent currency, vrsc-btc , but no referrer. We're now able to use z_addresses to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr """
vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@{}"}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary...

Assets 6

v0.9.2



[Asherda](#)
[v0.9.2](#)
[5c91a86](#)
[v0.9.2](#)

Announcing v0.9.2, OPTIONAL, HIGHLY RECOMMENDED FOR MAINNET, MANDATORY FOR TESTNET USE - THIS IS A BIG ANNOUNCE, BUT IF YOU CARE ABOUT THE PBAAS UPGRADE, WE RECOMMEND READING IT ALL.

v0.9.2 RESETS TESTNET AND ADDS A COMPLETE SET OF NEW CAPABILITIES PREVIOUSLY CONSIDERED STRETCH GOALS THAT ENABLE BRANDED SUB-ID ISSUANCE WITH BOTH DECENTRALIZED AND CENTRALIZED OPTIONS AS WELL AS THE FIRST DISRUPTIVE TECHNOLOGY TO SOLVE FOR BRANDED LOYALTY IDENTITIES WITH NOVEL SOLUTIONS FOR ROYALTY AND IP LICENSING BUSINESS MODELS.

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

FOR MAINNET (AND TESTNET), v0.9.2 PROVIDES ENHANCED PRIVACY FEATURES FOR IDENTITY CREATION

PROTOCOL FEATURE COMPLETE

With v0.9.2, The Verus network, public blockchains as a service (PBaaS), VerusID extensions, and Verus DeFi protocols are considered feature complete for the coming mainnet PBaaS upgrade. The

core developer focus now is protocol hardening and enabling usage first, followed by user experience upgrades and supporting infrastructure such as identity and login support for lite mode, mobile and web extension wallets as well as merge mining for pool servers.

NEW 3D CROSS-CHAIN LIQUIDITY NETWORK VISUALIZER (GUI FOR TESTNET)

In v0.9.2 Verus Desktop, there is a new 3-dimensional cross-chain network visualizer, which shows all currency conversion and fractional network connections both on chain and cross-chain, with the ability to move around and zoom in and out in 3 dimensions. To access the visualizer, click on “Visualize Network” on the Metaverse tab of Verus Desktop. If you’re interested in seeing the visualizer now, here’s a link directly to the visualizer as part of a recent video members of the community helped make (https://youtu.be/CkgUVjx_g1M?t=43).

NEW PBAAS CAPABILITIES

All PBaaS capabilities, including for application development, are available now on the new Verus testnet. Our goal for interoperability on this new testnet is to launch the Ethereum/Rinkeby bridge in the next couple days. Now would be a good time for development capable projects hoping to get a head start on using Verus’s transformative technologies to do a deep dive on the Verus testnet. These new capabilities in v0.9.2, as well as all those we are familiar with, available nowhere else, will be on mainnet SOON(™):

All fractional and centralized {"proofprotocol":2} currencies can now issue as a decentralized operation, or with centralized currencies, sell, IDs using their own currency as the payment currency, with options for price, referrals, permission from the currency identity required, which means the transaction must be created or co-signed by the currency identity, or even referrals required, which means that an identity registration must have a referral and co-signature of a referring ID that is either the currency ID or another ID from that currency namespace.

For fully decentralized fractional currencies, ID registration is paid for by burning the specific fractional currency, reducing the supply, and raising the on-chain value of the primary currency relative to its reserves by reducing the current on-chain supply for everyone.

The minimum price of ID registration for fractional currencies can be set in either the fractional currency itself, or in one of its reserves, for example USDC or DAI.

More than the minimum price can be paid for ID registration, which may also include VDXF key and contract bindings to allow for multiple types of content or IP licensing in a single ID namespace.

Each currency, either centralized and/or fractional, can have multiple levels of referral support (limit of 5 levels), meaning that the ID registration fee will be reduced when referred and part of it will be divided evenly among past referrers. The remaining amount of the ID registration fee will be burned for decentralized currencies, or be sent to the currency ID for centralized currencies.

While Native IDs, those registered directly from a blockchain as the name parent, will be able to launch currencies on that blockchain, and if on Verus, actual independent blockchains, IDs issued by fractional or centralized currencies cannot launch currencies, but will still be valuable as revocation and recovery IDs for all other VerusIDs as well as for personalized on-chain and cross-chain addresses with all identity and NFT features, for proofs, for brands, contract binding, on-chain

marketplace trading, or login, and will have the ability to cross-prove common ownership with other IDs as well.

The exception to non-native IDs creating currencies, is that *.veth gateway IDs will be available for decentralized registration from the Ethereum bridge converter and will be able to create Ethereum mapped currencies. Registration of .veth IDs will be priced at a level to prevent misuse and support bridge liquidity, with all fees besides the 1 VRSC import fee, which goes into the mining and staking fee pool going to the bridge converter liquidity basket.

All of these features are currently enabled on the v0.9.2 new testnet. Stay connected on the #pbaas-development channel and participate in the launch at no actual cost as a testnet liquidity provider and get some nice testnet IDs, launch your own currencies, or start your exclusive ID-restricted club, governance group, or business.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes "" to leave fields blank, the example below specifies a parent currency, vrsc-btc , but no referrer. We're now able to use z_addresses to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqkxzcata02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds
verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservat
```

```
ion": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": ["RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0  
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfn
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to who...

v0.9.1-2

 [Asherda](#)

[v0.9.1-2](#)

[6c5c68b](#)

[v0.9.1-2](#)

ANNOUNCING PBAAS TESTNET UPGRADE v0.9.1-2, OPTIONAL FOR MAINNET USERS -- MANDATORY FOR TESTNET USERS

v0.9.1-2 RESETS TESTNET -- THERE ARE NO MAINNET CHANGES

IF YOU HAVE ALREADY USED TESTNET WITH VERSIONS PRIOR TO v0.9.1-2, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED BELOW.

Testnet Featuring Verus PBaaS

The most powerful and interoperable currency, blockchain, and ERC20 launch, scale, and MEV-resistant DeFi platform to exist, all with no programming required!

Verus PBaaS Features Live on v0.9.1-2 Testnet

- Provable or pseudonymous, identity-based currency, liquidity pool, NFT and multi-blockchain, fully decentralized network.
- On-chain 100% decentralized, launches with fair launch and/or crowdfunding options, on-chain auto-refund option for missed targets, automatic Ethereum ERC20 contract deployment, auto-created, MEV-resistant, liquidity baskets and even auto-bridged, merge mineable, fully independent rent-free PBaaS blockchain launches.
- All DeFi AMM conversions are verified via mining and staking as part of the L1 consensus protocol. All conversions in a single liquidity basket are calculated simultaneously for all transactions in any given block, meaning all participants get the same price in all directions of conversion, with a minimum conversion fee of 0.025% and a maximum of 0.05%.
- All currencies and identities are primitives at L1 and are validated and verified on UTXO transactions, which check all inputs and outputs just as single currency L1, such as Ethereum or Bitcoin check the native currency inputs and outputs of transactions.
- All currencies on all independent connected chains, once launched can be sent back and forth to other multi-currency capable networks (currently other independent PBaaS chains and Ethereum as ERC20s).

- Currencies on Verus can also be defined as "mapped currencies", which map 1:1 to an existing currency on Ethereum (eg. DAI), and can then be sent from Ethereum with simple transactions and received and used as the new "mapped currency", or sent back to Ethereum and used there as the original currency.
- Currency launches can raise funds in 3 ways. Each of these fundraising options creates a currency that is not 100% backed, and has a price that responds to market forces. 100% backed currencies do not change their relative price to the underlying basket reserve when they are converted to or from reserves. Fractionally backed currencies, which result from fundraising or endowing launch grants/pre-allocations to DAOs or entities such as foundations, whether time-locked or not, take their funding from a percentage of the reserve backing. Except in the case of a new blockchain launch, which creates new native currency for operating the proof of stake of the blockchain, any currency that uses fundraising options will be a fractionally backed currency which will appreciate when more people convert to it and depreciate when more convert from it. If a launch is refunded due to minimum participation options that are not met, no fundraising is received:
 - Pre-launch discount - all participants contributing to the launch before it goes live on its start block get a discount to the initial on-chain price of the currency. This is usually accompanied by a maximum cap on pre-conversion participation to ensure that the currency would still be in demand when it goes live.
 - Pre-launch carve-out - this enables a percentage of each of the initial participation currencies to be carved out of the launch and total reserve ratio in a price neutral manner and sent directly to the launching identity upon successful launch.
 - Pre-allocation - this enables some of the newly launched currency to be pre-allocated to one or more identities. If the currency is fractional, this happens in a price neutral manner, and again is subtracted from the reserve ratio. Recipients can optionally be time locked IDs for vesting or unlock periods.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.Komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming

- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency ***
' [{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 100, "currency": "vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":"<destination_address>","amount":<transaction_amount>}]]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by `z_getoperationstatus` to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which t...

Assets 6

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
- [Privacy](#)
- [Security](#)
- [Status](#)
- [Docs](#)

- Contact GitHub
 - [Pricing](#)
 - [API](#)
 - [Training](#)
 - [Blog](#)
 - [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

- Code

- [Issues 21](#)
- [Pull requests](#)
- [Actions](#)
- [Projects](#)
- [Wiki](#)
- [Security](#)
- [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v0.9.9-1



[v0.9.9-1](#)
[08a713f](#)
[v0.9.9-1](#)

Announcing v0.9.9-1, MANDATORY FOR TESTNET, no mainnet changes

This release implements a fix for communication with BridgeKeeper for the Testnet Goerli Ethereum Bridge, as well as edge-case notarization improvements. We need every testnet participant to upgrade to this version as soon as possible to resume notarizations for the ETH bridge and the pbaas chains. Once enough people upgrade, the testnet upgrade can be broadcasted via oracle and we can get closer to getting pbaas on mainnet.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents,

marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Veru...

Assets 6

v0.9.9

 [Asherda](#)
[v0.9.9](#)
[ad8fb8d](#)
[v0.9.9](#)

Announcing v0.9.9 - PBaaS Mainnet Preparation Release, MANDATORY FOR MAINNET AND CONTINUED TESTNET USE

What's New

There is so much new and important in v0.9.9, and we are so close to the mainnet PBaaS release that this is a hard announcement to keep to a digestible size. Key points:

- 1. Preparing for #LaunchPBaaS:** v0.9.9 is MANDATORY, meaning that you will need to upgrade to this release ASAP to stay reliably connected to mainnet after the first preparatory PBaaS upgrade is triggered, both for testnet and mainnet. While the current deadline for being upgraded is in 1 week, it is a soft fork that we would like to activate as soon as possible, and if we can believe that we have more than 50% of the validating network updated earlier than 1 week, we will trigger the activation with our Oracle Notification Technology described in 2. **(Mainnet + Testnet)**
- 2. Enabling Seamless Decentralized Network Upgrades:** This release includes a Verus invention of consensus-capable oracle notifications built into the core protocol, which enable dynamic coordinated consensus changes that can be triggered at a time decided after the actual release which enables them, all without any compromise on decentralization. The Verus Community may decide that we are ready to go earlier than one week on this upgrade and activate it, even before its scheduled time. The good news is that this first test upgrade will primarily affect testnet and prepare for the mainnet PBaaS upgrade, so it is a perfect test case for a mainnet test run before we #LaunchPBaaS. This release actually has the ability to upgrade mainnet to PBaaS, but we believe it is prudent to have at least some time on this new protocol on testnet before the v1.0 release, which will upgrade mainnet. If we make no additional changes to the protocol in this release, which we do not intend to do at this time, this version may be able to follow the v1.0 upgrade. If we determine that it cannot, the oracle upgrade technology will enable the daemon to recognize that it is not capable and request that the user upgrade to a newer version without requiring a bootstrap for the upgrade. **(Mainnet + Testnet)**
- 3. This upgrade fixes all known testnet issues with no reset:** Using v0.9.9, you should be able to sync fully and resume all cross-chain operations on the Verus test network, if you were not able to perform cross-chain operations or sync to the network on the prior version. As mentioned, someone seems to have mined with an intermediate version along the way on testnet and made the network unavailable without a bootstrap on the older daemon. This version should fix that issue as well as any cross-chain difficulties. Auto-notarization will be enabled on testnet in sync when the testfork executes. Until then, all witnessed cross-chain operations should be able to resume, if they have had difficulty. **(Testnet until the #LaunchPBaaS release)**
- 4. Bridging Blockchain Networks Without Witnesses:** v0.9.9 is the first release of any blockchain protocol anywhere that we know of which enables cryptographically provable, cross-chain bridging across Verus and all PBaaS and other connected blockchains in the network, either combined with or independent of the need for notary witnesses. With notary witnesses, cross-chain transactions complete faster, but they are also checked against cryptographic evidence, making it difficult to impossible for anyone, including notary witnesses to take any action, even colluding with validators that would go against the wishes

of the most chain power. We have been referring to this new cross-chain consensus technology as “auto-notarization”, as that is what it actually is, but it would also be accurate to call it Proof of Power, as that is how it works. For purposes of ensuring everyone understands the state of the technology, it is fully functional and we have taken great care to ensure it is sufficiently secure, but we have not yet had the protocol proven to an academic standard. Because of that, while it provides a security fall-back against misbehaving notary witnesses and will be fully functional and we believe sufficiently secure without notary witnesses on release, we currently recommend to consider the protocol unproven at this time and that any serious PBaaS chains be launched with operational notary witnesses until further notice.

Using “notarizationprotocol”:1, which is also the default, a chain launch can specify notary witnesses that will be used when they do their job. Even if they stop witnessing for any reason, the protocol will fail-over to auto-notarization, ensuring that cross-chain transactions still function as expected, even if just for people to move their assets to another blockchain, although significantly more slowly and based solely on cryptographic proof and challenge resolution between merge miners and stakers. This cross-chain proof and challenge protocol operates even when witnesses are also operating and if miners and stakers continue to prove a more powerful chain than the one witnesses represent, the most powerful chain can still be confirmed. (**Testnet until the #LaunchPBaaS release**)

5. **Expanding L1 VerusID APIs:** This release provides a major enhancement to the identity content multimap support with two new APIs, `getidentityhistory` and `getidentitycontent` that provide real-time mempool access and also introduce self-sovereign delete operations to identity content that can be used by applications, making it easy for applications to index, organize, and aggregate an unlimited amount of user content referenced by an identity and made available to the application. This is the technology we used for oracle upgrade notifications, as all of this is available at L1, making it possible to imagine future protocols that can deal smoothly with diverse community opinions worldwide, but always respect the sovereignty and decisions of all members on the network. (**Testnet until the #LaunchPBaaS release**)
6. **Introducing Restricted VerusID Staking for PBaaS Chains:** This release enables the stretch goal of restricted ID staking as an option for launching PBaaS chains. This option means that on such a PBaaS chain, only funds controlled by IDs with a parent of that PBaaS chain may stake blocks on that chain. When combined with restricted forms of ID issuance, whether approval or referral required and/or referral rewards, this capability opens up an entire new area for public/private blockchains and applications, as well as government, voting security, corporate, or organizational use cases. (**Testnet until the #LaunchPBaaS release**)

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming

- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even con...

Assets 6

1 person reacted

v0.9.6-2

 [Asherda](#)

[v0.9.6-2](#)

[86d0881](#)

[v0.9.6-2](#)

v0.9.6-2, CRITICAL UPDATE FOR MAINNET

This release contains a critical update for the Verus daemon. This version does not support testnet, but a future version will be released shortly to support it. For more information on running testnet, please refer to the previous release or the next one.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an

unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and

a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will

automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains....

Assets 6

v0.9.6-1

 [Asherda](#)

[v0.9.6-1](#)

[1b02282](#)

[v0.9.6-1](#)

v0.9.6-1, RECOMMENDED UPDATE FOR MAINNET, CRITICAL FOR CONTINUED TESTNET AND ETHEREUM BRIDGE (GOERLI) USE

What's New

THERE IS NO TESTNET RESET, AND ALL OTHER NOTES REMAIN THE SAME AS LAST RELEASE.

v0.9.6-1 has no mainnet changes and fixes the following testnet issues:

- Fault when initiating mining and staking on a small number of machines
- Sync from scratch issues on testnet

This release fixes all reported issues in all community testing. There are no protocol changes, and no known changes required. All network functions should work as intended.

Verus PBaaS and DeFi RC6 - BUG BOUNTIES

At this time, we are not aware of any bugs or errors in the DeFi, ID, or PBaaS cross chain protocols on this version of testnet. We do not anticipate any changes to the PBaaS protocol before mainnet

release, and we hope you will join us on testnet trying out all of the incredible capabilities and looking for any potential, remaining issues that can possibly be found before mainnet release. While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people (3 still unclaimed) who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended. If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 \$VRSC.

If you have reported a bug, since the bounties for the 0.9.5+ versions were announced, you are on the list, and someone from the Foundation will reach out to you in the near future.

The rest of these release notes are the same as the prior release. We encourage you to drop by the #pbaas-development channel, help the community prepare for mainnet release this coming week, and start building your vision today!

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's

the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along w...

Assets 6

2 people reacted

v0.9.6



[Asherda](#)

[v0.9.6](#)

[7f971ed](#)

[v0.9.6](#)

Announcing v0.9.6, HIGHLY RECOMMENDED UPDATE FOR MAINNET, MANDATORY FOR NEW, RESET TESTNET AND ETHEREUM BRIDGE (GOERLI) USE

v0.9.6 RESETS TESTNET AND COMPLETES ALL PROTOCOL UPDATES FEATURES AND CAPABILITIES EXPECTED FOR THE MAINNET PBAAS UPGRADE. WE DO NOT HAVE ANY NEAR TERM PLAN FOR ANOTHER TESTNET RESET AFTER THE v0.9.6 UPGRADE.

IF YOU HAVE USED TESTNET WITH VERSIONS PRIOR TO v0.9.6, MAKE SURE TO DELETE AND CLEAR ANY EXISTING TESTNET DATA FOLDERS AS DESCRIBED IN THE RELEASE NOTES LINKED BELOW.

What's New

For mainnet, v0.9.6 fixes the following issues:

- An error in fee calculation that can result in nodes having difficulty accepting large, complex transactions with many z-outputs into their mempools.
- Marketplace offers were not showing up properly under getoffers.
- When using the `-privatechange` wallet option with a private change address, attempts to make marketplace offers could fail.

For testnet, v0.9.6 adds the following:

- Caching and performance improvements for advanced PBaaS functions, providing enough performance to keep high transaction bandwidth and/or cross-chain throughput under load for each chain.

- Most hardening required for the mainnet release.
- **The Verus/Ethereum Bridge!** v0.9.6 includes the upgraded Ethereum bridge to the Goerli testnet live with all bridgekeeping, Ethereum Gas price tracking, and reward incentive capabilities for miners and stakers. Rewards for bridgekeeping are delivered both on Verus and on the Ethereum blockchain. Access the Ethereum bridge website (<https://ethtest.verus.services>) to move over some Goerli Ethereum or USDC, and participate in the Bridge.vETH liquidity basket launch to hold and earn fees on a basket with vETH, USDC, and VRSCTEST used as the fee converter for all cross chain transactions between Ethereum's Goerli and VRSC testnets.

v0.9.6 also includes support for new parameters -

`arbitragecurrencies=["VRSCTEST", "vETH"]` and `-arbitrageaddress=valid R-address, pubkey, or ID`. They're exciting, but don't try to use them just yet. They are for testing right now, and are preparing for an "auto-arbitrage" feature already supported in the protocol, but which we hope to have in the daemon before mainnet PBaaS activation that combines decentralized on-chain marketplace offers for currency with liquidity basket import processing to provide miners and stakers the opportunity to commit some amount of funds from their node wallet in common currencies to spot and leverage guaranteed arbitrage opportunities when making a block. These opportunities arise between orders on chain and liquidity basket pricing that if leveraged to earn, actually result in P2P crowdsourcing of liquidity for all baskets across the network, both from the blockchain itself as well as miners and stakers, while benefiting all participants in the following ways:

- Miners and stakers can capture guaranteed arbitrage opportunities and arbitrage with any currency that they choose from their wallets.
- Blockchain users can set limit orders and have them filled if they are near the market price for a currency, or if the market moves in their direction.
- LPs and users of liquidity baskets for conversion get as close to market prices as possible on their conversions, never releasing funds from a basket before an on-chain, MEV-resistant auction takes place for the pricing of those assets.

Although this release is likely to be robust enough for a mainnet, we have come this far with quality and will not rush for a few days earlier upgrade. We still plan to review and complete a small number of hardening items before PBaaS 1.0 is ready for its mainnet release.

That said, this release fixes all reported issues in all community testing. Community developers, including myself are grateful for all of the testing many have contributed to to date. It has made it possible to get incredible test coverage across all network capabilities, and we appreciate all of your participation and pushing of boundaries. This release is expected to be the final PBaaS protocol that will be released on mainnet. There are no known changes required, and all functions should work as intended.

The rest of these release notes are the same as the prior release. We encourage you to drop by the `#pbaas-development` channel, help the community prepare for mainnet release in the next couple weeks and start building your vision today!

Verus PBaaS and DeFi RC6 - BUG BOUNTIES

At this time, we are not aware of any bugs or errors in the DeFi, ID, or PBaaS cross chain protocols on this version of testnet. We do not anticipate any changes to the PBaaS protocol before mainnet release, and we hope you will join us on testnet trying out all the incredible capabilities and looking for any potential, remaining issues that can possibly be found before mainnet release. While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people (7 still unclaimed) who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended. If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 \$VRSC.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest, ~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest, ~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsctest\, %AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that ...

Assets 6

v0.9.5-2

 [Asherda](#)
[v0.9.5-2](#)
[bc894ba](#)
[v0.9.5-2](#)

Announcing v0.9.5-2, RECOMMENDED UPDATE FOR MAINNET - CRITICAL FOR CONTINUED TESTNET USE

YOU MUST BE UPDATED TO v0.9.5 OR LATER BY BLOCK 2291830, AT APPROXIMATELY 11:00 PM UTC, SATURDAY, NOVEMBER 19th, 2022 TO RELIABLY STAY SYNCHRONIZED TO THE VERUS MAINNET. WHILE THERE MAY BE A MAINNET PBAAS UPGRADE BEFORE THAT DATE, WE RECOMMEND THAT IF YOU HAVE NOT UPGRADED YET, THAT YOU DO SO NOW.

As with all v0.9.5-* releases, this release updates the Komodo notaries to season 6, with an activation height of 2291830, at approximately 11:00 PM UTC, Saturday, November 19th, 2022.

What's New

For mainnet, this release labels spendable transactions for the getaddressutxos RPC API, which is used by Verus Mobile. This update is critical for users that have not upgraded to v0.9.5+ before block 2291830.

For testnet, v0.9.5-2 introduces minor notarization fixes that will be important for those intending to launch and witness PBaaS chains.

Ethereum Bridge

Although its deployment has been delayed, there are no technical blockers, and we still expect the Ethereum bridge to be brought back up on the new Verus testnet within the next day or two. When it is up, you can access the Verus <-> Ethereum testnet bridge here: <https://ethtest.verus.services/>, and use your metamask to convert or send cross-chain to the Verus network from Ethereum's Goerli. To use these features you will need to get some Goerli Ethereum, either from a faucet or via the Verus testnet liquidity baskets on Verus.

The rest of these release notes are the same as the prior release. We encourage you to drop by the pbaas-development channel, help the community prepare for the imminent mainnet release of the PBaaS upgrade and start building your vision today!

Verus PBaaS and DeFi RC5 - BUG BOUNTIES

At this time, we are not aware of any bugs or errors in the DeFi, ID, or PBaaS cross chain protocols on this version of testnet. We do not anticipate any changes to the PBaaS protocol before mainnet release, and we hope you will join us on testnet trying out all of the incredible capabilities and looking for any potential, remaining issues that can possibly be found before mainnet release. While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people (7 still unclaimed) who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended. If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 \$VRSC.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the

blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh

private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional bas...

Assets 6

v0.9.5-1

 [Asherda](#)
[v0.9.5-1](#)
[9dc792b](#)
[v0.9.5-1](#)

Announcing v0.9.5-1, CRITICAL UPDATE FOR MAINNET MANDATORY FOR CONTINUED TESTNET USE

As with the last release, this release updates the Komodo notaries to season 6, with an activation height of 2291830, at approximately 11:00 PM UTC, Saturday, November 19th, 2022. YOU MUST BE UPDATED TO v0.9.5 OR LATER BY THEN TO RELIABLY STAY SYNCHRONIZED TO THE VERUS MAINNET. WHILE THERE MAY BE A MAINNET PBaaS UPGRADE BEFORE THAT DATE, WE RECOMMEND THAT IF YOU HAVE NOT UPGRADED YET, THAT YOU DO SO NOW.

Please make sure that if you do choose to wait for the possibility of a mainnet PBaaS upgrade before this activation that you do upgrade to the latest release before Saturday, November 19th.

What's New

For mainnet, this release has no material changes over v0.9.5, but is still considered critical, as mainnet users should be upgraded to v0.9.5+ before block 2291830.

For testnet, v0.9.5-1 fixes an issue discovered on testnet where duplicate ID exports blocked cross-chain processing temporarily and also introduces a protocol improvement designed to make the Verus anti-MEV DeFi protocol even more resistant than what we believe was already the leading algorithm to multi-block attempts at MEV. The protocol change makes the following adjustments:

Conversions and cross-chain transfers still roll-up into exports that are collected from one or more blocks of transfer transactions, but to determine when a block is the last of the prior export or first in the next requires a random bit pulled from later blocks.

The last miner or staker to complete a block of an export retroactively receives 10% of all fees from processing that export, including all conversions, once the rollup is processed as an import. This reward used to go to the miner or staker who processes the export, which is no longer the case. Import transactions can now include up to $n/2$ arbitrage transactions, where n is the number of reserves in a liquidity basket currency.

These changes together serve to disincentivize any miner or staker from attempting to exclude a large set of transactions from a block and front run all of them, as they will have an even chance of having all of those excluded transactions put back in with their transaction, turning potential gain into potential loss. They would be better off saving their working capital to arbitrage imports by joining transactions, bringing them close to market, and earning money without risk of loss. In addition, if they do end up putting more value of transactions in a block that caps the export rollup, they will reliably earn meaningful rewards without risk of loss in that way as well.

Ethereum Bridge

Although its deployment has been delayed, we expect the Ethereum bridge to be brought back up on the new Verus testnet within the next day or two. When it is up, you can access the Verus <-> Ethereum testnet bridge here: <https://ethtest.verus.services>, and use your metamask to convert or send cross-chain to the Verus network from Ethereum's Goerli. To use these features you will need to get some Goerli Ethereum, either from a faucet or via the Verus testnet liquidity baskets on Verus.

The rest of these release notes are the same as the prior release. We encourage you to drop by the #pbaas-development channel, help the community prepare for mainnet release in the next couple weeks and start building your vision today!

Verus PBaaS and DeFi RC5 - BUG BOUNTIES

At this time, we are not aware of any bugs or errors in the DeFi, ID, or PBaaS cross chain protocols on this version of testnet. We do not anticipate any changes to the PBaaS protocol before mainnet release, and we hope you will join us on testnet trying out all of the incredible capabilities and looking for any potential, remaining issues that can possibly be found before mainnet release. While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people (7 still unclaimed) who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended. If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 \$VRSC.

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the

blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables id...

v0.9.5

 [Asherda](#)

[v0.9.5](#)

[91fd525](#)

[v0.9.5](#)

Announcing v0.9.5, CRITICAL UPDATE FOR MAINNET AND TESTNET

This release updates the Komodo notaries to season 6, with an activation height of 2291830, at approximately 11:00 PM UTC, Saturday, November 19th, 2022. You must be updated to v0.9.5 or later by that time to reliably stay synchronized to the Verus mainnet. We are also working to get a mandatory update release during these next couple weeks, which will have the Verus PBaaS mainnet upgrade finalized as well. If you would prefer to wait for the PBaaS mainnet upgrade and upgrade once, we cannot guarantee that the PBaaS mainnet release will be available in the next couple weeks, but we are working to make that a reasonable choice as well. Please make sure that if you do choose to wait to upgrade that you do upgrade to the latest release before Saturday, November 19th.

New Signature Capabilities on Mainnet & Testnet

In addition to continued support for all of the `signmessage`, `signdata`, and `signfile` APIs, as well as their `verify*` counterparts, v0.9.5 introduces 2 new APIs and CLI commands, `signdata` and `verifysignature`, that encompass all of the capabilities of previous signing and verification APIs and add new capabilities for advanced signatures and verification. These new advanced capabilities were designed for community work by [@allbits](#), designing a more powerful and flexible framework for NFTs and ID-bound legal rights and agreements. In addition to digital signing of documents or information using an ID, the new APIs enable easily binding together multiple sources of content, documents, identities, and VDXF keys together into one verifiable signature and then being able to verify that signature, including all of its specific bindings.

Verus PBaaS and DeFi RC5 - BUG BOUNTIES

At this time, we are not aware of any bugs or errors in the DeFi, ID, or PBaaS cross chain protocols on this version of testnet. We do not anticipate any changes to the PBaaS protocol before mainnet release, and we hope you will join us on testnet trying out all of the incredible capabilities and looking for any potential, remaining issues that can possibly be found before mainnet release. While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended. If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 \$VRSC.

Emoji IDs and 🎃Chain

In the spirit of this release day, we have taken advantage of VerusIDs and currency emoji support and created a new, merge mineable PBaaS chain, 🎃. It has been said that those who merge mine 🎃chain at midnight will disappear into the crypto-metaverse, never to be seen again, become VerusID avatars, and be forever doomed to being bought and sold on the VRSCTEST marketplace for all eternity (or until the next testnet reset). YOU HAVE BEEN WARNED!

We expect the Ethereum bridge to be brought back up on the new Verus testnet within the next day or two. When it is up, you can access the Verus <-> Ethereum testnet bridge here: <https://ethtest.verus.services>, and use your metamask to convert or send cross-chain to the Verus network from Ethereum's Goerli. To use these features you will need to get some Goerli Ethereum, either from a faucet or via the Verus testnet liquidity baskets on Verus.

The rest of these release notes are the same as the prior release. We encourage you to drop by the #pbaas-development channel, help the community prepare for mainnet release in the next couple weeks and start building your vision today!

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Tokenizing ID control (next generation NFT):

The currency definition have flags `OPTION_NFT_TOKEN` + `OPTION_TOKEN`, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a

parent currency, vrsc-btc , but no referrer. We're now able to use z_addresses to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsctest registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr """
vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsctest registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsctest registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzLt", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@{}"}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":"true","amount":100,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-sig...
```

Contributors



allbits

Assets 6

v0.9.4-5

 [Asherda](#)
[v0.9.4-5](#)
[25ed684](#)

v0.9.4-5

Announcing v0.9.4-5, RECOMMENDED FOR MAINNET AND TESTNET

This release fixes an RPC API issue on mainnet and testnet that can cause a daemon fault when the getaddressmempool API is used. This fault while running is only caused by non-typical API, insight explorer, or CLI use, but if it does happen, may result in the need to rescan or resync your wallet. We know of no one on mainnet affected by this issue. We recommend that all users upgrade.

This release also includes a new capability for testnet, which was pioneered by CHIPS developer [@biz](#), who worked to modify and test the Verus PoP algorithm to support 10 second block times, which the CHIPS chain uses to enable its game application focus. These new parameters will enable the CHIPS blockchain to move to being an interoperable, merge mineable PBaaS chain with a 10 second blocktime when PBaaS goes live on mainnet. To learn how to test the new CHIPS test chain on testnet, visit the #pbaas-development channel, as the support in v0.9.4-5 is not as complete as it will be in the next and hopefully final testnet reset. In that next version, blocktime and the pow averaging window for any PBaaS chain's PoP consensus will be able to be configured between 10 seconds and 2 minutes in addition to its notarization frequency as part of the chain definition, enabling a greater variety of PBaaS use cases, transaction throughput, and application capabilities.

As a reminder, you can access the Verus <-> Ethereum testnet bridge here:

<https://ethtest.verus.services> , using your metamask, converting or sending cross chain through the Verus network from Ethereum's Goerli. To use these features you will need to get some Goerli Ethereum, either from a faucet or via the Verus testnet liquidity baskets on Verus.

The rest of these release notes are the same as the prior release. We encourage you to drop by the #pbaas-development channel, help the community prepare for mainnet and start building your vision today!

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`

- Windows 10: %AppData%\Roaming\Komodo\vrsc\, %AppData%\Roaming\VerusTest or %AppData%\Komodo\vrsc\, %AppData%\Roaming\VerusTest

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes "" to leave fields blank, the example below specifies a parent currency, vrsc-btc , but no referrer. We're now able to use z_addresses to fund the name commitment and identity registration

```
# verus -chain=vrsc test registernamecommitment name controladdress referral
parent sourceoffunds
verus -chain=vrsc test registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsc test registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds

verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "namereservation": { "version": 1, "name": "ID", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzL", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referrer": ""}, "identity": {"primaryaddresses": ["RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@"}}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute

transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditible by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency **"  
'[{"address":"IDNAME@","exportto":"PBaaSChainName","exportid":true,"amount":10  
0,"currency":"vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address":<destination_address>,"amount":<transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an inter...

Contributors



biz

Assets 6

v0.9.4-4

 [Asherda](#)

[v0.9.4-4](#)

[538c91a](#)

[v0.9.4-4](#)

Announcing v0.9.4-4, UNCHANGED FOR MAINNET -- RECOMMENDED FOR CONTINUED TESTNET USE, MANDATORY FOR CONTINUED TESTNET MERGE OR BRIDGE MINING

This release has no material mainnet changes.

This release primarily upgrades the ETH bridge and also fixes a PBaaS cross-chain notarization issue that made miners and stakers consider themselves ineligible to enter cross-chain notarizations. There are no actual protocol changes.

The ETH bridge upgrade introduces the ability to enter notarizations between Ethereum Goerli and VRSCTEST using bridgekeeper without an Ethereum private key in your conf file. This will enable you to still mine and potentially earn by agreeing or disagreeing with notarizations mined into the Verus blockchain, without needing to make transactions on or spend Ethereum for fees. This should also work with a free Infura account.

As a reminder, you can access the Verus <-> Ethereum testnet bridge here:
<https://ethtest.verus.services> , using your metamask, converting or sending cross chain through the Verus network from Ethereum's Goerli. To use these features you will need to get some Goerli Ethereum, either from a faucet or via the Verus testnet liquidity baskets on Verus.

The rest of these release notes are the same as the prior release. Drop by the #pbaas-development channel to help us prepare for mainnet and start building your vision today!

Tokenizing ID control (next generation NFT):

The currency definition have flags OPTION_NFT_TOKEN + OPTION_TOKEN, and a max supply of 1 satoshi that is either pre allocated or pre-converted to. If the token is pre-allocated, then the maximum pre-conversion must be 0.

```
verus -chain=vrsctest definecurrency
'{"name":"ID","options":2080,"preallocations":
[{"ControlTokenRecipient@":0.00000001}],"maxpreconversion":0}'
```

Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest`, `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest`,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest` or `%AppData%\Komodo\vrsctest\`, `%AppData%\Roaming\VerusTest`

Creating an identity with a fractional currency as its parent

`registernamecommitment` now takes two more positional arguments to specify a currency parent and a funding address. Use quotes `""` to leave fields blank, the example below specifies a parent currency, `vrsc-btc` , but no referrer. We're now able to use `z_addresses` to fund the name commitment and identity registration

```
# verus -chain=vrsctest registernamecommitment name controladdress referral
parent sourceoffunds
```

```
verus -chain=vrsc test registernamecommitment subID
RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr "" vrsc-btc
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

Specify the parent in the identity definition. Enter `false` for `returntx` to sign and submit the id registration, `0` for the `feeoffer` to use the default fee, and the funding identity, transparent address, or z-address

```
# verus -chain=vrsc test registeridentity '{ID registration with name
commitment}' returntx feeoffer sourceoffunds
```

```
verus -chain=vrsc test registeridentity '{"txid": "67635331cbccb7a2cbf408a9e97b3f8986133964e0315a8b9fd237a5fd95ac8f", "name": "name", "parent": "i84mndBk2Znydpgm9T9pTjVvBnHkhErzL", "salt": "b7070f2ca7495e49c85ab41b5a368150e2c217be6d08cc4102a1b682cddb6f01", "referral": ""}, "identity": {"primaryaddresses": [{"RDnf7mH7RQki9b7PqdBD2Er6WXv3DTawGr"}], "minimumsignatures": 1, "name": "ID", "parent": "vrsc-btc@{}"}' false 0
zs1s2mteau9tcalvk55cnepw3aq7dr6w7f447pqqkxczat3a02208d3ersx60wz9srw3nkd25ppfny
```

If a currency's ID issuance require permission from the currency's identity then it must sign the name commitment and identity registration. Either use the parent identity to fund those transactions, or receive a raw transaction to give the identity owner to sign by setting `returntx` to `true`

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

Exporting an ID to a PBaaS chain

```
verus -chain=VRSCTEST sendcurrency "*"  
'[{"address": "IDNAME@", "exportto": "PBaaSChainName", "exportid": "true", "amount": 100, "currency": "vrsctest"}]'
```

Signing transactions from multi-signature IDs (testnet and mainnet)

Create transaction, get raw transaction data:

```
verus sendcurrency <multi-signature-ID>@  
'[{"address": "<destination_address>", "amount": <transaction_amount>}]'  
verus z_getoperationstatus <operation_id_returned_by_sendcurrency>
```

Take the raw hex transaction data provided by z_getoperationstatus to each additional wallet(s) containing the additional signing addresses/IDs:

```
verus signrawtransaction <raw_hex_transaction>
```

After the last necessary signature is applied, broadcast on the network using:

```
verus sendrawtransaction <raw_hex_signed_transaction>
```

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking...

Assets 6

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
- [Privacy](#)
- [Security](#)
- [Status](#)
- [Docs](#)
- Contact GitHub
- [Pricing](#)
- [API](#)
- [Training](#)
- [Blog](#)
- [About](#)

Releases · VerusCoin/VerusCoin · GitHub

[Skip to content](#)

•

- Pricing



Sign in

Sign up

[VerusCoin](#) / [VerusCoin](#) Public
forked from [miketout/VerusCoin](#)

- -
 -
 - [Code](#)
 - [Issues 21](#)
 - [Pull requests](#)
 - [Actions](#)
 - [Projects](#)
 - [Wiki](#)
 - [Security](#)
 - [Insights](#)

Releases: VerusCoin/VerusCoin

[Releases](#) [Tags](#)



v1.0.5



[Asherda](#)

[v1.0.5](#)

[8022fec](#)

[v1.0.5 Latest](#)

Announcing v1.0.5 - MANDATORY MAINNET UPGRADE WITH DEADLINE TBD BY COMMUNITY (STRAWMAN SUGGESTION MONDAY, MAY 29th, 19:00 UTC)

While preparing the cross-chain VRSC currency launch and as part of continual review, we determined that one numeric function that is used in the cross-chain challenges and is calculated differently for mainnet, due to the original launch that included the deprecated time locked rewards, was incorrect. This had never been hit, as it would have been calculated incorrectly only if multiple chains experienced a challenge of two competing forks, but in that case, it would falsely reject any challenge proof. That would mean cross-chain challenges on mainnet, even though they were fine and fully tested on testnet, could fail and block a bridge until that calculation was made correct and the network upgraded.

Because we have the notification oracle technology, we triggered a rip-cord notification selectively for cross-chain that made all nodes listening to the oracle, which we expect to be near 100%, disallow cross-chain operations without affecting same chain operations. That means that all functions on the Verus network, except launching PBaaS chains or Ethereum bridges are fully functional and unhampered in any way at this time. If someone tries to launch a PBaaS chain while this oracle notification is active, all nodes following the oracle will reject such an action without incident until the notification is removed.

This release, v1.0.5, properly calculates the aptly named “magic number” for each chain, in the correct way on the mainnet network, and as long as the notification is set, is fully compatible with current versions on the network.

Once everyone has had a chance to upgrade to v1.0.5, the network/community should agree to remove the notification from the oracle and allow PBaaS launches without false rejection of cross-chain challenges. Once we do that, and when someone launches a PBaaS chain that issues such a challenge, the network would become incompatible with Verus versions prior to 1.0.5.

From our perspective, we are not aware of urgent efforts to launch a PBaaS chain in the next few days by anyone who cannot wait. All things considered, we believe that we can at least wait until the community discussion on Saturday, and that a strawman deadline would be **Monday, May 29th, at 19:00 UTC** for everyone to have upgraded. If you are urgently wanting to launch a PBaaS chain, have a plan, and are prepared to do so sooner, we invite you to join the community meeting on Saturday at 19:00 UTC and let everyone know.

Having this notification remain in place until removed keeps the network running smoothly as it has been since activation, with all DeFi, multicurrency, and non-cross-chain functions enabled. Since the ETH bridge is undergoing final changes and review, is not quite ready to launch on testnet or mainnet, and will likely require at least a couple days on testnet to finish its review, we do not believe the current notification is hampering or slowing anything or anyone down at this time.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your

funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved...

Assets 6

1 person reacted

v1.0.4

 [Asherda](#)

[v1.0.4](#)

[5ab050b](#)

[v1.0.4](#)

Announcing v1.0.4 - CRITICAL, NON-MANDATORY MAINNET UPGRADE
THIS UPDATE IS CRITICAL FOR MAINNET DEFI OR CROSSCHAIN WALLET
FUNCTIONS, AS IT RECTIFIES POTENTIAL REFUND FAILURES IN EDGE CASES - IT IS
NOT REQUIRED FOR MAINNET SYNCHRONIZATION OR MAINNET FUNCTION PAST
PBAAS ACTIVATION, BUT IT IS HIGHLY RECOMMENDED.

DO BE SURE TO UPGRADE TO v1.0.3 OR LATER BEFORE MAY 23, 2023, EXPECTED
BLOCK #2549420

FOR CONTINUED TESTNET USE, UPGRADE ASAP

Mainnet improvements:

- Properly inserts refund addresses in all cases when making DeFi or cross-chain transactions
- Merge mining improvements from [@hellcatz](#) for higher merge mining performance on all platforms and full pool merge mining support when used in combination with [@hellcatz](#) and [@Oink70](#)'s SNomp pool improvements
- CLI now defaults to mainnet for all PBaaS chains unless -testnet is specified

- 2 versions of GUI will be released for now, a mainnet and separate testnet version. Full PBAAS support in the wallet will only be for either mainnet or testnet, depending on the version used.
- Extends deprecation height from the previous 20 weeks to 52 weeks

Testnet fixes/improvements:

- Separate GUI release for testnet
- Refund address fix

This is the last expected and recommended release before the mainnet activation, Tuesday at block #2549420, which will add some pool merge mining improvements. Please make sure you update to v1.0.4 or greater before the activation block.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the

blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh

private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, s...

Contributors

- 
- 

hellcatz and Oink70

Assets 6

1 person reacted

v1.0.3

 [Asherda](#)

[v1.0.3](#)

[42e83aa](#)

[v1.0.3](#)

Announcing v1.0.3 - MANDATORY MAINNET UPGRADE - HOPEFULLY THE LAST IN A WHILE - ACTIVATION DATE HAS BEEN PUSHED BY 2 DAYS

FOR CONTINUED MAINNET USE, UPGRADE TO v1.0.3 OR LATER BEFORE MAY 23, 2023, EXPECTED BLOCK #2549420

FOR CONTINUED TESTNET USE, UPGRADE ASAP

Mainnet: fixes:

- Fixes an issue discovered that could have resulted in behavior different than what has been tested in an edge case when mainnet activates

Testnet fixes/improvements:

- Forgives issues pre-1.0.2 on testnet that were blocking challenge resolution by lacking complete cryptographic proof for cross-chain notarization confirmation

What's New

v1.0.3 has extended the API fundrawtransaction to allow Verus Mobile and other applications to easily construct transactions for DeFi conversions and cross-chain operations as well.

We expect at least one more non-mandatory release before the mainnet activation, Tuesday at block #2549420, which will add some pool merge mining improvements. Please make sure you update to v1.0.3 or greater before the activation block.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus

PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch

discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchain...

Assets 6

1 person reacted

v1.0.2



[Asherda](#)

[v1.0.2](#)

[576c5b2](#)

[v1.0.2](#)

Announcing v1.0.2 - MANDATORY MAINNET UPGRADE
UPGRADE TO v1.0.2 OR LATER IS MANDATORY FOR CONTINUED MAINNET AND
TESTNET USE AFTER MAY 21, 2023, EXPECTED BLOCK #2546600

Mainnet: fixes:

- Fixes a regression that prevented staking on locked IDs as well as modifying locked IDs

Testnet fixes/improvements:

- Fixes issues discovered when challenges occurred on the Gravity chain that blocked resolution of cross-chain challenges.
- Addresses all known issues discovered in testing

What's New

v1.0.2 has no feature changes, some minor API fixes, proof improvements, and we believe the third time's a charm 😊. The testnet will transition to the v1.0.2 protocol with no reset required, Wednesday, May 17, 2023 0:00:00, UTC.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming

- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an or...

v1.0.1

 [Asherda](#)

[v1.0.1](#)

[d733d8e](#)

[v1.0.1](#)

Announcing v1.0.1 - MANDATORY Mainnet Upgrade, UPGRADE TO v1.0.1 OR LATER IS MANDATORY FOR CONTINUED MAINNET AND TESTNET USE AFTER MAY 21, 2023, EXPECTED BLOCK #2546600

Issues addressed

In addition to fixing an issue discovered by cautionfun#3236 on the current public testnet, this version also addresses an issue discovered during additional coverage testing in the oracle based, reversible rip-cord implementations to ensure the ability for the network to respond to any unexpected events quickly, safely, and without compromise to decentralization.

What's New

In addition to the full PBaaS protocol and all capabilities previously described, version 1.0.1 will also include some new capabilities that were previously not in the PBaaS protocol. The capabilities basically extend the idea of a centralized currency to include the ability to launch temporarily centralized currencies, including those that can issue identities. This is achieved by setting the “endblock” on a centralized currency launch of either a token or liquidity basket, enabling:

1. Minting of currency on demand by the identity of the currency as part of the `sendcurrency` command.
2. Burning a liquidity basket currency, but instead of affecting the supply/reserve ratio as normal burns do, the identity of a centralized currency can also burn currency to affect the reserve ratio weights of the currency.
3. Registering identities on a currency. When an identity is registered on a centralized currency, the fee for the registration is not burned into the currency, but sent directly to the identity of the currency. This is true for token or liquidity basket currencies. For decentralized currencies, fees are burned, which on a liquidity basket, puts them into the liquidity pool for all LPs to share in earnings, and for a standard token, reduces the supply and available future registrations on that currency.
4. In v1.0.1, setting the endblock of a centralized currency causes all centralized capabilities of the same ID to end, converting it in whatever state it currently is in a decentralized currency with no special control over it after that occurs by any identity on the network.
5. v1.0.0 did not allow a decentralized non-liquidity token to register identities. v1.0.1 does.

Thanks to ejuliano#8606 for suggesting the initial idea that led to the realization that these features could be added quite easily and with very little protocol risk. All of these new capabilities have had full coverage testing and validation over the past week thanks to [@Asherda](#)'s leadership on that front.

Meanwhile, [@alexenglish](#), [@monkins1010](#), [@Asherda](#), quipacorn#5205, and others have been working on the Ethereum bridge deployment in preparation for this release. The testnet will transition to the v1.0.1 protocol with no reset required, Monday, May 8, 2023 4:00:00 PM, UTC.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operatio...

Contributors

- 
- 
- 

alexenglish, Asherda, and monkins1010

Assets 6

1 person reacted

v1.0.0

 [Asherda](#)
[v1.0.0](#)
[f672ccf](#)
[v1.0.0](#)

Announcing v1.0.0 - PBaaS Mainnet Upgrade, UPGRADE TO v1.0.0-x MANDATORY FOR CONTINUED MAINNET AND TESTNET USE AFTER MAY 21, 2023, EXPECTED BLOCK #2546600

What's New

This release delivers on every part of the vision (and most stretch goals) either described in the vision paper of June 2018 or ever planned for the PBaaS release. Thanks to the incredible combined efforts of so many people in the community ranging from development, companies and projects joining and their open contributions (Valu/Arkeytyp, CHIPS, vDEX, VaultAlert, cragslist, and more) to community members supporting users to helping educate others who can contribute as well, we have run these protocols now for years as we've improved them, and all of these capabilities will go live on the Verus mainnet May 21, 2023, with the activation expected targeting block 2546600.

As a community, we've discussed what the Verus PBaaS protocol can do for years. Now that it has existed on testnet for as long as it has, only those who have either been part of those conversations or experienced it first hand have a real sense for how much better Verus PBaaS is as a solution for cross-chain, DeFi AMMs, decentralized markets, self-sovereign provable recoverable IDs, NFT capabilities, anti-phishing, anti-MEV, scale, or just about any of the challenges people have on crypto platforms today. In fact, as we prepare for activation of this protocol that enables so many new use cases for crypto, not to mention easy onboarding when entrepreneurs discover how to really leverage VerusIDs, PBaaS is so far beyond what people are experiencing on any crypto platform today that it has been easy to dismiss our community as describing the impossible.

Once we are live on mainnet and people can actually use all of this themselves, along with every project, chain and currency on the PBaaS network and EVERY ERC20 or ERC721 on Ethereum or even bridged to Ethereum, the truth of Verus will be self-evident, and those who build on Verus will more easily build faster, better, more secure apps and services with the possibility of provable identity + privacy, crowdfunded projects, businesses, economies, and public infrastructure efforts all seamlessly integrated into UIs that do not need business deals, permission, or any centralized infrastructure to connect services and users, enabling everyone to communicate in provable, private, or public ways that always include bidirectional, secure commerce of all kinds.

This core capabilities of this release include (lots to learn to understand it all, but here's a partial list):

1. **Mainnet PBaaS Activation MAY 21, 2023, EXPECTED BLOCK #2546600:** v1.0.0 is MANDATORY, meaning that you will need to upgrade to this release or a later version v1.0.0-x, before MAY 21, 2023 to stay reliably connected to mainnet. Testnet will also be reset within the next day, and this release will allow you to connect to the new testnet, which we see as a core development platform and staging area for the Verus PBaaS network and have no plans to reset in any foreseeable future.
2. **Verus Intersystem Protocol (VIP) - Layer 0 Multichain and Inter-chain Protocol:** To our knowledge, the VIP protocol is the only fully decentralized, provable cross-chain technology available on any network that is based on cryptographic proofs of each chain with optional witnesses to confirm chain state. From what we've been able to learn about LayerZero, Cosmos, Polkadot, Thorchain or others, VIP is different from (though closest in some ways to LayerZero), superior to, and more decentralized than all other cross-chain protocols we have seen. We can of course get into any level of discussion on the topic, but expect more descriptions and educational material to follow, now that the protocol was originally conceived of 5 years ago, and has been developed and tested/hardened for years.

3. **VerusID Content Multimaps - Unlimited Provable Data for Every VerusID:** With the PBaaS upgrade, VerusIDs gain a powerful new capability that can be used for social networking, voting, oracles, publishing any information, various forms of provable attestations, ratings, and much, much more. With the scalability of PBaaS and VerusID content multimaps together, the Verus PBaaS network becomes the most scalable, permissionless, provable source of human data from which AIs can learn about humanity in the world. We don't believe anything else even comes close, and with the recent rise of AI, it feels great to have this come together as a core original goal at a very appropriate and important time in history.
4. **Permissionless Chain, Token, and Liquidity Basket Currency Launches:** Whether you are looking to crowdfund an effort in a manner much like Kickstarter or IndieGoGo, sell identities to which you expect to add value or that are sub-IDs of a very cool root ID, or launch an entirely new blockchain economy and blockchain that starts with all of the Verus technology and a bridge to Verus and Ethereum from day 1, Verus is probably the best platform for you. In fact, as we were planning for the next testnet and wanted to have our own version of DAI to use in the bridge, since we want only decentralized currencies in the live mainnet bridge, we used a command to launch a token on the old testnet, exported it and all its supply to Ethereum's Goerli on the existing bridge, and that will be the DAI proxy/simulant that we will use on the new testnet. It saved us all some time, proving that Verus PBaaS will also be the easiest, most efficient way to launch even an Ethereum ERC20, whether that ERC20 represents a token, DeFi basket, or even another PBaaS blockchain 😊
5. **Decentralized Ethereum Bridge:** Shortly after PBaaS activates on mainnet, the Ethereum bridge will launch as a decentralized gateway and 1:1 provably mapped currency called "vETH" on the Verus network, also available to all PBaaS chains. The Ethereum bridge will also include a 100% backed basket of 33% Verus, 33% Ethereum, and 33% DAI, which will have the following functions:
 - Auto conversion of fees from Verus <-> Ethereum when sending cross chain, based on the on-chain conversion price in the liquidity basket
 - Permissionless ability to register *.vETH IDs, which in addition to the normal sub-ID capability of creating a single token that has control over the ID and can be exported to Ethereum as an automatic ERC721 NFT, can alternately be used to create a "mapped currency", which can be provably mapped 1:1 to any ERC20 currency on the Ethereum network by exporting it to the vETH bridge. Only Verus root IDs, and IDs of a gateway can create mapped currencies to a gateway.
 - Decentralized, fair bridge launch. The vETH bridge will launch as other currencies, chains, and gateways can launch, including the bridge converter. When the bridge launches, the contracts will have a surplus of Verus that comes from the fees paid to launch the gateway. In the protocol, that surplus is first used to solve the chicken and egg problem of none of one currency on the other chain. The contracts will cover the Verus fees for all sends from Ethereum before the Bridge converter launches, the liquidity basket becomes active, and the bridge recognizes that launch. Once that happens, the remainder of $\frac{1}{2}$ the fees (5000 VRSC) that are left over from the launch and did not go to miners and stakers or pay Verus fees for people sending from Ethereum

6. The Verus Fee Pool and Rewards: Another technology and solution unique to Verus, the Verus Fee Pool technology goes live with PBaaS. What this means is that as people use the cross-chain, DeFi, or purchase/register IDs on the network, the miner or staker will put all fees into the on-chain “fee pool”, and then take 1/100th of the pool in addition to the block reward. This gives everyone incentive to still prioritize transactions based on fees, while preventing validators from gaming the fee system by washing fees-in vs. fees-out or working to reorg/rewind to capture a particularly juicy block reward. Instead, this technology aligns all validator incentives with the health/proper operation of the network and creates a circular on-chain economy that can last well beyond block rewards.

For those who actually understand the challenges actual users have in crypto today, this release represents a historic move forward in public cryptographic networks and credibly and actually neutral infrastructure for society-wide human and AI collaboration and learning. As a community, this version really does realize the vision laid out in the original Verus Vision Paper, and at the same time provides much more than was described there. We are a community of individuals, and it is only because that is what we are that we have all arrived here together. The rest of the world is stuck on their Munchausen’s protocols, trying to figure out how to share the front-running and back-running spoils taken from users, but Verus activation gives them a better path forward.

As always, we do not promise, we describe a vision and welcome contributors from everywhere. Now that we, as a community, have delivered on the original phases of the vision together, it is time for all of us to consider what we might want to build over this incredible, geoscale, ID-enabled, fully decentralized platform. Our next efforts across the community should be to realize the promise of this network in the use cases we create, use cases that we will have an advantage creating on Verus over any other platform because it really is that much better.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertisin...

Assets 6

2 people reacted

v0.9.9-5

 [Asherda](#)

[v0.9.9-5](#)

[a7adac2](#)

[v0.9.9-5](#)

Announcing v0.9.9-5 - CRITICAL PRIVACY FIX PARSING FRIENDLY PRIVATE NAMES, CRITICAL UPDATE FOR RELIABLE TESTNET OPERATION

Mainnet Changes - Wallet Hardening

v0.9.9-5 Fixes an issue introduced in v0.9.9 during the ID name hardening work that prevented parsing id@:private to the z-address of id@ when running a native node. Instead, such an address parsed to the main id@. Since this issue could impact on-chain privacy of someone intending to send to or from a private address and using friendly names, we are considering this a critical privacy release.

This version also prevents the wallet from crediting transactions that can be constructed to make a burn look like a spendable output, ensuring that unspendable funds are not added to a wallet or address balance. [@Alrighttt](#), who has been focusing on Verus security testing, reported a way that someone might create such an output, and we appreciate that it is before we have ever seen such transactions on the public or test networks. Thanks [@Alrighttt](#) for the report and discretion that helps keep them off the network(s) altogether! We recommend that everyone update as soon as possible to version 0.9.9-5 or later.

We are working towards the mainnet PBaaS upgrade version 1.0.0, but we cannot put a specific date beyond  SOON on its release.

What's New for TestNet

This release fixes hopefully the last remaining issue that was causing testnet users to experience sync issues on PBaaS chains.

There are no other significant changes.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that incre...

Contributors

- _

Alrighttt

Assets 6

1 person reacted

v0.9.9-4

 [Asherda](#)
[v0.9.9-4](#)
[94df9fe](#)
[v0.9.9-4](#)

Announcing v0.9.9-4 - RECOMMENDED FOR MAINNET, MANDATORY UPDATE FOR RELIABLE TESTNET OPERATION

Mainnet Changes

Low level stakeguard improvements, no functional changes

What's New for TestNet

This release fixes an issue that was causing testnet users to experience sync issues on chains that had more than one oracle upgrade for the same capability activated.

There are no other significant changes.

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your

funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Simultaneous blockchain and blockchain liquidity pool launches

- Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain. Bridge converter currencies have the same flexibility as other fractional 100% asset backed or partially asset backed currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Blockchain-based, crowdfunding currency launches with minimum participation or automatic refunds, including for dual launches (blockchain and bridge)

- Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees. The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts

for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

An interoperable, multichain network for new use cases and unlimited scale**

- The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network. Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access. There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the...

Assets 6

v0.9.9-3

 [Asherda](#)
[v0.9.9-3](#)
[2213c9f](#)
[v0.9.9-3](#)

Announcing v0.9.9-3 - FULLY OPTIONAL UPDATE FOR MAINNET, MANDATORY UPDATE FOR CONTINUED TESTNET OPERATION

Mainnet Changes - no protocol changes

V0.9.9-3 introduces the `prunespentwallettransactions` CLI command and RPC API, providing a method for holders on testnet or mainnet of very large wallets to prune older spent transactions, which can improve performance for wallets that may be slowing down due to size. Thanks [@himu007](#) for the PR! There are no mainnet protocol changes in v0.9.9-3 over the prior release.

What's New for TestNet

This release includes an activation that allows a major upgrade to the testnet Ethereum bridge contracts that is both not expected to be needed again on testnet and not ever needed on mainnet with the new contract upgrade model. This upgrade should clear all known issues on the bridge to Goerli, meaning any remaining bridge transactions will be delivered, and the ETH bridge should then be fully functional.

The Path for PBaaS to Mainnet

This testnet has actually been through a lot already, more than we expected before mainnet, and this last experience, along with its full restoration of cross-chain and all function has been very useful and was the impetus for the Verus Oracle Notification technology. In the process of getting here, we implemented the necessary changes and fixes to get everything back on track, but we also have remnants of these events in the daemon, and on the final protocol, these are there to enable compatibility with the history of testnet, not because we need these upgrades on mainnet PBaaS. It gave us experience, increased confidence, and pushed us to develop a fast, decentralized network response capability. At the same time, it would be best for our code base, both on the Verus side and in the ETH contracts if that testnet history is removed from the code before mainnet activation. Doing so would make the next upgrade incompatible with today's testnet and its earlier history.

As a result, we have decided that we will create one final testnet, but this time, we would do it a little differently than in the past. We will extend the pre-PBaaS upgrade on the existing testnet to a future date via the oracles. If you have started a chain and control the default oracle on any chain, please check the [#pbaas-development](#) channel for instructions on setting your oracle to extend the upgrade.

We will leave the current testnet running and fully functional, as we prepare the next release, which we expect to be ready for a mainnet PBaaS activation. We will effectively abandon this testnet and leave it running long enough to overlap with the next and final testnet instance before PBaaS release. You should be able to actually send currencies over from the first testnet to the second via Ethereum's Goerli testnet, even map some VRSCTEST-CLASSIC or something else through the bridge, but we do not intend to update that daemon again, or continue to run nodes or notaries on the old test network for more than a couple weeks after the new testnet goes live.

Thanks for all your help and contributions, both already made and those to come for this historic upcoming release!

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's

the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Consensus integrated DeFi liquidity pools and fractional currency baskets

- Any ID owner may define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100% backing. The Verus DeFi protocol ensures that all curren...

Contributors

-  [himu007](#)

himu007

Assets 6

v0.9.9-2

 [Asherda](#)

[v0.9.9-2](#)

[69628ef](#)

[v0.9.9-2](#)

Announcing v0.9.9-2 - CRITICAL SECURITY UPDATE FOR MAINNET AND REQUIRED UPDATE TO RESUME SUCCESSFUL MERGE MINING NOTARIZATION ON TESTNET

Important Mainnet Security Updates**

v0.9.9-2 introduces new mitigation code for the **Rab13s** vulnerability, announced by Halbourne 3 days ago as a zero day vulnerability in over 280 blockchains, including Zcash and Verus, that could allow someone to attack and bring down specific nodes that they may target through exhausting the node's resources. The Verus fix adapts a fix implemented in Zcash that also does a better job of preserving privacy while gossiping transactions around the network by making it harder to track their node of origin via listening nodes.

This release also includes an important update of the OpenSSL library, which Verus uses for encrypted node connections and addresses a buffer overflow vulnerability in that library.

What's New for TestNet

This release re-enables cross-chain notarization, and represents the completion of a great deal of learning from the first roll out of auto notarization. Basically, we encountered two issues:

1. There was a notarization bug in a significantly updated approach that caused notarizations to fork, effectively simulating a worst case, unmitigated attack on the notarization system. While that did not cause the system to fail per se, it resulted in the generation of many perceived, although not real, notarization forks on the network.
2. Due to our effort to err on the side of more evidence than we believed was sufficient with auto notarization, then optimize to reduce the proofs required, the protocol was simply taking too long and generating too large proofs to be reasonable, required, or what we need on mainnet in any real scenario.

As a result, we will take a few days to change some parameters in the auto notarization protocol that will both reduce the total and maximum amount of proof required, as well as provide a sufficient level of cryptographic proof, similar in strength to the current proof model with more efficient and selective proof choices. We don't expect this important step to take long, and until then, we will leave testnet on the pre-PBaaS to mainnet preparatory state.

The good news is that since we have the new Verus Notification Oracle technology, we can enable everyone running on the current testnet to remain connected as all cross-chain transactions to and from PBaaS chains will resume flowing. We are still seeing an error in BridgeKeeper, but since this release includes a mainnet security update and addresses all other known issues on testnet, we'll release this immediately and update Bridgekeeper as soon as possible. In order to merge mine PBaaS chains with Verus or the Ethereum bridge to Goerli going forward, you will need to update to v0.9.9-2, which everyone should run for mainnet anyhow.

Once you update, please feel free to resume merge mining all PBaaS chains and Verus. You will need to wait for the BridgeKeeper update to be able to merge mine and stake with Ethereum's Goerli.

Thanks for all your testing! We've learned a great deal from this upgrade, and although it may feel like a delay that we took the time to learn and address it in place on the public network before moving on, this learning and consequent hardening will ensure the mainnet release will be more hardened than ever. Next steps are to upgrade auto notarization to be resilient to anything like this level of buildup over weeks automatically and prepare the mainnet release. Join us in testing everything in the #pbaas-development channel. Contribute as you learn and build, and let's all make history together!

Additional Verus Capabilities

- On-chain Launches of Token, Centralized Currency, and Liquidity Basket AMMs
- On-chain Launches and Merge Mining of Independent, Connected, Interoperable Blockchains without Programming
- On-chain Self Sovereign, Provable Identities, NFTs, and Individual or Organizational Profiles

Verus ID and NFT Marketplace

Buy and sell VerusIDs on-chain, advertising your offer directly to the owner of an ID or NFT, or posting the sale of your NFT on the worldwide blockchain for all the world to see. Execute transactions in a completely decentralized way. Pay or offer to pay from a transparent or zero-knowledge private address, still auditable by you. Accept payment to either as well, and best of all, execute your transactions directly, peer-to-peer without any intermediary necessary. Don't worry the on-chain model still makes room for owners to select and share proceeds with value added agents, marketing organizations, or other participants in a new economy of provable digital ownership. It's the next step in the evolution of VerusID, the most powerful self-sovereign identity and secure storage model for funds in the digital world.

Verus Vault

With Verus Vault you can now protect funds on a VerusID, even from theft of a private key! If you lock your VerusID with Vault you cannot spend funds from that identity at all until it is again unlocked. While locked, you can still stake those same funds on the Verus network and earn by doing so. Of course, you can also still receive funds.

IT IS IMPORTANT TO NOTE THAT ENABLING REVOCATION, RECOVERY, AND ALL VERUS VAULT CAPABILITIES REQUIRE YOU TO HAVE ONE PRIMARY IDENTITY, AND AT LEAST ONE REVOCATION/RECOVERY ID CONFIGURED.

A locked VerusID can always be revoked and recovered by its revocation and recovery authority identities, which circumvents the lock. At the same time, anyone with only the primary keys, even a multisig of primary keys must first unlock, then wait for the predetermined unlock time before they can spend or access funds. This gives you, or maybe a company that specializes in watching the blockchain to whom you've assigned the revocation ID to revoke and recover whenever an

unauthorized unlock occurs. That means that like a bank, setting a 24 hour unlock delay on your locked IDs actually provides the first decentralized solution to the infamous 5 dollar wrench attack.

In addition to a new level of blockchain protection and decentralized funds recovery, Verus Vault provides the same security for your IDs and NFTs as well as time locks for other purposes, such as vesting schedules, trusts, and inheritance. With Verus Vault, you can now protect and recover your funds, preserving all your assets and generational blockchain wealth from common forms of crypto loss or theft, no bank required.

New Verus Multicurrency, Multichain, DeFi Enabled Testnet

With an easy GUI for basic operations or command line for more advanced functions. Without any programming, you can now create new identities, currencies, liquidity pools, and blockchains for your business, your government, your projects, a worthy cause, your family, or your next decentralized application suite. Send currencies worldwide on the same chain, or across blockchains with ease. Even convert currencies to others on the network without an exchange by sending to yourself and converting along the way.

The new Verus testnet is a full-featured, intrinsically decentralized multi-chain blockchain platform with an unlimited number of identities, currencies, liquidity pools, and blockchains. It is accessible from the released versions of Verus Desktop and Verus CLI wallets, and it is the beginning of a new age in crypto. There are so many things you can do with Verus that you cannot with any other cryptocurrency platform, and you can try them all today.

As Verus PBaaS offers completely new capabilities that go beyond today's decentralized platforms in many fundamental ways, the worldwide Verus community put its energy into creation, rather than convincing everyone that its capabilities are possible. Members across the Verus worldwide community have worked hard to make this all possible, and we are more than excited that you can now experience it firsthand. If you have an interest in the future of crypto, you owe it to yourself to learn about Verus, an unlimited scale, decentralized future with truth and privacy for all.

The Verus testnet, available in the Verus Desktop or cli wallets as the VRSCTEST coin, has the following capabilities, which to our knowledge are unique in crypto today.

Self sovereign, revocable, recoverable identities (currently on mainnet) VerusID

- Enables permissionless registration of friendly name strong identities and funds addresses that are simultaneously fully self-sovereign, revocable, and recoverable.

Staking-capable time locking and theft prevention (Verus Vault)

- Enables identities to be locked, preventing any funds under their control from being spent while locked, but still allowing seamless staking of funds. When locked, a user specifies an unlock delay, typically long enough to notice when someone who might have compromised a user's keys would have to unlock the ID before spending. The only way to circumvent the unlock delay is to revoke and recover an ID. Users may also choose to create and use fresh private keys when unlocking an ID as well. This enables virtually theft proof workflow and

a solution to inheritance, trusts, vesting schedules, the 5\$ wrench attack, and identity theft. IDs may be used as friendly name cryptocurrency addresses for all currencies on all Verus PBaaS blockchains in the Verus network. The VerusID protocol is a protocol, which can also be implemented on non-Verus systems.

Multi-currency, user created, decentralized tokens and merge-mineable, interoperable blockchains without programming

- Enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it. All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. I...

Assets 6

Footer

© 2023 GitHub, Inc.

Footer navigation

- [Terms](#)
- [Privacy](#)
- [Security](#)
- [Status](#)
- [Docs](#)
- Contact GitHub
- [Pricing](#)
- [API](#)
- [Training](#)
- [Blog](#)
- [About](#)

Releases · VerusCoin/VerusCoin · GitHub



Privacy, Community, Power

Verus consensus non emitur

“True consensus cannot be bought” - anonymous

Michael J. Toutonghi

Michael F. Toutonghi

Alex R. English

June 12, 2018

June 12, 2018

Abstract

The Verus Project aims to establish a secure, privacy-centric, fairly-distributed cryptocurrency. But – beyond this currency – Verus seeks to become much more than a zero-knowledge privacy coin, one with two completely new highly-decentralizing proof of work and proof of stake algorithms.

In addition to payments, decentralization, and privacy features, Verus Project plans include its direct use as a currency for provisioning scalable and secure public blockchains as a service (PBaaS), for Verus applications built upon these parallel chains to scale. What this will do is simple: It will enable all people – as well as all nodes on the Verus network to participate in and benefit from a decentralized, blockchain service economy.

This paper details the Verus vision and describes the function of Verus as its own platform, and also as a member of the Komodo platform ecosystem, in the context of its first applications. Verus core applications will provide a foundation to build additional applications and services, which will leverage Verus’ automatically created blockchains, called autochains. Autochains – or PBaaS – will be provisioned and notarized by the Verus blockchain miners and stakers, in exchange for Verus currency.

Autochains will be validated through proof-of-stake by their user populations. In addition to extremely scalable, dynamic, publicly-secured autochain applications, this will add a dynamic isolation and security component to applications that can also create, manage, and verify transactions on the main Verus chain or any other Komodo-compatible, Crypto-Conditions [17], Interledger Protocol [25] enabled blockchains.

The ways we apply this technology to our world – to our biggest contemporary challenges – has the potential to completely remake the fabric of our society.

Preface

Human progress leapt forward with the invention of money. Money enabled worldwide, trade-based economies to move from a primitive, barter-based system to a consensus-based valuation and accounting of verifiable, storable, divisible, and scarce commodities as early currencies of exchange.

Even in recent times, actual or perceived scarcity and authenticity of source, whether genuine gold, an original giant coin of Micronesia [\[27\]](#) – or government backed notes with trusted fiscal management and the ability to exchange for oil – provide the foundation of value upon which, ultimately, human resources are bought and sold.

What blockchain does is straightforward. It enables the creation of cryptographic tools and applications that do nothing less than provide us with a new future of programmable money. This programmable money integrates directly with existent accounting functions and, eventually, with all automated services. Blockchain technologies provide humans with a new, verifiable solution to ensure scarcity, authenticity of source, inbuilt transferability, fungibility through privacy, and programmable rules. More than that, fully peer to peer, public blockchains disintermediate and provide tools that offer the opportunity to revitalize our economic systems and societies, potentially creating a more equitable and honest economic framework for all.

About This Document

This vision paper is not intended as a technical reference, but as a vehicle to communicate our vision, our plans, and our thinking – as we work on realizing the true potential of Verus.

All technologies described are based on a clear understanding of methods intended to implement each solution but – as with any project – the specific implementation details will be refined as we actually realize their functionality. To supplement this vision paper, we will follow soon after its initial release with a white paper, describing the new technologies developed for Verus Coin’s initial launch, what benefits they provide, and provide details of their implementation. We will continue to supplement this vision paper with white papers, as appropriate, for each completed phase of Verus Coin’s development, in the future.

As of this point, we have completed Phase 1, having released Verus Coin with zero knowledge privacy, two brand new algorithms combined for simultaneous CPU-mineable proof of work (POW) and fair proof of stake (POS) consensus, in-wallet mining and staking on PCs, leverage of and support for advanced Komodo platform technologies, and wallets and miners for every major PC operating system.

1 Introduction

Information vies for our attention in today's digital world, trying to convince us of what product to use or which politician to believe. We are watched and measured as we react to that information, in order to convince us what to buy or to believe. This is an active process that exploits our lack of online privacy – combined with weaknesses in human psychology – in order to make profit; extracting value from, even affecting our behavior, learning about and influencing each of us individually to open our wallets or add our voices to another's agenda.

No single person can digest and fully understand – let alone verify – a fraction of the information thrust at us in daily life. We are told to give up on the notion of privacy and to trust networks of companies with our most private data, our identity, our credit histories, our location, our habits. We are also told that our voices are lost in the digital sea of information. Yet how is this the same sea through which we are laser-targeted based on correlating our behaviors to learn so much about us, individually? What if these technologies could be turned towards the benefit of society, first – and then allowed to support businesses, in that context?

We on the Verus team believe it **is** possible to support businesses and governments requirements with digital systems that: 1) respect your privacy, 2) give you control over your data, and 3) enable you to speak your mind with the anonymous authority of an authorized voter or member of a community, in a way that can directly be heard and affect actual change.

Before we explain how, it would help to understand the shared beliefs behind our vision for Verus. These beliefs underpin everything we build into the Verus network:

1. We believe that every human has ideas, knowledge, and value to contribute to our society. By using technology to reward people for their contributions, we can enable each of our verifiable, yet anonymous voices to be heard as a collective truth.
2. We believe that those who contribute positively over time to the system should be rewarded for that contribution and provided with ongoing incentive.
3. We believe that a world-scale, peer-to-peer system that can enable humans to be queried directly, confidentially, verifiably – and in a transparent manner – can directly provide populations of people and the world with valuable, transformative tools.

With Verus, we will introduce digital tools to enable us all to build a better world together. We will monetarily incentivize – with our technologies – behavior that strengthens communities and institutions. This is the thing that's missing in the online world today: Fiscal incentive for communal behavior built into the very fabric of its function.

The Verus Project's tools will make it easy to create an identity – or multiple identities – on the Verus multi-blockchain system, which can accumulate value and even have multiple personas, each to represent a facet of your identity as a whole. This reflects how we might express identity in our personal or professional lives, where some situations call for provable credibility, yet others require no more information than what you might reveal when encountering a casual stranger.

Each identity will have its own, unique address. Unless its owner reveals information to link two or more identities, it is cryptographically hard – meaning virtually impossible – to correlate one identity to another. At the same time, the owner of an identity can still cryptographically verify statements made about identities under their control, attesting to identity properties (such as passport, age, height, citizenship, photo, etc.) as strongly as is possible with today’s digital technology. As part of the Verus vision, to be described in more detail in later phases, we intend to support fully decentralized verification of identities that can provide as strong verification as today’s centralized systems. At the same time, we believe it is important for practical reasons to enable compatibility with centralized forms of identity and to enable people to optionally support KYC in identities. To enable a smooth bridge between centralized and decentralized identity systems, today’s ID systems, including biometric and government issued IDs, will be supported via centralized or decentralized verification to enable use of Verus identity in situations that require conformance to know-your-customer (KYC) regulations. These forms of ID, however, are not required to establish or use even strong, decentralized identities on the Verus network.

Verus autochains, will operate parallel to the main Verus chain and enable large-scale applications – such as polls or elections – to run simultaneously without concern for congestion or excessive fees. Autochains will enable poll application users to provision their own secure blockchains just by using the application – spawning dynamic parallel chains that can process thousands (or potentially millions) of transactions per second when needed. Autochains will operate by proof of stake, enabling each chain to have security isolated to its direct user population. Autochains will also be backed by Verus notarization and block time synchronization – providing the full weight of its PoW/PoS security layer as well as the Komodo platform’s delayed proof of work (dPoW), to provide notarization all the way back to the full power of the actual Bitcoin blockchain.

For **Phase II**, what we expect to be an extended development phase, we will work to implement autochains and their first application in the world. We will eventually use them to create, secure, scale, and perform polls for everything from classifying online content, to identifying real public opinion, to actual, real-time elections for an organization or – conceivably – a government.

Our goal is to make these polls easy to use from a PC or mobile device, yet industrial strength and suitable for serious, secure elections. They will leverage the latest cryptographic technologies for privacy – known as zero-knowledge proofs – to preserve confidentiality.

They will be:

- **Confidential** — No one but the voter knows who or what they voted for – unless the voter discloses. Results of the vote can be withheld until the vote is complete, at which time they can be released to everyone, simultaneously.
- **Verifiable** — Only voters that are authorized to vote can vote. Each voter can vote only once. Each voter can look at the released results and see that their vote was counted.
- **Transparent** — Anyone can validate the number of votes counted, and the number of votes, each person or selection received.

- **Secure** — Our novel autochains – dynamic, security isolated, proof-of-stake (PoS) blockchains – are layered over proof-of-work (PoW) and delayed proof-of-work (dPoW) security, all the way back to the real Bitcoin blockchain through Komodo.

By default, the first layer of security is run by the actual voters, themselves. Together, this provides unprecedented layered security both in the autochain and in delayed proof-of-work, leveraging the network with the most hash power in the world.

We like to think that – on a Verus autochain – a 51% attack is called winning a poll.

- **Scalable** — Each poll is conducted on its own, automatically-created-and-validated blockchain, operating under its own visible validation rules.

2 Our Vision for Verus Foundational Applications

Throughout history, understanding what a population truly thinks or feels has been an invaluable capability. The manipulation of that understanding can, and has, repeatedly changed the course of history.

The Verus Project plans to use highly decentralized blockchain technology to enable all people to safely, anonymously, and confidently express their opinions – on any issue. Verus users will be able to share their knowledge in a public forum, query a population of humans (or eventually both AIs and humans), quickly and effectively, and – importantly – earn cryptocurrency in exchange for these contributions to collective knowledge.

A primary goal of the Verus Project is to enable societies and organizations to make decisions based on a more honest understanding of the public’s actual beliefs. To achieve this goal, we consider the need for both confidentiality and transparency, to ensure that – first and foremost – the system can be trusted. It is also critical to prioritize decentralization and develop a core platform that can leverage fully transparent blockchain technologies, while running many applications that respect privacy. With Verus, these applications can leverage zero-knowledge succinct non-interactive arguments of knowledge – or zk-SNARKs – the most reliable and tested iteration of proven, zero-knowledge privacy technology available today.

Our novel approach ensures that this technology can be used at scale, across any population size, throughout the world. Using the tested-at-scale infrastructure and tools of the Komodo platform, and Verus planned development of PBaaS – the Verus developers will build human organizational tools, including voting and fully self-sovereign identity with a reputation system that respects privacy. The goal, here, is a significant one, and these tools are planned in a way to allow entire human populations to transparently and directly share their knowledge and opinions – without the risk of privacy violation, spin or censorship.

Verus tools will also leverage human (and eventually non-human) intelligence at scale to solve previously challenging problems in a decentralized manner – financially rewarding those who contribute to their proper function.

To serve all people in the world simultaneously with these public, peer-to-peer tools, Verus introduces autochains. Autochains are a novel scaling model for blockchain systems. Autochains will enable full node miners to provide public blockchain provisioning services. These services will occur in transient, parallel chains, chains that are isolated from congestion, disruption, or interference from the main Verus blockchain. As an autochain operates, it is

almost completely parallel and isolated from the main Verus chain, except for the capability of posting transactions and proofs to Verus, for results and coordination across applications or people.

By combining privacy, polls at scale, and identity, Verus can be used to post polls and ask almost any question of a population or subpopulation of people, receive an honest – and, importantly – verifiable answer, and still respect the privacy of the respondents. When even the first support for polling is active on the Verus network, the Verus community will be able to vote on the long term direction of the Verus Project, itself.

Over time, Verus polls – used for purposes such as determining the accuracy of news online, for research, or even for political polling, will be combined with machine learning to enable all of us, as a society, to benefit directly from our collective intelligence without today's risks to our individual privacy. Verus will combine auditability and verifiability of transactions with the option of verified, confidential participation – meaning societies and organizations can directly understand what people think, feel, and believe, as populations – without taking control of their personal data or targeting some individuals, based on their answers or beliefs.

Ultimately, our vision for the Verus Project is to enable us all to directly participate in our own worldwide economy. Verus applications will enable any individual to speak up with the power of an authorized participant – and with the confidence of anonymity – to produce a verifiable, transparent, and honest flow of information throughout the globe. It will transform the way that anonymity functions in our current technological environment – prioritizing cooperation and providing the financial incentive for that cooperation to occur.

3 Important Verus Concepts

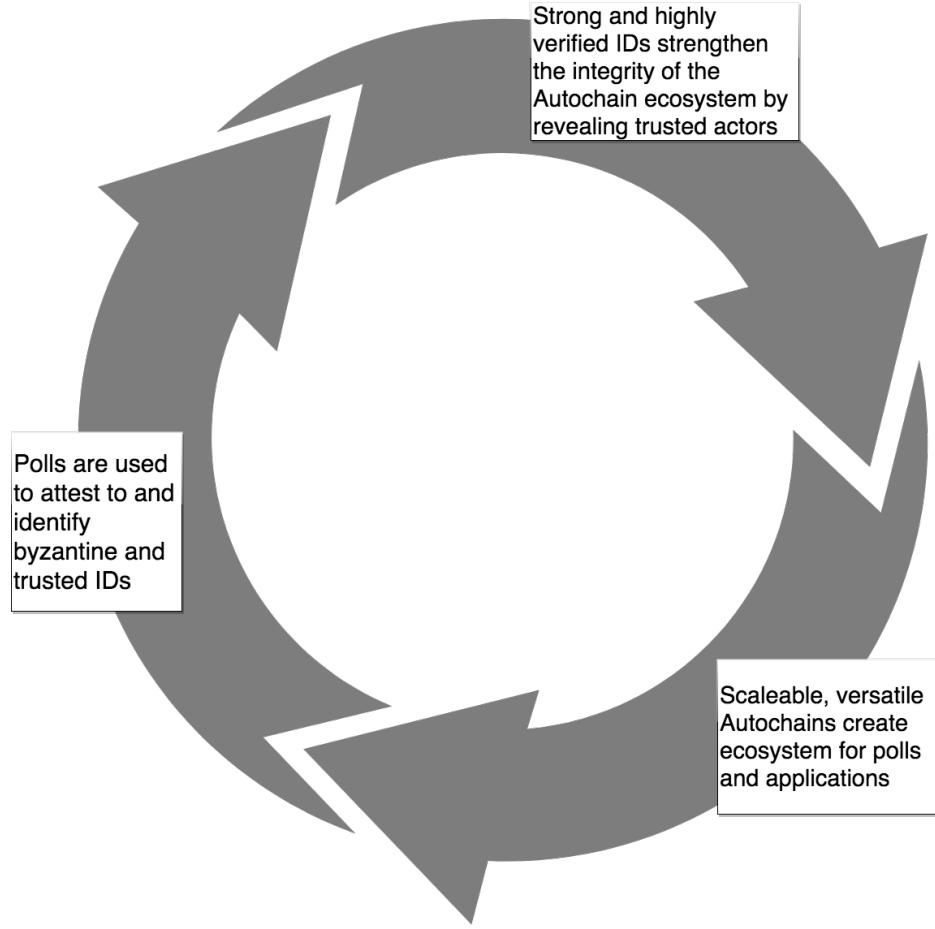
The longer term Verus Project vision of a better society through public blockchain services relies upon three fundamental pillars –where the correct operation of each strengthens the others in the form of a virtuous circle.

These pillars are:

1. Extreme transaction throughput and scalable decentralized applications with autochains, which provide public blockchains as a service (PBaaS)
2. Selectively strong, private identity, and
3. Open application support with a foundation of polls, voting, and lottery selection services.

Each of these pillars will help us process human knowledge, understanding, and/or opinion on any content or topic. Together, they will enable direct querying – with confidential and truthful – answers across an entire population. It will create a secure, public platform that respects privacy, and can potentially serve as the foundation of a more respectful society.

It is also important to know about the the concept of **Verus Virtue** when reading further in this document. The best way to think about Virtue for the purposes of this document is as a separate currency that can be earned, but not purchased or sold, and is part of the



evolution of the Verus proof-of-stake algorithm, slated for later phases of the Verus project. Proof of Virtue is a technology that will be described in much greater detail and implemented in later phases of the Verus project.

3.1 Autochains

With Verus, we plan to create a platform upon which we will build voting and even election systems that support our vision of enabling a better society through blockchain.

To run simultaneous polls across disparate populations on one blockchain-based system, we need another dimension of information – one that retains privacy, and is anchored to a primary store of value, Verus.

One approach to creating another dimension of meaning would be a system of “coloring.” The Ethereum blockchain system and some layers over Bitcoin or Bitcoin compatible coins use a version of a coloring system. [\[5\]](#) [\[6\]](#) [\[7\]](#) [\[8\]](#). While, at first glance, this seems like a reasonable approach, it is actually a suboptimal solution, at best.

With Ethereum and the ICO experiences of 2017 [\[9\]](#) [\[10\]](#), we now know what happens when you create one blockchain and overload it with transactions: Congestion and unnecessarily high fees. Developers have proposed many solutions to this problem of scaling – including a move to proof-of-stake systems and sharding (effectively trying to separate the various

functions on a blockchain), or even a move to large systems of parallel chains with a common design.

Yet instead of implementing a “solution” with colored coins sharing one blockchain, we have decided to take a completely new approach to this problem. Our approach will not only provide an extra dimension of information and an unlimited number of token types, or “colors”, but it will do so with highly scalable PBaaS autochains, which also reward Verus miners and node operators as provisioning agents. This is a novel method of scaling, one which provides significant scale and security benefits through isolation of transaction processing for each application instance on its own blockchain.

Rather than thinking of Verus as a single blockchain, it is more appropriate to think of it as a blockchain-based system, one that is rooted in a primary value chain.

Autochains are exactly what they sound like – automatically instantiated blockchains that are validated initially – and when notarizing – by Verus miners, and otherwise, only by their users, often a disjoint set from the whole of Verus users. Autochains, once instantiated, can be used indefinitely or can have a finite lifetime to be used for a specific purpose. In this document, we will often discuss autochains in the context of a few concrete applications that we intend to implement on the Verus network. It is important to keep in mind that their versatility is **basically unlimited**. Due to their security being inherited from the main Verus chain, autochains eliminate perhaps the biggest disadvantage that creating consensus-based distributed ledger applications suffers from: Weak defense against attacks in their earliest stages.

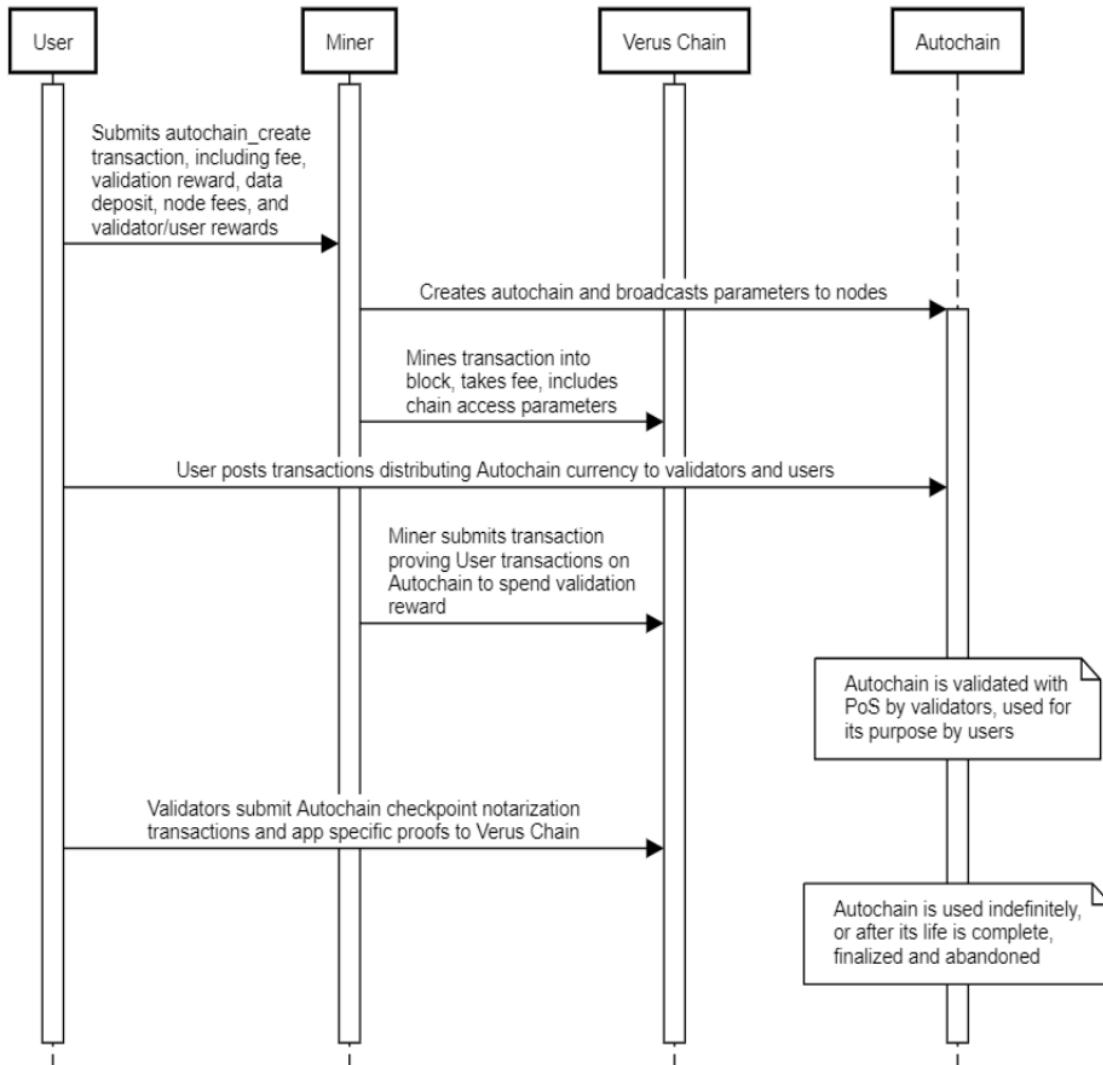
In **Phase II**, Verus will define an autochain provisioning protocol, and specific voting protocols, based on blockchain transactions. These protocols will include zero knowledge proofs, Crypto Conditions [17], and Interledger Protocol [25] technologies to provide proper incentive for all participants to engage in the automatic creation and use of on-demand, isolated or public blockchains, ones that rest on a secure, public foundation.

The autochain protocol will also provide the ability to place a value on the data generated by the autochain’s operation. This value will then either be released to the public or paid out, through rewards, to the participants in the creation of that data. This creates a model that incentivizes release of learning and information to the public – also making applications which do so much less expensive to run.

For Verus poll applications, validation will default to a form of proof-of-stake (PoS), where all members of the chain are stakers. This is because that security model matches exactly the interests of the majority of voters in having an accurate record. Because PoS does not involve powerful mining equipment or competition to solve a cryptographic puzzle, it can operate on much less powerful hardware – even on mobile devices. Since voters, themselves, secure the chain simply by running the voting software, poll chains are generally not affected by activity on the main Verus chain (except when posting interledger transactions). In addition, an autochain’s operation, independent of scale, adds no overhead or congestion to the main Verus chain’s transaction load – besides the transactions to provision the poll chain and post results back to the main chain itself.

Since we want to be able to create autochains from clients that may not be running full nodes or even own stable IP addresses, full nodes will be incentivized to provide reliable static node support for autochains. Once they provide this support, they will be recognized

Verus Autochain Lifecycle



by validators on the autochain. They will then be able to use this recognition to boost their reputation for selection in future lotteries.

By advertising autochain node support on the main blockchain – along with a stable payment address via a transaction – the node IP can be selected in a random lottery. The Eigentrust ratings of all nodes can then be recorded by all other nodes that serve the chain at regular intervals. Nodes which consistently offer better QoS will get better ratings. Nodes that get poorly rated will still share in the node rewards for the current blockchain – but will be less likely to be selected in the next node lottery.

3.2 The Virtue Autochain

3.2.1 Roots of Trust in Distributed Consensus

The Bitcoin [14] proof of work implementation introduced the world to systems that establish trust, not due to belief in any individual human's behavior, but based on the tendency of humans to act in their own self-interest.

The Bitcoin protocol rewards winners of a contest, referred to as "miners", in statistical proportion to the computing power they employ to "mine" for Bitcoin, as compared to the total computing power of all miners. For every block of validated transactions added to the Bitcoin blockchain, someone wins a competition to earn the right to define that block while adhering to specific, verifiable rules. The competition involves verifiably solving a statistical mathematical puzzle that is specific to exactly that block, meaning the same work cannot be reused on any other block. The winner of each competition earns the right to accurately process the next block of pending Bitcoin transactions and to claim a specific measure of Bitcoin, called the "block reward" (plus fees of all transactions in the block that was processed).

Mining competition serves to determine the amount of computing power a would-be attacker would have to control in order to mine blocks that were not earned according to the intended rules, execute transactions that might otherwise be rejected, and prevent certain transactions from executing altogether. Based on the way that miners achieve consensus on what is the correct chain, forging a false Bitcoin chain to achieve this would require an attacker to control more than 50% of the total power of all miners – both honest and byzantine.

Since, right now, it is likely infeasible for any single actor to mount such an attack against Bitcoin, the largest network of hash power today, this type of competition, called proof-of-work (PoW), currently manages hundreds of billions of dollars of value, sends transactions of that value to anyone, anywhere, at any time, and does this all without any company, bank, or trusted third party, of any kind.

In addition to enabling secure management of large sums of money, PoW has also sparked a mining arms race and significant investment in computing power to claim Bitcoins and other cryptocurrencies, creating the equivalent of the world's largest distributed supercomputer doing nothing but the same calculations, albeit on different data, over and over again. As a result, one side effect of the public blockchains' PoW security system is massive power consumption and significant ecological cost.

Due to this, a great deal of research and development has gone into alternative consensus mechanisms [11] [12] [13]. Most serious PoW alternatives center around the concept of proof-of-stake (PoS) – the idea, that by putting something at stake commensurate with the value being transacted in the transaction flow, a rational staking person, known as a "validator," will choose good behavior within the ecosystem, so as not to lose their "stake."

Even though large systems are being built that depend on 100% PoS, there remains controversy over its robustness when users have incompatible interests, or when an attacker's stake is not valued as highly as the perceived value to be gained by corrupting the system.

Most modern PoS and PoW systems make a **fundamental assumption** about all miners; that they are either 'byzantine', and intend to compromise the system as a whole, or

‘rational’, in statistically the same proportions for all participants equally. One need only consider that if there was a way to learn a more accurate statistical function for each participant’s probability of being **either** byzantine or rational, using such a function to determine who participates in the system’s validation would improve its resistance to attack.

In fact, any accurate method of recognizing those who were attempting to strengthen the system – and thus who could statistically be more trusted than a byzantine or even average participant – could both decrease the power consumption and ecological cost of blockchain security and further strengthen resistance to attack. This then brings up the question: **“How do we recognize trustworthy participants?”**

Some methods studied and proven effective are the Eigentrust reputation management algorithm [20], and its improved derivative the Eigentrust++ mechanism implemented in the NEM blockchain [22]. Originally designed by Sep Kamvar, Mario Glosser and Hector Garcia-Molina in 2003, the Eigentrust algorithm is built to function on peer-to-peer networks. It aims to isolate byzantine actors in said networks by assigning each peer a public, global trust value based on their history of activity – clearly displaying a form of “rating” for each peer. Those with lower ratings are shown not to be trustworthy, and thus, are interacted with less by their peers.

Simply put, the algorithm assumes that if any given peer a trusts any given peer b , then it would also trust the peers trusted by b . Every peer calculates a local trust value for each other peer it has interacted with, based on the either satisfactory or unsatisfactory transactions it has had with each one. These local trust values are determined by each peer, and when peer a wants to know if they should trust peer b , say, before making a transaction, peer a would ask all other peers it knows to report on their local trust values of peer b and weigh their responses according to the trust values peer a has for each of them.

3.2.2 Proof of Virtue (PoV) Reward System

Using a model similar to the EigenTrust algorithm, Verus will introduce the idea of Proof of Virtue (PoV) **enhancement** to the PoS algorithm initially released in **Phase I**. In PoV, we intend to weight staking contests with both Verus stake and “Virtue”, a special “currency”, expected to be tracked on a Virtue autochain, and used as a **trust rating** of identities within the network.

In a similar way to the “amount of stake” used when staking, Virtue will add another component to the probability of being selected to process a block of transactions on the Verus Network. Its probabilities upon introduction will not change returns on the Verus PoS system, but addresses will gradually be able to improve earnings with a Virtue weighting – **based on recorded activity and behavior**.

In order to ensure that Virtue is both a rare and valued property, which drives correct behavior, it will neither be purchased, nor sold. Verus holders will have opportunities to earn a small measure of Virtue, which can then be further increased by staking the Virtue itself, when attesting to a fact for the network, validating information, or providing another measurable benefit to the network.

Since Virtue will have intrinsic value on the Verus network, there will be methods for transferring it through wallet ownership – for cases of probate or other necessity – but such behavior as a tool for trading in virtue will not provide a secure or intended method of

exchange.

The Proof of Virtue model will effectively be a **modified** form of PoS that is not based just on monetary value, but also on long term measurement of contribution, which will affect trustworthiness and earning power in the Verus network. The intent is to provide opportunities for more people to participate in the growth of a **positive functioning network** and to prevent potential attackers from being able to easily buy their way into an attack, adding yet another layer of security.

We expect to keep the PoW component as part of Verus security system for an indefinite period of time. We also expect to leverage the dPoW security of Komodo and its notarization into the Bitcoin blockchain as well.

3.3 Polls, Voting, and Elections

3.3.1 The Importance of Secure Polling

A great deal of research has delved into the best way to achieve confidential, verifiable, and transparent (CVT) elections electronically – with a few notable systems actually used in real life situations. These attempts, however, have been plagued with significant limitations, limitations that almost always risked either revealing the identity of a voter and thus eliminating confidentiality, or allowing attackers to impersonate legitimate voters, and put the validity of the vote into question [1].

This was shown in one of the first majorly adopted North American electronic voting systems, the Diebold AccuVote TS, which was announced in the year 2000 [4], at a time when, as a result of the Florida 2000 presidential election, the general public began to recognize numerous flaws in the widely-used punch card voting system [2]. Despite multiple studies discussing electronic voting systems – studies which clearly warned of security risks such as insider threats, issues with auditing, and network vulnerabilities – by 2004, the Accuvote TS system was deployed for major political elections in 37 US states [2], resulting in multiple serious problems that impact the legitimacy of numerous election results since, due to inherent flaws in its design that were never fully addressed [3].

Firstly, the “solution” that the system introduced to deal with voters submitting multiple ballots – and to solve the ballot anonymity problem – was to issue a single personal voting ‘smartcard’ to each vote. This smartcard, unfortunately, did not contain a complete identity, but instead contained a common election key among voters that voting machines simply checked to determine if the card belonged in the correct election [2]. The lack of any cryptographic unique identification on these cards was a significant security flaw, as user-programmable smart cards and readers almost instantly became commercially available on the internet for reasonable prices – making it extremely easy to mass produce homemade copies. Furthermore, due to the lack of a truly secure boot loader, operating system, or application, the system had numerous potential attack vectors, many of which were quite simple for an adversary. For example, any attacker with access to the operating system had the ability to modify ballot program files through the standard Windows Explorer application already included on each machine.

These serious failures in previously adopted electronic voting systems have literally **changed the course of history**, affecting the outcomes of major political elections that

decide the state of world politics. This highlights the importance of creating a system that is truly confidential, verifiable and transparent, and indicates the magnitude of one problem we plan to solve with Verus.

3.3.2 Running, Recording, and Scaling Polls - Transient, PoS Chains

Using autochains, Verus will dynamically create parallel voting chains on demand for each poll in its ecosystem. This creates improved, isolated security and allows for scaling in a virtually unlimited manner.

The Verus wallet, which will be used for voting, will also have the ability to validate blocks through proof-of-stake, blocks which, when enabled, will allow voters who are validating the election to earn rewards. These rewards will either be paid in Virtue for intrinsic polls, when available, or from outputs of the payment transaction that instantiated the poll. When a poll is complete, results are posted back to the main Verus chain. The chain used to run it can then be archived or deleted, and it is typically abandoned.

3.3.3 Poll Content and Distributed Hash Tables

While blockchains make excellent databases for permanent, non-repudiable records and public key management, they are not the best solution for storing large amounts of data, due to their massive duplication across the distributed network. Instead, a better model for distributed storage would look something like the Interplanetary File System, or IPFS, which is a peer-to-peer file system based on the Kademlia [18]19 protocol in an implementation that currently works well enough to use as storage for create/read/delete operations using hash commitments on the Verus blockchain to represent and index content.

While we are evaluating the IPFS, we are also looking at other distributed hash table solutions, such as Open DHT [26]. A key requirement for the Verus network is that mobile devices be able to participate in its operation. Open DHT is an efficient C++ library intended for use with small devices. IPFS does not yet seem to have much support for mobile at this time, though it does have a Javascript library that can be used from the browser or mobile applications [24].

Until IPFS is supplanted with another implementation, Verus applications will assume that when off-chain content is needed, it should be retrieved from IPFS, and for future compatibility, we will add a storage and version specifier. In order to store supporting poll or other content for dApps, the content owner must ensure that the content is pinned in IPFS, until it is no longer needed. By **Phase IV**, or earlier if there is significant demand, we expect to either support or provide a blockchain-metered storage solution that is more integrated and automatic to use for poll makers. Two options for such a system include a cross-blockchain integration with a decentralized payment/pinning solution, or to offer a simple service that charges in Verus to store and automatically pin content for specific lengths of time in IPFS.

3.3.4 Voting Models and Types of Polls

Verus will support numerous types of polls, those that use weighted or unweighted voting, that seek to expose truth or opinions, and polls that rate and choose collectively. In order to

create a sensible taxonomy for polls, one that people will understand, we define the following types of voting:

1. Multiple choice
2. Weighted multiple choice
3. Ranking
4. Rating

We also define the following types of polls:

1. Polls to classify
2. Polls to select
3. Polls to rank
4. Polls to rate

We do not expect Verus to support all voting and poll types in the early phases. In fact, until **Phase IV**, the Verus core team plans to focus on known requirements of multiple choice and weighted multiple choice voting. During or after completion of **Phase IV**, expanding support for voting and poll types will be higher priority.

Each type of Verus poll may support confidential or public voting, live or delayed poll results, online or offline vote authorization, and lottery selected vote authorization. We will roll out working vote models in phases, as well, initially providing solutions for weighted and unweighted selection voting, which fits well for both political and opinion polls as well as content classification and ID verification.

3.3.5 Intrinsic Polls

Once Virtue is activated, the Verus network itself will run specific, intrinsic polls that will be regularly available to identities with Virtue, based on a statistical lottery and weighted by their actual measure of Virtue. Participating in these polls, which will be used to classify content and perform basic, human verification functions, enables participants to help the system classify and rate content across Verus applications, enabling a rating system rather than censorship to allow each voter to set preferences for the ratings of polls they may wish to see.

Miners who provision autochains for these intrinsic polls will get the same rewards as miners setting up autochains for commissioned polls. The content for these polls will be taken from other polls and will be used to classify and rate those polls for easier discoverability and the user experience of poll services.

3.3.6 Privacy

There is a fair amount of confusion among the public about privacy on the blockchain. One common belief is that Bitcoin transactions are private, when in fact all Bitcoin transactions, as well as the public addresses between which value is transferred, are public information [21]. As with the Verus token and its blockchain, poll autochains will also support zk-SNARKs to ensure that votes which have not been disclosed by a voter remain provably anonymous to all other parties, while still being verifiable by the person who holds the keys to the address that cast them. Currently, based on standard smartphone hardware specifications and the memory requirements of zk-SNARK transactions, mobile phones cannot generate z-transactions. In coming releases of new Zcash technology this year, the z-transactions will be more efficient and still provide the same privacy guarantees. When these technologies are available, we will work to support them on the Verus chain, as well as use the increased efficiency to enable z-transactions and all Verus features on mobile devices as soon as possible. Until that is available, we will work to provide a mobile wallet for Verus transparent transactions.

3.3.7 Content Classification

While Verus is designed to provide invaluable tools to people across the world, any system without censorship will also inevitably allow the underbelly of humanity to show through. That means that while we would want all people to participate, we recognize that in order for that to provide the most positive experience for all participants, we must use the Verus poll system itself to recognize, rate, and classify the content on its network, enabling those who provide such a service through participation in polls to earn, while enabling the selective filtering of content based on value judgements and classification of honest populations.

While we recognize that the best early use case for such capability is in rating and classifying non-intrinsic polls themselves, the ability to classify content is a fundamental strength that Verus will increasingly develop over time. At first, we expect such classification to be applied to relatively simple tasks for humans, such as identifying toxic comments, hate speech, or categories of information. Even though these types of classification systems have largely employed machine learning systems running against privately curated training data, our vision is for Verus content classification to enable people to earn, as they classify content better than any machine learning system, but in a way that can be followed by the best machine learning algorithms, and generalized at scale.

3.3.8 Truth vs. Opinion

When classifying content, especially when you start to consider challenging classifications, such as misleading, propaganda, generally accepted fact, accurate versions of history, vs. classification of what should generally be accepted as toxic or rated content that people could reasonably answer objectively, one must consider the difference between fact and opinion. Selecting the winner of an election is a matter of expressing an opinion. Classifying content according to level of toxicity is a question of determining the factual answer to the question: what does the majority of the voting population believe the classification to be. Determining

whether an image in one photo is the same image as that from another is a question with an objective, if not determinable answer.

To ensure that Verus polls support polls that attempt to discover accepted truths, facts, and even credible emerging arguments, Verus voters will often be able to stake and either earn or lose Verus and in later phases, Virtue, by answering in a manner that matches consensus. For other polls, which are recognized as opinion polls, all answers are equally valid contributions.

3.4 Random Sampling as a Service

The Verus system will randomized selection for many of its functions, similar to its **Phase I** PoS block validation system. Blockchain lotteries will be generalized and used for selection of participants in randomized polls and in other cases where pseudo-random sampling is desired. The general principles behind Verus lotteries are the same, regardless of whether something similar to Algorand [29] for guaranteed selection or an original Verus algorithm based on difficulty, such as Verus PoS is used for selection. Lotteries use the blockchain as a random oracle and are based on the assumption that the exact value of a specific block hash from a past block in the chain cannot be modified by a byzantine participant to weigh the odds in their favor.

3.4.1 Participating in a Lottery

Since the Verus network is completely decentralized, all lotteries, including PoS, are potential, and there is no central server selecting and sending messages to those who qualified in a lottery. When lotteries are implemented as a general feature, if you would like to participate in any Verus lottery or poll that is distributing tickets via lottery, you must first determine that you qualify for the poll's requirements. You will then need to submit a transaction that proves that you have a valid claim and spend the output transaction of the lottery ticket to your address using that proof. By default, that spend will authorize you to be a validator on the autochain for the poll you are participating in, by providing you with the currency of that poll. You may validate the poll by leaving your wallet running, getting potential rewards in Verus or Virtue for both activities.

Lotteries will be useful for selecting subpopulations based on identity and claims people make about themselves or that others claim about them, which they are willing to share. While detailed discussion of Verus identity is beyond the scope of this paper's release, the Verus system will provide self-sovereign identity with the ability for identities to make very flexible claims and have them attested to by other identities. For example, a lottery may look for males between the ages of 18 to 25 among identities willing to share that information, which will even be possible to verify as strongly as through a validated passport or driver's license. Other polls may not require the same evidence backing claims. Polls may also have restricted voting, such as local political polls, where verification and authorization for the poll must be provided explicitly, often by mail to a physical address or in person in the form of a QR code or electronically delivered transaction to your Verus wallet.

3.4.2 Proof of Stake Lottery

A block validation lottery is a somewhat different form of lottery which allows you to claim the right to validate a block and its associated reward based on proof-of-stake (for autochains) or 50/50 PoW/PoS (for the main chain). The lottery requires that a prior block hash 100 blocks past combined with a qualifying UTXO of the destination public key hashed with the staking block height and then divided by the UTXO value must be under the current proof of stake validation difficulty. By dividing by the UTXO value, Verus weights proof-of-stake or proof-of-virtue in proportion to the amount of stake or virtue in the UTXO. If no one who qualifies to validate a block is online or no one responds with a submitted block validation in a certain amount of time, the blockchain will wait until someone mines a block using proof-of-work.

3.4.3 Ticket Harvesting Lotteries

There will be multiple ways to acquire tickets to participate in any particular poll. The manual ways emphasized so far may include receiving a poll with QR code in post or email or perhaps directly from someone taking a poll. In addition to these direct ways to receive a voting ticket, Verus will enable polls to be posted on the poll blockchain as transactions, the tickets for which are distributed from its outputs by lottery. This will work when the Verus ID functionality is available, and will enable polls to require presence of specific claims or attestations on an ID, as well as the number of participants to select in the poll by creating one transaction output for each, which is spendable by a cryptocondition [17] that defines the lottery conditions and random difficulty. The difficulty determines how often a particular block will match any attempt to harvest a ticket from that block.

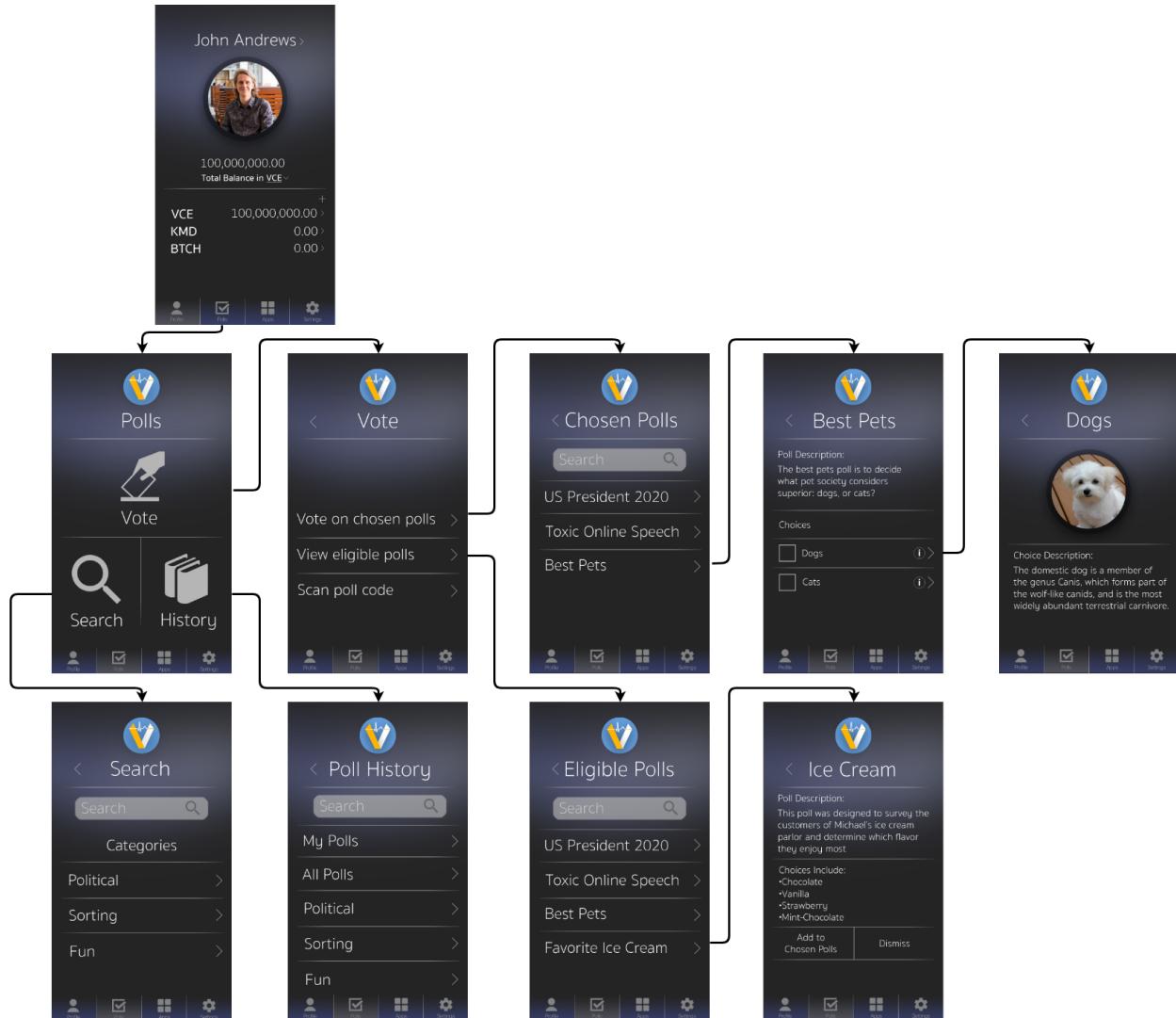
3.5 Machine Learning Integration

Verus enables humans to share and expose human knowledge through voting mechanisms to make decisions, express opinions, and classify accurately. The data used for that classification, as well as the classifications or decisions themselves provide an ideal source of training data for machine learning systems across any domain of human knowledge or opinion. As corporations gather, sequester, and learn from massive amounts of personal data in order to compete and boost their bottom line, rather than accrue to society's benefit, these massive private databases serve to affect and influence human populations by knowing more about them than others, or even than they know about themselves.

Since Verus enables humans to use voting mechanisms to classify and make decisions, and stores the results of this process in IPFS and on the blockchain, we will train machine learning algorithms on these results over time, allowing them to eventually classify and make decisions consistent with human decisions and values at an even higher scale. This will enable Verus to perform certain tasks automatically, such as the recognition of false or incorrect identification data among participants in the system, the initial classification or rating of certain content that can be overruled, but may not be appropriate for all audiences, and much more. The end result of this is a more secure and trustworthy network, built on consensus and trustless principles, leveraging worldwide human knowledge as a base of information and machines for scale.

A major Verus advantage over almost any other system in its use of Machine learning on human data is the innate privacy it provides to those whose data is used for learning, along with the default that all resulting information be made public, unless those making polls pay the Verus community a going rate to keep the data for themselves. In today's society, the goal of most machine learning systems is to match consumers with products they are likely to buy and/or manipulate them into an actual purchase, thus generating revenue. This creates a situation that can create unintended negative consequences when algorithms disregard any positive or negative effects its predictive abilities may have on society [23]. For example, if the algorithm recognizes that people with depressive episodes are more likely to gamble, and thus buy airline tickets to gambling-oriented locations, such as Las Vegas, it will advertise those airline tickets to those people. In the Verus system, machine learning algorithms will be able to learn from human beings, while being limited to accessing only the data users permit, and at the same time, being unable to easily target any specific individual.

3.6 Verus Mobile Polls



In addition to a desktop wallet that supports Verus and other Komodo platforms or compatible applications, mobile support is a high priority and will provide yet another layer of convenience for users, enabling them to use, earn, and spend Verus in everyday life. We intend to provide a Verus mobile experience that functions as a wallet, self-sovereign identity, provides easy access to the polling and earning applications, and is intended to serve as an extensible application browser, capable of supporting additional applications that leverage the Komodo or compatible ecosystems. While we have already begun thinking about and storyboarding design and development of user experiences, we do not expect any mobile experience to be ready until completion of **Phase II** at the earliest. Even when the first mobile experiences are ready, they will not include the support for private transactions that we intend to enable as a foundation for confidential, mobile polling and communications. To get a feel for the way we envision Verus experiences working in practice, below is an example of some poll screens from early designs.

4 Implementation and Roadmap

After researching options from building the technology ourselves to leveraging unreleased advanced projects, to using one of the few well thought through blockchain application platforms, we decided to start building the Verus project as a friendly fork of Komodo and its asset chain technology, enabling us to both become a contributing member of and also enhance and extend the Komodo platform as Verus builds blockchain platform technologies and real world applications of PBaaS.

By leveraging proven zk-SNARK zero knowledge privacy technologies and Komodo's delayed proof of work (dPoW), which notarizes the main Verus blockchain into Komodo's blockchain, which is then notarized into the Bitcoin blockchain, we underpin Verus with a foundation of state-of-the-art privacy, security, and interledger transaction capabilities as our baseline. Above that, we have already developed advanced features that further enhance security, improve decentralization, and prepare for the implementation of autochains and proof of virtue.

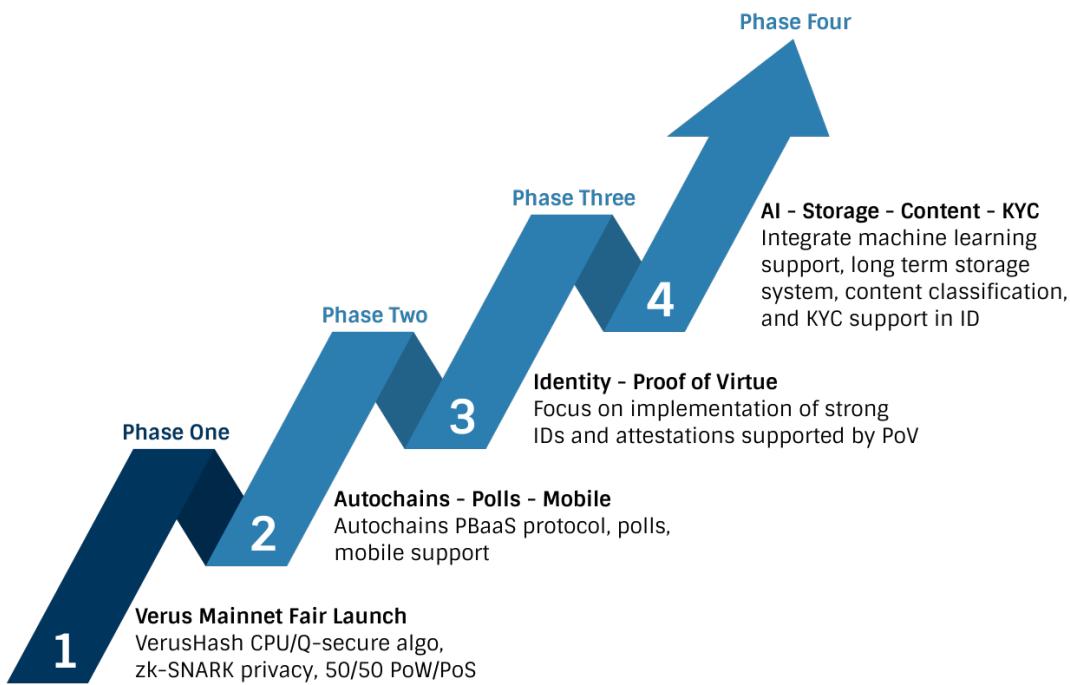
We will maintain our own open source fork of the Komodo platform and collaborate with the Komodo team when possible to develop new features or capabilities that can be accessed cross-chain by projects across the Komodo platform and even other blockchains that are simply compatible with Komodo's Interledger Protocol implementations. We will maintain our development as an open source project and contribute back to the broader open source community, as we leverage the contributions of others that have made it possible for us to begin realizing our vision as a community.

All that said, roadmaps are never perfect. Unless we significantly underpromise and prioritize a date so highly that no feature or capability is important in and of itself, targeting timeframes and milestones without specific target dates allows us to be ambitious in our goals and adapt to a changing world as we deliver. Even in **Phase I**, which we have delivered as of the writing of this paper, we adjusted our plan, and in an uncharacteristic turn of events, delivered even more than we had originally planned. We do not expect that to always be the case, and we will focus more on correct and complete phases than specific dates when possible. Sometimes, as with any major endeavor, the right choice is to recognize an

unexpected opportunity or obstacle, take two steps to the right or left, and only then proceed forward.

For the purpose of this whitepaper, we will discuss four phases of the Verus vision. We do not see these four phases as a completion of the vision as much as four phases of an ongoing mission to advance Verus and its contribution to society that we can currently express. From its inception and throughout the project, the Verus chain will serve as our fungible value chain, and use of Verus tokens on this chain will be the core value around which we continue to develop the Verus vision.

The first Verus chain, which is already available at the time of this white paper's release, includes zk-SNARKs for privacy, both transparent and private addresses, a brand new CPU-mineable hash algorithm for proof of work, a brand new proof of stake algorithm, and a unique emission schedule among fair launch cryptocurrency projects. Each phase of our project's development will introduce capabilities and experiences that provide independent value on their own, further leverage Verus Coin to power operation, and create a foundation upon which to build the next phase.



We intend and hope for the Verus project to become a worldwide, inclusive community effort, which welcomes and rewards those who contribute. Although we articulate these first four phases here, we see these phases as just the beginning, and we hope you will join us by participating in and contributing to the Verus project to make our world a better place.

4.1 Phase I – Mainnet – 50/50 PoW/PoS – Time Locks – Fair Launch

In phase one, the Verus main network began with a slow start and 15 minutes advance notice at 7:15am GMT, Monday, May 21st, 2018.

The Verus network began emitting first 0 block reward, rising linearly each block over the course of 7 days until block 10080 to its peak of 384 coins per block. The Verus emission schedule is as follows:

Era 1:

Block 0 - 10080 reward: 0 to 384, rising linearly and changing each block

Era 2:

Block 10080 - 53279 reward: 384
Block 53280 - 96479 reward: 192
Block 96480 - 139679 reward: 96
Block 139680 - 182879 reward: 48
Block 182880 - 226079 reward: 24

Era 3:

Block 226080 - 1277279 reward: 24
Block 1277280 - 2328479 reward: 12
Block 2328480 - 3379679 reward: 6
... halving indefinitely every 1051200 blocks (approximately 2 years)

We also added another fair twist on the launch that we believe will contribute to a more stable value growth in the Verus currency, without any unusual risk of dumping by any single party. During the first 5 months, Verus will have an accelerated reward curve for mining and staking, with a halving every month. During the first two months, when the block rewards are at or above 192 Verus, mined blocks will have time locked coinbase transactions, preventing spending, staking, or transferring of those coins for a period that varies from approximately 3 months after the genesis block, based on numbers of blocks, to 2 years and 3 months. These time locked coins provide a smooth release of the Verus currency supply as well as incentive to all of their owners to support the long term success of the Verus network and project.

From the very beginning of its operation, Verus operated with a dual proof of stake / proof of work mechanism for all participants. Verus mining with the VerusHash algorithm, as of this point, is a CPU-only algorithm, designed with a Haraka512 V2 [30] core to be quantum secure and to maximize performance on modern CPUs. While VerusHash was designed to be CPU-optimized and did not begin life with GPU or ASIC miners capable of beating CPUs in its mining, the Verus developers have no illusions that it is possible someone might develop either a GPU or ASIC-based miner that could be meaningfully superior to CPUs.

To ensure that such a solution if developed, remains open, jl777, lead developer of the Komodo Platform, has offered a 1 Bitcoin bounty to any developer who can make and publish in an open manner, a GPU miner for the VerusHash algorithm that can significantly outperform the current CPU miner. In order to win the bounty, the source code for the miner must be available under open source licensing. To be eligible for the bounty, any implementation must be able to perform 5x better than a modern, high thread-count, high clock-rate CPU on a GPU costing under \$1000. By bringing up the Verus network this way, we established an immediate, baseline set of functionality, above which we can build out our project and community around a functional coin and project, as we stay in sync with advancements in the underlying Komodo platform.

Verus phase one release was pre-announced on Bitcointalk [28] with zero premine, and team members mined and staked along with everyone else to generate coins. In addition to coins mined by individual team members for their own benefit, the Verus developers intend to donate most of their mined and staked earnings to a community Verus foundation along with other donating community members who will join us, in order to support the ongoing growth and project development by core developers and the community for years to come.

4.2 Phase Two – Autochains, Polls, Verus Mobile

As soon as we completed phase one, we entered phase two, which includes release of this white paper, activation of the community through donations of some of our mined coins, and a push to finish planning for, development and release of mobile support, autochain capability, and the ability to setup, and then easily run CVT polls at scale.

4.3 Phase Three – Identity, and Proof of Virtue

In phase three, we plan to further improve the mobile experiences, implement the identity system, including support for strong, decentralized identity and attestations, separating photo IDs and other photo content into components that can be separately verified in unbiased human polls. At this point, we intend to support optional KYC strong identities, via both notarization of identities as well as poll-based identity verification. This will also be the first phase that supports Verus chain validation PoV enhanced PoS.

4.4 Phase Four – Integrated Machine Learning, Content, Storage

In phase four, Verus will begin to truly leverage the foundation built in phases one, two, and three by focusing on broad content classification capabilities for off-chain content, improving storage management support in its distributed hash table implementations, supporting storage monetization in some way, and providing open source, public implementations of machine learning systems that can learn from data on the Verus network to solve real world challenges of today and tomorrow.

5 Forward Looking Statements

This paper includes predictions, statements of intent, discussion of plans, estimates or other information that might be considered forward-looking. While these forward-looking statements represent our judgment and expectation of what the future holds, this is not an offer or solicitation to purchase any product, good, service, or security. All statements herein are subject to risks and uncertainties that could cause actual results of the Verus Project's development to differ materially. Furthermore, we intend to use the Verus blockchain as our open source development platform – contributing these technologies under permissive licensing for the betterment of society, not focusing solely on profit of anyone affiliated with the Verus project. You are cautioned not to place undue reliance, especially in any financial decision, on these forward-looking statements, which are subject to modification, update, or change for legitimate reasons both within or beyond our control. By expressing our vision and goals, the Verus Core developers are not obligating ourselves to revise or publicly release the results of any revision to these forward-looking statements in light of new information or future events.



Testnet Reset

To reset your testnet make sure Verus is closed (and no testnet daemon running) and delete the following directories, then restart the testnet daemon (or relaunch Verus Desktop, deactivate verustest and re-add verustest native):

- Linux: `~/.komodo/vrsctest` , `~/.verustest`
- Mac OS: `~/Library/Application Support/Komodo/vrsctest` ,
`~/Library/Application\ Support/VerusTest`
- Windows 10: `%AppData%\Roaming\Komodo\vrsctest\` , `%AppData%\Roaming\VerusTest` or
`%AppData%\Komodo\vrsctest\` , `%AppData%\Roaming\VerusTest`

Information compiled by Oink.vrsc@ from Verus Release Notes.

Note: revision date 2023-05-15.



Verus Information Page.

There's a lot of information about the Veruscoin project. We have gathered as much information and resources and bundled them into this FAQ, as to give a quick overview

VRSC Wallet & data location on different OS:

Linux GUI: `~/.komodo/VRSC`

Mac OS: `/Users//Library/Application Support/komodo/VRSC`

Windows: `%AppData%\Komodo\VRSC\`

CLI binaries location in Verus Enhanced Agama installation:

Note: All locations are relative to the installation location of Agama:

Windows Verus binaries: `resources\app\assets\bin\win64\verusd\` contains `verusd` and `verus`

Windows Komodo (and asset chains) binaries: `resources\app\assets\bin\win64\` contains `komodod` and `komodo-cli`

Linux Verus binaries: `resources\app\assets\bin\linux64\verusd` contains `verusd` and `verus`

Linux Komodo (and asset chains) binaries: `resources\app\assets\bin\linux64\` contains `komodod` and `komodo-cli`

CLI binaries location in Verus Desktop installation:

Note: All locations are relative to the installation location of Verus Desktop:

Windows Verus binaries: `\resources\app\assets\bin\win64\verusd\` contains `verusd` and `verus`

Windows Komodo (and asset chains) binaries: `\resources\app\assets\bin\win64\komodod` contains `komodod` and `komodo-cli`

Windows Zcash binaries: `\resources\app\assets\bin\win64\zcash` contains `zcashd` and `zcash-cli`

Linux Verus binaries: `\resources\app\assets\bin\linux64\verusd` contains `verusd` and `verus`

Linux Komodo (and asset chains) binaries: `\resources\app\assets\bin\linux64\` contains `komodod` and `komodo-cli`

Various links:

Website: [Official VerusCoin website](#)

GitHub: [VerusCoin source code for the VerusCoin wallet, miner and explorer](#)

Block explorer: [Main VerusCoin Explorer](#)

Block explorer: [VerusCoin Backup Explorer](#)

Block explorer: [Explorer for all Komodo Ecosystem chains](#)

Social Media: [Bitcointalk](#)

Social Media: [Discord](#)

Social Media: [Twitter](#)

Social Media: [Medium](#)

Social Media: [Reddit](#)

Overview: [Chain specifications](#)

Miner: [Monkins cccminer](#) for CPU (recommended) or GPU

Miner: [nheqminer](#) for Windows, Linux, and MacOS

Reported mining speeds: [spreadsheet](#) to compare hashrates

Mining pool: <https://luckpool.net/verus>

Mining pool: <https://pool.verus.io/>

Mining pool: <https://zergpool.com/>

Mining pool: <https://vrsc.mcmpool.eu/>

Mining pool: <https://wattpool.net/>

Mining pool: <https://www.nlpool.nl/>

Mining pool: <https://vrsc.dev-codex.com/>

Mining pool: <http://vrsc.52hash.com/>

Mining pool: <https://vrsc.ciscotech.dk/>

Mining pool: <http://www.lepool.com.cn:8088/>

Mining pool: <http://verus.bcmonster.com/>

Exchange: <https://atomicdex.io/> (VRSC/any listed coin)

Exchange: <https://app.stex.com/de/trade/pair/BTC/VRSC/1D> (VRSC/BTC)

Exchange: <https://safe.trade/trading/vrscbtc> (VRSC/BTC, VRSC/SAFE)

Exchange: https://www.aacoin.com/#/trade?symbol=VRSC_BTC (VRSC/BTC)

Exchange: <https://graviex.net/markets/vrscbtc> (VRSC/BTC, VRSC/KMD, VRSC/USD)

Exchange: https://www.kuangex.com/#/exchange/vrsc_usdt (VRSC/USDT)

Guide: [How to mine in under 5 minutes](#)

Wallet: [Verus Desktop releases](#)

Wallet: [CLI wallet releases](#)

Mobile wallet: [Android Mobile Wallet](#)

Mobile wallet: [iOS Mobile Wallet](#)

Paper wallet: [Verus Paper wallet](#)

Bootstrap: [Download Verus Bootstrap](#)

VerusPay: [VerusPay setup guide](#)

Information:

[[↑](#)]

Platform:	Komodo
Project type:	Decentralized public blockchain
	Open source, fair launch, no ICO
	Community driven project
Privacy:	Sapling enhanced, zk-SNARKs zero knowledge proofs
Website:	https://verus.io/
Github:	https://github.com/veruscoin There is the VerusCoin source code for the VerusCoin wallet, miner and explorer.

Block explorers:	https://explorer.verus.io/
	https://explorer.vrsc.0x03.services/
	https://dex.explorer.dexstats.info/
Announcement:	https://bitcointalk.org/index.php?topic=4070404.0
Discord:	https://discord.gg/VRKMP2S
Twitter:	https://twitter.com/veruscoin
Medium:	https://medium.com/@veruscoin
Reddit:	https://reddit.com/r/veruscoin
Mining Algorithm:	VerusHash 2.1 - PoW/PoS 50/50 = Proof of Power - 51% attack resistant
	CPU targeted, non-discriminatory, CPU/FPGA equalizing algorithm
Block time:	1 min
Block reward:	24 VRSC since 09/28/2018 (both mining and staking; next halving on Sept. 28, 2020)
Total supply:	83'540'184, no premine, no ICO.
Reward emission schedule:	Started with linear ramp, changing every block for the first 10,080 blocks (0 to 384 VRSC) and then halving every month for the following 5 months, then every 2 years after that. All rewards, at or above 192 are time locked for random release between block 129,600 and 1,181,520 (3 months to 2 years and 3 months after genesis).
Reward maturing:	100 blocks to maturity
PoS Period:	150 blocks cooldown of UTXO
Unique features:	VerusID: VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. VerusIDs provide plenty of opportunity for identity applications.

Mining:

VerusCoin can be mined with CPUs, GPUs and FPGAs, solo and in pools. However, the algo is carefully designed for CPUs, and they still substantially outperform GPUs.

ARM mining works in general, but although fairly efficient, without high hashrates. FPGAs can mine this coin, but don't outperform CPUs by much.

Miners:

Besides solo-mining with Verus Desktop or CLI wallet, you can use:

ccminer for CPU (efficient for most modern CPUs) or GPU <https://github.com/monkins1010/ccminer/releases>; nheqminer <https://github.com/VerusCoin/nheqminer/releases> (for Windows, Linux, and MacOS);

A spreadsheet to compare hashrates can be found here:

<https://docs.google.com/spreadsheets/d/1RrSYJDV0Mjj3X->

myMC3aQDGkciplvxHsD7ZxJ3r5f_A/edit#gid=201266774

You can also compare older and current algos

Mining pools:

<https://pool.verus.io/> (fees will go to VERUS foundation)

<https://luckpool.net/verus>

<https://zergpool.com/>

<https://vrsc.mcmpool.eu/>

<https://wattpool.net/>

<https://www.nlpool.nl/>

<https://vrsc.dev-codex.com/>

<http://vrsc.52hash.com/>

<https://vrsc.ciscotech.dk/>

<http://www.lepool.com.cn:8088/>

<https://zpool.ca/>

<http://verus.bcmonster.com/>

Staking pools:

<https://discord.gg/4FJJRY5> (Ginasis Staking pool, 5% fee)

https://docs.google.com/spreadsheets/d/1Up1WbMuCR21e6TxLae6zjLJePu_RIOVZwqShRj9vVvc/edit?usp=sharing (Dudezmobi Staking pool, 1% fee)

https://www.aacoin.com/#/trade?symbol=VRSC_BTC (Technically not a pool, but it stakes your VRSC on the exchange, 20% fee)

Price: <https://veruspay.io/price/>

Exchanges:

<https://app.stex.com/de/trade/pair/BTC/VRSC/1D>

<https://safe.trade/trading/vrscbtc>

https://www.aacoin.com/#/trade?symbol=VRSC_BTC

<https://graviex.net/markets/vrscbtc>

<https://atomicdex.io/> (in Beta)

https://www.kuangex.com/#/exchange/vrsc_usdt

Also check out the Guide on how to mine in under 5 minutes: <https://medium.com/veruscoin/how-to-start-cpu-mining-verus-coin-vrsc-from-your-laptop-in-under-5-minutes-f69c9aae340e>

Wallets:

Verus Agama GUI wallet is a forked Agama Desktop App for multiple coins (for Win, Linux and Mac). It's been deprecated now and no longer supported;

Verus Desktop GUI wallet is a newly developed multi-coin wallet supporting VerusID

<https://github.com/VerusCoin/Verus-Desktop/releases> (for Win, Linux, Mac and ARM Linux);

the CLI wallets can be found here: <https://github.com/VerusCoin/VerusCoin/releases/>

There's also possibilities to test coming PBaaS functionality in this wallet.

A beta version of the Android mobile wallet can be found here: <https://github.com/VerusCoin/Verus-Mobile/releases>

A beta version of the iOS mobile wallet can be accessed via this Apple TestFlight invite

<https://testflight.apple.com/join/ZS43lYcw>

Our Paper Wallet can be accessed here: <https://paperwallet.verus.io>

If you need a bootstrap, you can find it here: <https://bootstrap.verus.io>

(a guide how to apply is pinned in #community-support channel in Discord: <https://discord.gg/VRKMP2S> or in the HOW-TO & FAQ section of our website https://wiki.verus.io/how-to/how-to_bootstrap.md)

For those interested in VerusPay, a guide can be found here: <https://veruspay.io/setup/>

And finally for those interested in running various Veruscoin services:

<https://github.com/VerusCoin/VerusServicesSetup>

Information compiled by Thoskk.vrsc@, complemented by Godballz.vrsc@ and Oink.vrsc@.

Note: revision date 2020-11-11.



VerusID

VerusIDs are a fully functional blockchain protocol, not just an ID system. There is no corporation involved in the protocol, unlike most blockchain ID implementations. **VerusIDs** provide plenty of opportunity for identity applications. Specifically, **VerusID** provides:

Quantum-ready friendly crypto-addresses on the worldwide Verus network

VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.2 does not include quantum resistant signatures for transactions, Verus IDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID->destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully Decentralized Protocol

Anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers. Verus IDs are:

- Revocable -- each ID includes a revocation authority, which defaults to the identity self. If another ID is specified as the revocation authority it can be used to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft or turnover in an organization.
- Recoverable -- each ID also includes a separate recovery authority, which also defaults to self. If another ID is specified as the recovery authority it can be used to recover the ID from its revoked state, with the option to alter the primary authorities used to spend and sign.
- Private - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong identity, attested to with a quantum secure signature when that is available.
- Powerful - Multiple addresses or other IDs can be defined as primary addresses, and any number of those may be required to spend, sign, or alter the identity (N of M). The revocation authority may only be altered by the revocation authority, and the same applies to the recovery authority, either of which may be another identity with its own N of M multisig controls for its primary addresses.



Worldwide Verifiable Digital Signatures for All!

Verus digital signatures, based on Verus ID, offer the first worldwide verifiable, decentralized, single or multi-sig friendly-name signatures for content such as messages and files, authentication, and attestations, with full support for revocation and recovery in case of key loss or theft.

Bitcoin, which was first to enable worldwide, P2P decentralized blockchain transactions, and most of its derivatives, offer ways to sign messages with specific private keys that can be verified against public addresses. While that does offer some limited signing capability, it does not offer capabilities that can compete with centralized services offering more sophisticated key management. A single public/private key pair lacks critical capabilities to make it suitable as an actual identity, most notable and obvious being the ability to recover from loss or theft of private keys, but arguably as important are friendly name aliases, modifiable multi-sig signing, updates, privacy, and the association of signed attestations by other identities to statements about properties of the identity.

Verus ID enables free verifiable digital signatures for all through the Verus ID protocol as just one of the many new capabilities it enables. It is also the foundation upon which many new applications and additional capabilities can be built.

For example, using Verus ID signatures, it's possible for any journalist anywhere to sign photos, videos, and content, establish a reputation for authenticity, and counter the potential for deep-fakes to make the truth harder to find. Open source projects can now create their own identities and digitally sign their binary releases, ensuring that not only can a file be verified by hash as the one downloaded from a particular server, but by signature as the actual file initially signed by the developer or release engineer. Signatures also form the basis for any attestation of one party to the validity of another. In fact, there are so many applications for digital signatures, from things listed already, to physical entry systems, to workflow applications, to new earning opportunities that a full discussion of use cases would overwhelm these release notes.

In any case, we are happy to release digital signatures for all, and we hope you enjoy using this new, simple capability, maybe even think of a new use case you'd like to pursue yourself as a business opportunity on the Verus Network!



VerusHash 2.1

VerusHash 2.0 was the first algorithm to significantly equalize FPGAs dominance over CPUs, once they were introduced on the Verus network. While FPGAs were intentionally not blocked completely, which would simply drive the performance battle to the higher end and further into secret, the VerusHash 2.0 algorithm was developed to explicitly equalize FPGAs and modern CPUs and has met its original goals in keeping FPGA performance for the price under 2x of CPU. VerusHash 2.1 introduces an adjustment to the equalization technology, which we expect to tilt the balance a bit more favorably towards CPUs, while still enabling FPGAs to operate on the hash algorithm with minor modifications. Verus Developers have proactively reached out to FPGA manufacturers and made the new algorithm available to them, so that everyone will have an opportunity to mine and stake when the Verus economy starts to roll and identity rewards, which will not inflate the currency, but should far exceed the potential for block rewards, begin streaming from the network.



Welcome to Verus

[The Future Is Now](#)

[The VerusCoin Community is YOU!](#)

[NEW TO VERUS?](#)

[Getting started:](#)

[VerusID](#)

[VerusPay](#)

[Be a part of it all](#)

[Closing Thoughts](#)

The Future Is Now

[[↑](#)]

Here is a collection of information meant to span several platforms and services in order to keep all of us more informed on what's happening across the growing, world-wide community. It will also help to connect us, allowing us to discuss what's being worked on, what can be done to help and as a way to look ahead in a more involved way.

Verus, latin for “True”, combines all of the features we believe are important in a cryptocurrency, and provides a foundation for future development. Verus is a fork of Zcash and Komodo that leverages the Komodo platform, and we appreciate the contributions and support from those teams as well as the Bitcoin developers that created a foundation for us to launch new capabilities **in a system that supports Bitcoin and Zcash compatible transaction types as well as Komodo cross chain swaps** and dPoW security enhancements.

1. Verus Coin is a pure technology provider and does not endorse third party projects.
2. Be your own bank: **you** are responsible for securing your coins and taking backups.
3. **No VerusCoin community member, admin, or developer should ever, under any circumstances, ask for your private keys or for you to send coins to them. Please report any incident in the #community-support channel.**

The VerusCoin Community is YOU!

[[↑](#)]

If you've been around long enough and feel a little skeptical or disillusioned from a few bad experiences elsewhere, it's totally understandable. We've all had them and we know quite well that when unscrupulous projects abuse words like decentralization, interoperability, or protocol level solutions, it can have a “boy who

cried wolf" effect.

We believe that actions speak louder than words. Anything that may be considered hype can be backed up with functional examples. What we're doing here is genuine. We are trying to be the real deal, **just as Bitcoin started out, as an open-source, fair launched, no ICO, or pre-mined, or even dev funded project**. Despite this, a foundation has been established by and for fellow community members. There is always more to be done and the foundation regularly pays out bounties to community members that help to make the Verus vision a reality. In fact, just about everything has been designed to reinforce the community. Designed with efficiency in mind, only the miners and stakers are rewarded for securing the chains. This ensures bare minimum costs to the end user starting up a chain and the sleek UI removes the technical barrier.

NEW TO VERUS?

[↑]

What we're about:

Put simply, Verus is much more than any single ordinary blockchain; more of an entire ecosystem of **interconnected blockchains that all operate decentralized and at the protocol level**. Verus introduces Verus PBaaS (**Public Blockchains as a Service**), a true publicly notarized blockchain as a service with an easy to use built in wallet UI, designed to make it so that now anyone can start their very own full-fledged cryptocurrency so long as it can be funded or supported by miners. Newly created chains are just as secure as Verus itself and have the ability to be merge mined (up to **22x** at once!)

Benefiting both the user and the miners, self-strengthening the ecosystem and ensuring minimal fees (no room for middlemen to take a cut). Reserve backed currencies are also possible with **fractional reserve** capabilities and since interoperability is paramount, exchanging currencies in **cross-chain transactions** is as simple as sending coins from one wallet to a different coin's wallet. **Conversions** are handled by a built in market maker that automatically determines price based on a predetermined curve and issues orders fairly to all buy/sells within a block, with zero spread and the added benefit of eliminating the well-established problem of front-running, just some amount of slippage based on the net buy/sell.. With reserve currencies, buyers can make buys without even needing sellers (and vice versa).

Amazingly, everything is done at the **protocol level** on a decentralized network, meaning Verus and it's associated chains can't be censored or stopped.

Getting started:

[↑]

Important links:

Website: <https://verus.io>

Wallet: <https://verus.io/wallet>

GitHub: <https://github.com/veruscoin>

Explorer: <https://explorer.verus.io>

Discord: <https://verus.io/discord> Check <https://youtu.be/YVOfIMjRf30> if you only see one empty channel.

Bitcointalk: <https://bitcointalk.org/index.php?topic=4070404.0>

Max Supply: 83,540,184

Algorithm: VerusHash 2.1 PoW/PoS - 50/50

Block time: 1 minute

Proof-of-Stake (PoS) ROI is currently between 6% and 8%

Halving Frequency: roughly every two years

Next Halving: Current ETA: Jan 2025 (Block #3,381,840)

Halving Countdown: <https://countdown.verus.io/halving>

VerusID

[[↑](#)]

VerusIDs are true self-sovereign identities and aren't simply an ID system as much as a fully functional blockchain protocol. There is no business in the protocol, but plenty of opportunity for those who use what it can do for identity applications. Verus ID provides:

Quantum ready friendly crypto addresses on the worldwide Verus network -- VerusIDs can be used to receive and send funds, which are controlled by the single or multi-sig addresses specified in the identity itself. If these controlling addresses or the single or multi-sig properties are changed, which can be done by the controller of the identity, all future spends of UTXOs sent to that identity follow the updated spend conditions and are subject to the updated keys. Although Verus 0.6.0 does not include quantum resistant signatures for transactions, VerusIDs are themselves resistant to quantum attack with known algorithms, and we have already started to integrate a quantum secure signature scheme, which we expect to activate on mainnet early next year. When that is available, it will be possible to change an ID and have all of the funds sent to it made retroactively quantum resistant. Verus IDs can also be used to publish ID -> destination address mappings on other blockchains, but only the Verus ecosystem has the ability to revoke, recover, inherit, funds in existing UTXOs.

Fully decentralized -- anyone can create one and have complete, self sovereign control over it without permission to do so. All costs to create an ID go to miners, stakers, and ID referrers.

Revocable -- each ID includes a revocation authority, which defaults to the identity self, and which has the permission to revoke the identity, which creates a valid transaction that, once mined into a block, prevents the identity from being used to spend or sign until it is recovered, effectively freezing all of its funds, for example, in the case of key theft.

Recoverable -- each ID also includes a separate recovery authority, which also defaults to self, and which can recover the identity through redefining its primary state and the recovery state as well, though it cannot modify the revocation state, or vice versa, unless they are both controlled by the same underlying authority.

Private - Each ID contains a set of zero-knowledge private addresses, which can be used as messaging, financial, or voting endpoints, and each ID also contains a content map of key-value hashes, intended to be used alongside applications and various identity policies to provide everything from private yet selectively provable claims and attestations to selectively provable components of a strong passport, attested to with a quantum secure signature when that is available.

A Built-in Decentralized Referral Program, Enabling Natural Growth.

Verus IDs will cost Verus to acquire, 100 Verus per ID to be exact, which can be discounted to 80 Verus with referral of an existing Verus ID. The interesting twist is that all of the cost of an ID goes back into the network, either as referral fees, which are a way to get discounts or possibly even make money through referrals and built into the identity transactions themselves, or as mining and staking fees. No one besides people participating in the network as miners, stakers, or ID referrers take any proceeds from the cost of an identity. If you refer someone new and they purchase an ID with your ID as a referral, they will receive a 20% discount on the ID. In addition:

1. You will receive 20 Verus directly
2. The person who referred you, if there is one, will receive 20 Verus
3. The person who referred person b, if there is one, will receive 20 Verus, and

4. The miner or staker will receive the remainder of the discounted 80 Verus cost

As you might expect from looking at the fact that anywhere from 20 to 100 Verus goes to miners and stakers of Verus for each new identity once identities hit mainnet, some Verus blocks may have VERY high rewards for some time. The best thing about that is that regardless of how much the Verus blockchain rewards miners over and above the pre-determined coinbase reward, it will be as a result of the **on-chain economy**, paid for by people buying identities with **no inflation** of the money supply! If you or your friends missed the early days of the Verus launch, **you don't want to miss this** new opportunity to be mining, staking or referring now **Verus IDs are activated on the mainnet!**

VerusPay

[[↑](#)]

Blockchain-integrated payment gateway for accepting Verus Coin (VRSC) in a WooCommerce ecommerce store

This plugin extends *WooCommerce* on *Wordpress*, adding the ability to accept cryptocurrency payments in Verus Coin (VRSC) using either an on-store wallet daemon (best for VPS or dedicated hosting stores) or manually configured VRSC addresses (best for shared hosting stores).

When an order is submitted via the VerusPay gateway, the order will be placed "on-hold" while awaiting payment from the customer. The customer has a limited time wherein to send the payment and the store monitors the wallet/address to confirm payment received before releasing the order and redirecting the customer to the Thank You page.

VerusPay uses limited API functionality for Manual Mode, to communicate with the blockchain explorer in verifying payments and with the veruspay.io API to get up-to-date price data. These API's do not receive any private data either about the store owner, store, or customer. The only data sent to the block explorer API is the public/transparent blockchain transaction and address used. For VerusPay.io API price data, only the store-set currency is sent to retrieve the current fiat exchange rate for Verus Coin.

[VerusPay installation instructions](#)

Be a part of it all

[[↑](#)]

Focus on Verus' social media presence has picked up with new, fresh looking Facebook and YouTube channels so check them out and feel free to post something or simply explore the content. There's so much going on, plenty to learn about.

So much so that there is an *open call to anyone and everyone out there who can submit graphical or textual content* that can be used to help support the upcoming mainnet launch.

A stronger effort from our already wide base of members to post and utilize social media tools to help spread the word and generate new interested users is something we can all do right now. There is no better time than now and getting fresh new minds in here to discover what is being done is the first step in getting the great snowball rolling. To those of us who truly care and believe in this project, then this message is meant for you.

Looking Ahead

Since we are on the verge of an historical mainnet release with capabilities that have the potential to revolutionize both the financial and technical sides of fintech and send shockwaves across many industries. There are still lots of things to be done but each day we collectively push closer and closer to making it a reality.

Social Media Links

Check out the brand **new** community website, community social media pages, new Veruscoin YouTube and Facebook pages for helpful how to or to just keep up to date with things. Speaking of how to, we can always use more helpful videos if any community members would like to take it upon themselves to chip in. The Medium page also has lots of helpful guides and there's always something to Tweet about. Be sure to drop by the discord to meet the vibrant and helpful community in real- time, where it isn't uncommon to find the developers offering assistance. Where else can you find that?

We are all working together to realize something great. If you'd like to join us and be a part of that special something but don't know how to, it can be as simple as checking out the links below and helping other's to understand that there's never been anything in existence quite like Verus, a complete, easy to use, fully decentralized blockchain ecosystem, designed specifically to benefit all participants in one of the most low cost yet efficient in every way, end-to-end systems to date.

Facebook - <https://www.facebook.com/VerusCoin/>

YouTube - https://www.youtube.com/channel/UC_-KCHBxaDwSgNMdE3LMThg

Discord -<https://verus.io/discord>

Twitter -<https://twitter.com/veruscoin>

Medium - <https://medium.com/@veruscoin>

Reddit -<https://reddit.com/r/veruscoin>

Community twitter - <https://twitter.com/VerusCommunity>

Closing Thoughts

[↑]

As a community project of just regular Joes, it can't be stressed enough how important each and every one of us all are. The true power of decentralization is putting power back in the hands of **YOU**.

created by Rozo@ and Godballz@

Note: last revision date 2022-03-09.



Question: How do I direct all my solo mined rewards to a single Verus wallet?

[Verus , Chaindata & standard locations](#)

[Prerequisites](#)

[In Verus Desktop](#)

[In Agama Wallet](#)

[In Verus CLI](#)

Attention: Read it completely before using.

Verus `Wallet.dat` , Chaindata & `VRSC.conf` standard locations

[[↑](#)]

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help` , `Show Verus data folder (default)`

Prerequisites

[[↑](#)]

- Have a **native** VRSC wallet running

In Verus Desktop

[[↑](#)]

In Verus Desktop, there is at this moment no way to enter an address to mine to. However, editing the `VRSC.conf` can be done to achieve our goal:

In the receive window of your wallet, click the hamburger (three vertical dots) next to the address you want to receive your rewards in and click `Copy Public Key` . Close down Verus Desktop.

Edit `VRSC.conf` (see standard locations at the top) and add the line `pubkey=THELONGSTRINGCOPIED` . Save and exit.

* Start Verus Desktop as you normally do.

In Agama Wallet

[[↑](#)]

Step 1 - First get your wallet address you want to mine to:

- * If you don't have an address, click "Receive", click "Get New Address" and choose "Transparent Address" from the drop down.

Step 2 - Next we need to retrieve our pubkey,

- * click on the hamburg next to the address that you want to receive the rewards in and click [copy pubkey](#)

Step 3 - Set your PubKey

Go to 'Settings', 'App Config (config.json)' and enter your pubkey(*THELONGSTRINGCOPIED*) into the 'Pubkey VRSC mining key' field. Click 'Save app config' to save these settings.

- * Restart Agama

In Verus CLI

[[↑](#)]

Step 1 - First get your wallet address you want to mine to:

You can find an address if you already have previous transactions, or you can create a new one. To find an address from a previous transaction, use the command line verus listtransactions and copy the address found after "address".

To generate a new wallet address, use the command line [verus getnewaddress](#) and a new address will be created.

Step 2 - Next, using your new address, enter the command with verus-cli [verus validateaddress](#). From the output find the long string after "pubkey", copy without the quotation marks.

Step 3 - Set your PubKey

Option 1: use this pubkey when starting your daemon by adding the following line to the end of your command, just before the "&" sign: -pubkey=THELONGSTRINGCOPIED Option 2: edit your [VRSC.conf](#) and add the line [pubkey=THELONGSTRINGCOPIED](#). Then start your whallet as you are used to.

Your rewards will now be mined to that address. It would be a good idea to keep notes and associate the wallet address with the pubkey...also to double check that you did validate the correct pubkey for the wallet address, making sure you made no errors.

(submitted by @Oliver Westbrook, edited by Oink.vrsc@)

note: last revision date 2020-02-24.



Question: What should I do if I end up on my own fork because of a network issue or having an old version of the wallet?

[Procedure 1 \(Easy, installing bootstrap\)](#)

[Procedure 2 \(Time consuming, no extra download\)](#)

This solution can be solved in 2 ways: you can simply install the latest bootstrap file (less work, big download) or search manually for the forked block and invalidate that block (time-consuming, no download needed.)

Procedure 1 (Easy, installing bootstrap)

[[↑](#)]

For all GUI or CLI users.

1. Stop the wallet/mining process by cleanly shutting down the program.
2. Update your wallet if necessary.
3. Follow the procedure in [HOW-TO Backup, Install or Update and Bootstrap your wallet.md](#) to efficiently rectify the problem.
4. Do not be dismayed if it seems that your mining rewards suddenly seem to come to a halt. Remember, when you mine to the wrong chain rewards can come in very quickly, but they are worth nothing.

Procedure 2 (Time consuming, no extra download)

[[↑](#)]

Verus-Desktop

1. The commands are **all** entered in the *Native Client Terminal* that is located under [Settings](#), [Coin Settings](#).
2. Search for the **earliest** block that not matches the blockchain:
`run getblockhash <suspected blocknumber>` will show you the blockhash for the blocknumber you filled in
The response shown in the *Native Client Terminal* will be similar to this:
`5cc7844973fb95ef17f1772ea4aba579f0d8273fb0ee6064cd8e707d1056c646`
3. Check the blockhash your command gave you against the blockhash the [explorer](#) shows.
4. If the blockhash from the explorer is different than yours, repeat steps 2 & 3 until you find the earliest block that is different.
5. Use the **earliest incorrect blockhash** from your system to invalidate that block:
`run invalidateblock <earliest incorrect blockhash>`
The *Native Client Terminal* will not give feedback on this command.

6. Now use the **correct blockhash** that the explorer gave you for the block you just locally invalidated:

```
run reconsiderblock <correct blockhash>
```

Again the *Native Client Terminal* will not give feedback on this command.

7. Once your wallet connects to a node that is on the correct chain, it will quickly synchronize.

If needed you can either restart your wallet to force new connections or manually disconnect bad nodes.

CLI

1. Search for the **earliest** block that not matches the blockchain:

```
./verus getblockhash <suspected blocknumber>
```

 will show you the blockhash for the blocknumber you filled in

The response will be similar to this:

```
5cc7844973fb95ef17f1772ea4aba579f0d8273fb0ee6064cd8e707d1056c646
```

2. Check the blockhash your command gave you against the blockhash the [explorer](#) shows.

3. If the blockhash from the explorer is different than yours, repeat steps 1 & 2 until you find the earliest block that is different.

4. Use the **earliest incorrect blockhash** from your system to invalidate that block:

```
./verus invalidateblock <earliest incorrect blockhash>
```

The daemon will not give feedback on this command.

5. Now use the **correct blockhash** that the explorer gave you for the block you just locally invalidated:

```
./verus reconsiderblock <correct blockhash>
```

Again the daemon will not give feedback on this command.

6. Once your daemon connects to a node that is on the correct chain, it will quickly synchronize.

If needed you can either restart your daemon to force new connections or manually disconnect bad nodes.

(submitted by @jimboscott, Edited by Oink.vrsc@)

Note: last revision date 2020-11-11.



Question: ERROR: Your wallet.dat is not matching the blockchain. Please restart the wallet with -reindex param.

It seems you probably trying to move coinbase coins that you must move them through a private address (zaddress) first by shielding your coinbases, which is required by Verus blockchain rules.

Procedure

[↑]

1. go to the "Receive" screen and make sure you have a private address (starting with `zs`), if you don't have that create one.
2. Copy the new private address to your clipboard
3. Go to the "Mining" screen, select Verus and select "Shield Rewards"
4. Leave "All unshielded funds" as the source and paste the new zaddress into the destination
5. Click on "Continue"
6. Wait a while until you have a private balance showing on the Wallet screen
7. Send again, this time from your private address to any normal transparent receive address that you have.

(submitted by @keda666, solution written by mikeout.vrsc@)

Note: revised 2020-04-24

Question: How do I unshield my coin rewards and get them staking on CLI?

Important General Information

`verus command "<userinput>"` needs to be entered literally, with `<userinput>` replaced by your specific userdata. So if the text directs you to use for example `"<Public Address>"`, you replace that (including the `<` and `>`) with the address, so it looks similar to this: `"RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r"`.

Remarks on Windows command line formatting:

The CLI help shows the command format for Linux and MacOS. On the native windows command prompt (`cmd.com`) the formatting is different.

In windows command prompt, substitute the shown `'` -character with the `"` -character. In windows command prompt, substitute the shown `"` -character with the `\\"` -characters.

* In windows command prompt, omit the preceding `./`.

Note: As an example, in Linux the command:

```
./verus z_sendmany <my_private_address_without_quotationmarks> '[{"address": "<my_transparent_address>", "amount": <95.9998>}]'  
should be entered on the Windows command prompt as:  
verus z_sendmany <my_private_address_without_quotationmarks> "[{\\"address\\":\"<my_transparent_address>\",\\\"amount\\\":<95.9998>}]"
```

Procedure:

[[↑](#)]

- 1) `./verus z_shieldcoinbase "*" my_private_address_without_quotationmarks`
this capture all so called coinbases, i.e. mined coins that are not yet staking.
you have to wait 100 blocks (minutes) after receiving them before being able to move them.
wait for a few minutes for the tx to be confirmed.
- 2) `./verus z_getbalance <my_private_address_without_quotationmarks>` this is to subtract the 0.0001 VRSC fee from the balance in the next step.
- 3) `./verus z_sendmany <my_private_address_without_quotationmarks> '[{"address": "<my_transparent_address>", "amount": <95.9998>}]'`
amount is minus 0.0001 from balance and without quotation marks.

(4) to verify: `./verus z_gettotalbalance`

after a few minutes (operations from private addresses are a bit time consuming)

Note: I am always using the same private address.

(submitted by @karero, based on @dukeleto)

Note: last revision date 2020-04-12.



Question: How can I check my immature balance in a graphical Wallets?

[Verus Desktop](#)

[Agama \(Deprecated\)](#)

Verus Desktop

[[↑](#)]

Verus Desktop shows your immature balance in your wallet dashboard, between the [Transparent Balance](#) and [Private Balance](#).

Agama (Deprecated)

[[↑](#)]

1. In the Agama wallet click on the hamburger menu (the three stripes) on the top right
2. click on [Settings](#)
3. click on the item [CLI](#)
4. Select [VRSC](#) as coin
5. Type the following command: [getwalletinfo](#)
6. click [Execute](#)
7. scroll down and find "immature_balance": which will give you the amount of time-locked coins in your wallet.

(submitted by [@karero](#))

Note: last revision date 2020-04-24.



Question: How can I tell the difference between staked and mined coins?

CLI:

Verus Desktop:

Verus Agama (Deprecated):

CLI:

[[↑](#)]

You can check this in the VerusExplorer <https://explorer.verus.io/>

1. Check last 10 transactions in the </>CLI. Use `listtransactions` .
2. Copy the blockhash of the received award
3. Add it at the end of [https://explorer.verus.io/api/getblock?](https://explorer.verus.io/api/getblock)

Hint: An example: <https://explorer.verus.io/api/getblock?hash=9e6fa91356211a554c580c90ec9c2067dd420ff74c7d33481775793f7b0e7f03> – so this one is minted....

Verus Desktop:

[[↑](#)]

In Verus Desktop, simply go to your Mining Dashboard and enter into Verus details.

Scroll down to the bottom of the page. It will list the rewards as `mined` or `minted` in green.

The TXIDs that staked the minted rewards are shown in blue.

Verus Agama (Deprecated):

[[↑](#)]

In the GUI you can also click yourself thru to that information.

1. Click on the magnifying glass all the way on the right of the transaction.
2. On the pop-up click on "Open in the VRSC Explorer" bottom left.
3. In the VRSC explorer click on the block hash value (in light blue) – now the block hash is displayed as the title of the box.
4. Click on the info "i" on the right and click on "search": in the result displayed for blocktype 'mined' or 'minted'.
(edited)

(submitted by karero, edited by Oink.vrsc@)

Note: last revision date 2020-11-11.



Question: How does staking work?

You'll start staking with the first VRSCs that are not time locked + in your public / transparent wallet + 150 blocks old (or about 2.5 hours).

Your chances to win a block: Your coins in your public/transparent wallet / Total staking supply in public/transparent wallets (which is max. about 485.000 VRSC from the first (sunrise) week as long as the rewards were below 192)

Remember: There will be only one reward every minute. It's going to be either mining or staking, so on average 720 mining and 720 staking rewards every day.

Example: I have 300 coins in a public/transparent address / 300.000 in public wallets (let's assume some part is lost/not staking or in private wallets), so that would be $1/1000 \times 720$ of a chance or around average 1,4 days for a staking block rewards. Hash power does not influence staking reward.

Regarding the Verus debug.log: "No eligible staking transaction found". It means that you are staking but have not received a reward yet. @miketout will change the message soon.

Regarding time locked coins:

The Zcash protocol requires you to send all coins received by mining (on wallet, not pool mining) or staking (reward transactions, also on wallet, not pool staking) once unlocked to a private address and then to a public/transparent address before you can use them either for staking or for making transactions (that's how you make use of your rewarded coins = coinbase coins). So, once your coins loose their time lock, you can unlock those coins as described in ["Shield Verus Coins via Command Line Interface"](#!how-to/how-to_shield_via_cli.md). Once you've transferred the coins from your private address back to (one of) your public / transparent address(es) and you'll automatically start staking.

(submitted by @karero, edited by Oink.vrsc@)

note: last revision date 2020-02-25.



Question: How do you know a block was minted (staking reward)?

You can see the following characteristics:

1. a block hash without a lot of leading zeros
2. a last transaction that in your wallet has no fee, is from the same address that it is to, which is also the address the coinbase is to (easiest confirmation) and
3. The POS difficulty is encoded in the low 32 bits of the nonce with 96 of the next nonce bits as zero/reserved, then a random hash at the top half

(submitted by @keda666)



My wallet is stuck on block number XXXX. It does not synchronize properly anymore.

note: Read it completely before use.

Verus `Wallet.dat`, `Chaindata` & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

Usefull links:

Link 1: [Download latest Wallet](#)

Link 2: [Show current blockheight](#)

Procedure:

[[↑](#)]

In case your wallet is not synchronized with the blockchain and restarting doesn't connect to any peers:

Compare your blockheight with the one Link 2 above is showing to make sure you are not synchronized anymore. If the blockheight of the link above is significantly higher (more than 10) than the blockheight your wallet is showing, follow the rest of the procedure.

If the numbers are equal or close, your wallet is synchronized and the procedure below will not solve any problems.

Close your wallet.

Go to the appropriate location for your OS as mentioned above.

Add a similar list to the bottom of your `VRSC.conf`, just below `rpcallowip=127.0.0.1`:

```
addnode=157.90.113.198:27485
addnode=95.217.1.76:27485
```

Save and exit the file.

An up-to-date list of working nodes can be found in Verus Discord in the `#tipbot` channel, by messaging `/peerinfo` in that channel.

After you added nodes, remove `peers.dat` that is in the VRSC folder.
(At least rename or move to a different location).

Make sure you don't remove any other files/folders, or you'll have to `bootstrap` your wallet.

Then start your wallet as you're used to.

If the problem persists, continue with this WIKI: [Recover from forking, network or old wallet problems](#)

Submitted by Oink.vrsc@ & Thoskk.vrsc@

Note: last revision date 2023-06-03.

Question: I accidentally send funds to my b-address and cannot move those funds

Funds sent to a b-address get locked in the same manner as the original coinbase reward was. However, because they were not sent using a script to lock those coins, they can be retrieved, without waiting the full unlock period (typically between 3 and 27 months).

Warning: DO NOT send to a b-address, unless you know what you're doing.

procedure

[↑]

First you need to determine the **TXID** of the locked funds.

1. The easiest way is to obtain the TXID from the send transaction you did in Verus-Desktop to the b-address. Make a copy of the TXID: you will need it for this procedure
 2. The next easiest way is look up the b-address on the [explorer](#) and examine the transactions to this address. The youngest transaction is usually the one you need. On the explorer the hash in the transaction is the TXID. Make a copy of the TXID: you will need it for this procedure
 3. In Verus Desktop, go to [Settings](#) --> [Coin Settings](#) and make sure [VRSC](#) is selected in the top right. Run the command:

```
run listunspent 0 <BLOCKCOUNT> '[ "<YOUR-b-ADDRESS>" ]'
```

Replace `<BLOCKCOUNT>` with the current blockcount your wallet is on.

Replace `<YOUR-b-ADDRESS>` with the ac

```
[{
  "txid": "fa5962ebf61ef31867ba73b173433841f8f68578d53b4bb30cfe1432b5820f15",
  "vout": 10,
  "generated": false,
  "address": "iBSUZSgXHEGGz65GTT6BGgchtkTHoFBs57",
  "amount": 2.20005763,
  "interest": 0,
  "scriptPubKey": "050403000000cc1b04030001011504575dc6ae7484c83c0dc97a4218f88e2cbe9b659c75",
  "confirmations": 159,
  "spendable": true
}]
```

Make a copy of the TXID: you will need it for this procedure

note: The above result is an example. **DO NOT** use data from it.

Now we need to create a raw transaction

To create a raw transaction, we will need to use the CLI-interface. In Verus Desktop

```
run createreawtransaction '[{"txid": "yourtxid here", "vout": fill in too}]' "{\"destination addr\": <amount>}" <current Blockheight -5>
```

1. adapt the above `createrawtransaction` command, making sure to subtract the 0.0001 VRSC free from the amount that is in the original TXID, similar to this example:

```
run createtrawtransaction '[{"txid": "fa5962ebf61ef31867ba73b173433841f8f68578d53b4bb30cf1e1432b5820f15", "vout": 10}]' '{"Oink@": 2.20004763}' 890450
```

In this example, the result is a long HEX-string:

Copy the string that your command gave as response, from the CLI interface of your wallet. You will need it in

the next step.

note: The above command and its result are examples. **DO NOT** use data from it. Use the results from your own wallet!

1. In the CLI interface adapt and issue this command `run signrawtransaction <string from step 1>`
In our example, that would look like this:

and your command will show a result similar to this example:

note: The above command and its result are examples. **DO NOT** use data from it. Use the results from your own wallet!

1. In the CLI interface adapt and issue this command `sendrawtransaction <"hex"-string from step 2>`. In our example that would look like this:

```
run sendrawtransaction 0400008085202f8901150f82b53214fe0cb34b3bd57885f6f841384373b173ba671
8f31ef6eb6259fa0a00000694c67010101012102c9ca37dac14c819a99ce4a71533ab8d3d5e37643ede9c4da0
981081a074f75df40531ea63fb3de6111949652111bbe524506999c97c06302715e85aa5c5813519b3eace4ac1
5bb3950600f968c0c555a935fd826f1a51e00bd2a7f12d035757fc5fefeffffff019b011d0d0000000240504030
000000cc1b04030001011504575dc6ae7484c83c0dc97a4218f88e2cbe9b659c7552960d006e960d000000000000
0000000000000000
```

and your command will show a result similar to this example:

4a5202327e6ed2ce20d3b146155ec92e52fae6c4481362faf6f8a072017b41f1

The result of this command is the TXID of the coins moving out of your b-address. You can monitor the progress in your wallet or look up the TXID in the [explorer](#).

note: The above command and its result are examples. **DO NOT** use data from it. Use the results from your own wallet!

Some words of advice after successfully removing funds locked in a b-address:
Pay attention to which addresses you send. It may be an idea to import the addresses you use into a fresh wallet, omitting all unused (b-) addresses.

Some words of advice after successfully removing funds locked in a b-address:
Pay attention to which addresses you send. It may be an idea to import the addresses you use into

omitting all unused (b-) addresses.

note: Created 2020-11-11 by Oink.vrsc@



Hardware and Software requirement for ARM.

[Verus Wallet on CLI](#)

[Verus Wallet on GUI](#)

[Staking](#)

[Solo Mining](#)

[Pool Mining](#)

There are minimum requirements on your hardware and software for running a Verus Wallet on ARM-devices. If your platform does not meet the minimum requirements, you may not be able to run the required software.

The listed requirements are for running **one** chain. Additional PBaaS chains require more resources.

Verus Wallet on CLI

[\[↑\]](#)

Absolute minimum requirements:

- 64-bit processor
- 64-bit Operating system (Raspbian is standard 32 bit)
- 2 GB memory + 6 GB Swap available to the CLI Wallet
- 20 GB storage space for Verus Blockchain and CLI wallet
- internet connectivity
- `libgomp1` and `zlib1g-dev` libraries installed

Recommended requirements

- 64-bit processor with AES functions enabled
- 64-bit Operation system (Raspbian is standard 32 bit)
- 4 GB memory or more + 6 GB Swap available to the CLI Wallet
- 50 GB storage on a *fast* medium (like NVMe device) for the Verus Blockchain & CLI wallet. This supplies room for blockchain growth over time and the ability to bootstrap the wallet.
- internet connectivity
- `libgomp1` and `zlib1g-dev` libraries installed

Verus Wallet on GUI

[\[↑\]](#)

Absolute minimum requirements:

- 64-bit processor
- 64-bit Operating system (Raspbian is standard 32 bit) with GUI interface
- 2 GB memory + 8 GB Swap available to the Verus Desktop Wallet
- 25 GB storage space for Verus Blockchain and Verus Desktop.
- internet connectivity
- `libgomp1` and `zlib1g-dev` libraries installed

Recommended requirements

- 64-bit processor with AES functions enabled
- 64-bit Operation system (Raspbian is standard 32 bit)) with GUI interface
- 8 GB memory or more + 6 GB Swap available to the CLI Wallet
- 50 GB storage on a *fast* medium (like NVMe device) for the Verus Blockchain & CLI wallet. This supplies room for blockchain growth over time and the ability to bootstrap the wallet.
- internet connectivity
- `libgomp1` and `zlib1g-dev` libraries installed

Staking

[[↑](#)]

Absolute minimum requirements:

- All requirements to run a wallet

Recommended requirements

- All requirements to run a wallet
- A fully configured and functioning NTP client, to keep your clock synchronized
- Low latency internet connectivity

Solo Mining

[[↑](#)]

Absolute minimum requirements:

- All requirements to run a wallet

Recommended requirements

- All requirements to run a wallet
- 64-bit processor with AES functions enabled
- Low latency internet connectivity

Pool Mining

[[↑](#)]

- 64-bit processor with AES functions enabled
- 64-bit Operation system (Raspbian is standard 32 bit)
- ARM release of CCMiner or NHEQminer
- Low latency internet connectivity
- A public address to mine to
- A public mining pool to connect to

Note: Revision date 2023-01-12.



error while loading shared libraries: libgomp.so.1: No such file or directory

error while loading shared libraries: libz.so: No such file or directory

When running `./verusd` on a Linux distro (eg Debian or Devuan), not all dependencies may be installed by default, resulting in the errormessage

```
error while loading shared libraries: libgomp.so.1: No such file or directory or  
error while loading shared libraries: libz.so: No such file or directory.
```

To solve this you need to install the libgomp and zlib1g-dev libraries:

```
sudo apt-get install libgomp1 zlib1g-dev
```

Solution supplied by: Oink.vrsc@

Note: revision date 2021-12-01.



Options available to the Verusd coind daemon.

[Important General Information](#)

[Verus , Chaindata & standard locations](#)

[General Options:](#)

[Index options:](#)

[Connection options:](#)

[Wallet options:](#)

[ZeroMQ notification options:](#)

[AMQP 1.0 notification options:](#)

[Debugging/Testing options:](#)

[Node relay options:](#)

[Block creation options:](#)

[Mining options:](#)

[PBaaS options:](#)

[RPC server options:](#)

[Metrics Options \(only if -daemon and -printtoconsole are not set\):](#)

[General options:](#)

[Debugging/Testing options:](#)

[Node relay options:](#)

[RPC options:](#)

Important General Information

[[↑](#)]

Verus Daemon version 1.2.0-1

Usage: `verusd [options]` Start Verus Daemon

The options can be issued from the command line as shown above, but they can also be stored in the `VRSC.conf` file.

The `VRSC.conf` file is loaded on the daemon startup and provides the standard configuration.

Options supplied at the command line will override any conflicting settings in the `VRSC.conf` file.

To use the options in the `VRSC.conf` file, omit the leading `-` -character.

Example of a `VRSC.conf` -file:

```
rpcuser=user
rpcpassword=pass
rpcport=27486
server=1
txindex=1
rpcallowip=127.0.0.1
rpchost=127.0.0.1
addnode=195.253.48.236:27485
```

Verus `Wallet.dat`, Chaindata & `VRSC.conf` standard locations

[[↑](#)]

Linux: `~/.komodo/VRSC`

Mac OS: `~/Library/Application Support/Komodo/VRSC`

Windows 10: `%AppData%\Roaming\Komodo\VRSC\`

Options:

General Options:

[[↑](#)]

`-?`

This help message

`-alerts`

Receive and display P2P network alerts (default: 1)

`-alertnotify=<cmd>`

Execute command when a relevant alert is received or we see a really long fork (%s in cmd is replaced by message)

`-blocknotify=<cmd>`

Execute command when the best block changes (%s in cmd is replaced by block hash)

`-bootstrap`

Removes previous chain data (if present), downloads and extracts the bootstrap archive.

`-checkblocks=<n>`

How many blocks to check at startup (default: 288, 0 = all)

`-checklevel=<n>`

How thorough the block verification of -checkblocks is (0-4, default: 3)

`-conf=<file>`

Specify configuration file (default: komodo.conf)

`-daemon`

Run in the background as a daemon and accept commands

`-datadir=<dir>`

Specify data directory

`-dbcache=<n>`

Set database cache size in megabytes (4 to 16384, default: 450)

`-exportdir=<dir>`

Specify directory to be used when exporting data

`-loadblock=<file>`

Imports blocks from external blk000???.dat file on startup

`-maxorphantx=<n>`

Keep at most unconnectable transactions in memory (default: 100)

`-mempooltxinputlimit=<n>`

[DEPRECATED FROM OVERWINTER] Set the maximum number of transparent inputs in a transaction that the mempool will accept (default: 0 = no limit applied)

`-par=<n>`

Set the number of script verification threads (-4 to 16, 0 = auto, <0 = leave that many cores free, default: 0)

`-pid=<file>`

Specify pid file (default: verusd.pid)

`-prune=<n>`

Reduce storage requirements by pruning (deleting) old blocks. This mode disables wallet support and is incompatible with `-txindex`.

Warning Reverting this setting requires re-downloading the entire blockchain. (default: 0 = disable pruning blocks, >550 = target size in MiB to use for block files)

`-reindex`

Rebuild block chain index from current blk000???.dat files on startup

`-sysperms`

Create new files with system default permissions, instead of umask 077 (only effective with disabled wallet functionality)

Index options:

[[↑](#)]

`-addressindex`

Maintain a full address index, used to query for the balance, txids and unspent outputs for addresses (default: 1)

-idindex

Maintain a full identity index, enabling queries to select IDs with addresses, revocation or recovery IDs (default: 0)

!!! Activating requires reindexing, not compatible with bootstrap!!!

-timestampindex

Maintain a timestamp index for block hashes, used to query blocks hashes by a range of timestamps (default: 0)

!!! Activating requires reindexing, not compatible with bootstrap!!!

-spentindex

Maintain a full spent index, used to query the spending txid and input index for an outpoint (default: 1)

-insightexplorer

If enabled, forces addressindex, spentindex and timestampindex to enabled

If disabled, forces timestampindex to disabled

(default: 0)

!!! Activating requires reindexing, not compatible with bootstrap!!!

Connection options:

[[↑](#)]

-addnode=<ip>

Add a node to connect to and attempt to keep the connection open

-banscore=<n>

Threshold for disconnecting misbehaving peers (default: 100)

-bantime=<n>

Number of seconds to keep misbehaving peers from reconnecting (default: 86400)

-bind=<addr>

Bind to given address and always listen on it. Use [host]:port notation for IPv6

-connect=<ip>

Connect only to the specified node(s)

-discover

Discover own IP addresses (default: 1 when listening and no -externalip or -proxy)

-dns

Allow DNS lookups for -addnode, -seednode and -connect (default: 1)

-dnsseed

Query for peer addresses via DNS lookup, if low on addresses (default: 1 unless -connect)

-externalip=<ip>

Specify your own public address

-forcednsseed

Always query for peer addresses via DNS lookup (default: 0)

-listen

Accept connections from outside (default: 1 if no -proxy or -connect)

-listenonion

Automatically create Tor hidden service (default: 1)

-maxconnections=<n>

Maintain at most connections to peers (default: 384)

-maxreceivebuffer=<n>

Maximum per-connection receive buffer, * 1000 bytes (default: 5000)

-maxsendbuffer=<n>

Maximum per-connection send buffer, * 1000 bytes (default: 1000)

-onion=<ip:port>

Use separate SOCKS5 proxy to reach peers via Tor hidden services (default: -proxy)

-onlynet=<net>

Only connect to nodes in network (ipv4, ipv6 or onion)

-permittbaremultisig

Relay non-P2SH multisig (default: 1)

-peerbloomfilters

Support filtering of blocks and transaction with Bloom filters (default: 1)

-port=<port>

Listen for connections on (default: 7770 or testnet: 17770)

-proxy=<ip:port>

Connect through SOCKS5 proxy

-proxyrandomize

Randomize credentials for every proxy connection. This enables Tor stream isolation (default: 1)

-seednode=<ip>

Connect to a node to retrieve peer addresses, and disconnect

-timeout=<n>

Specify connection timeout in milliseconds (minimum: 1, default: 5000)

-torcontrol=<ip>:<port>

Tor control port to use if onion listening enabled (default: 127.0.0.1:9051)

-torpassword=<pass>

Tor control port password (default: empty)

-tlsdisable=<0 or 1>

Disable TLS connections. (default: 0)

-tlesenforcement=<0 or 1>

Only connect to TLS compatible peers. (default: 0)

-tlsfallbackontls=<0 or 1>

If a TLS connection fails, the next connection attempt of the same peer (based on IP address) takes place without TLS (default: 1)

`-tlsvalidate=<0 or 1>`

Connect to peers only with valid certificates (default: 0)

`-tlskeypath=<path>`

Full path to a private key

`-tlskeypwd=<password>`

Password for a private key encryption (default: not set, i.e. private key will be stored unencrypted)

`-tlscertpath=<path>`

Full path to a certificate

`-tlstrustdir=<path>`

Full path to a trusted certificates directory

`-uacomment=<string>`

Set a User Agent to your daemon. (default: disabled/none)

Shows as appended text in the `"subver"` string on connected nodes

Using this option has implications on your anonymity!

`-whitebind=<addr>`

Bind to given address and whitelist peers connecting to it. Use

[host]:port notation for IPv6

`-whitelist=<netmask>`

Whitelist peers connecting from the given netmask or IP address. Can be specified multiple times. Whitelisted peers cannot be DoS banned and their transactions are always relayed, even if they are already in the mempool, useful e.g. for a gateway

Wallet options:

[[↑](#)]

`-arbitragecurrencies`

Either a JSON array or a comma separated list of currency names.

`-arbitrageaddress`

A valid wallet address or identity controlled by this wallet that will hold the arbitrage currencies to use.

`-cheatcatcher=<sapling-address>`

same as `-defaultzaddr`

`-defaultid=<i-address>`

VerusID used for default change out and staking reward recipient

`-defaultzaddr=<sapling-address>`

sapling address to receive fraud proof rewards and if used with `"-privatechange=1"`, z-change address for the `sendcurrency` command

`-disablewallet`

Do not load the wallet and disable wallet RPC calls

`-keypool=<n>`

Set key pool size to (default: 100)

`-maxtxfee=<amt>`

Maximum total fees (in VRSC) to use in a single wallet transaction; setting this too low may abort large transactions (default: 0.10)

-migration

Enable the Sprout to Sapling migration

-migrationdestaddress=<zaddr>

Set the Sapling migration address

-paytxfee=<amt>

Fee (in VRSC/kB) to add to transactions you send (default: 0.0001)

-privatechange

directs all change from sendcurrency or z_sendmany APIs to the defaultzaddr set, if it is a valid sapling address

-rescan

Rescan the block chain for missing wallet transactions on startup

-salvagewallet

Attempt to recover private keys from a corrupt wallet.dat on startup

-sendfreetransactions

Send transactions as zero-fee transactions if possible (default: 0)

-spendzeroconfchange

Spend unconfirmed change when sending transactions (default: 1)

-txconfirmtarget=<n>

If paytxfee is not set, include enough fee so transactions begin confirmation on average within n blocks (default: 2)

-txexpirydelta

Set the number of blocks after which a transaction that has not been mined will become invalid (min: 4, default: 20 (pre-Blossom) or 40 (post-Blossom))

-upgradewallet

Upgrade wallet to latest format on startup

-wallet=<file>

Specify wallet file (within data directory) (default: wallet.dat)

-walletbroadcast

Make the wallet broadcast transactions (default: 1)

-walletnotify=<cmd>

Execute command when a wallet transaction changes (%s in cmd is replaced by TxID)

-zapwallettxes=<mode>

Delete all wallet transactions and only recover those parts of the blockchain through -rescan on startup (1 = keep tx meta data e.g. account owner and payment request information, 2 = drop tx meta data)

ZeroMQ notification options:

[↑]

-zmqpubhashblock=<address>

Enable publish hash block in

`-zmqpubhashtx=<address>`

Enable publish hash transaction in

`-zmqpubrawblock=<address>`

Enable publish raw block in

`-zmqpubrawtx=<address>`

Enable publish raw transaction in

AMQP 1.0 notification options:

[[↑](#)]

all AMQP support options require `-experimentalfeatures`.

`-amqppubhashblock=<address>`

Enable publish hash block in

`-amqppubhashtx=<address>`

Enable publish hash transaction in

`-amqppubrawblock=<address>`

Enable publish raw block in

`-amqppubrawtx=<address>`

Enable publish raw transaction in

Debugging/Testing options:

[[↑](#)]

`-debug=<category>`

Output debugging information (default: 0, supplying is optional). If is not supplied or if = 1, output all debugging information. can be: addrman, alert, bench, coindb, db, estimatefee, http, libevent, lock, mempool, net, partitioncheck, pow, proxy, prune, rand, reindex, rpc, selectcoins, tor, zmq, zrpc, zrpunsafe (implies zrpc).

`-experimentalfeatures`

Enable use of experimental features

`-help-debug`

Show all debugging options (usage: --help -help-debug)

`-logips`

Include IP addresses in debug output (default: 0)

`-logtimestamps`

Prepend debug output with timestamp (default: 1)

`-minrelaytxfee=<amt>`

Fees (in VRSC/kB) smaller than this are considered zero fee for relaying (default: 0.000001)

`-printtoconsole`

Send trace/debug info to console instead of debug.log file

Node relay options:

[[↑](#)]

`-datacarrier`

Relay and mine data carrier transactions (default: 1)

`-datacarrier-size`

Maximum size of data in data carrier transactions we relay and mine
(default: 10000)

Block creation options:

[[↑](#)]

`-blockminsize=<n>`

Set minimum block size in bytes (default: 0)

`-blockmaxsize=<n>`

Set maximum block size in bytes (default: 2000000)

`-blockprioritysize=<n>`

Set maximum size of high-priority/low-fee transactions in bytes
(default: 1000000)

Mining options:

[[↑](#)]

`-defaultid=<i-address>`

VerusID used for default change out and staking reward recipient

`-equihashsolver=<name>`

Specify the Equihash solver to be used if enabled (default: "default")

`-gen`

Mine/generate coins (default: 0)

`-genproclimit=<n>`

Set the number of threads for coin mining if enabled (-1 = all cores,
default: 0)

`-mineraddress=<addr>`

Send mined coins to a specific single address

`-minetolocalwallet`

Require that mined blocks use a coinbase address in the local wallet
(default: 1)

`-miningdistribution={"addressorid":<n>,...}`

destination addresses and relative amounts used as ratios to divide
total rewards + fees

`-mint`

Mint/stake coins automatically (default: 0)

`-pubkey=<hexpubkey>`

If set, mining and staking rewards will go to this address by default

PBaaS options:

[[↑](#)]

`-acceptfreeimportsfrom=<i-address>,<i-address>,...`

"%s" no spaces - accept underpaid imports from these PBaaS chains or networks - default is empty

`-allowdelayednotarizations`

Do not notarize in order to prevent slower notarizations (default = %u, notarize to prevent slowing down)

`-alwayssubmitnotarizations`

Submit notarizations to notary chain whenever merge mining/staking and eligible (default = %u, only as needed)

`-approvecontractupgrade=<0xf09...>`

When validating blocks, vote to agree to upgrade to the specific contract. Default is no upgrade.

`-blocktime=<n>`

Set target block time (in seconds) for difficulty adjustment (default: 60)

`-chain=pbaaSchainname`

loads either mainnet or resolves and loads a PBaaS chain if not vrsc or vrsctest

`-miningdistributionpassthrough`

uses the same miningdistribution values and addresses/IDs as Verus when merge mining

`-notarizationperiod=<n>`

Set minimum spacing consensus between cross-chain notarization, in blocks (default: 10, min 10 min)

`-notaryid=<ID@>`

VerusID used for PBaaS and Ethereum cross-chain notarization

`-notificationoracle=<i-address>`

VerusID monitored for network alerts, triggers, and signals. Current default is "Verus Coin Foundation@" for Verus and the chain ID for PBaaS chains

`-powaveragingwindow=<n>`

Set averaging window for PoW difficulty adjustment, in blocks (default: 45)

`-testnet`

loads PBaaS network in testmode

RPC server options:

[[↑](#)]

`-server`

Accept command line and JSON-RPC commands

`-rest`

Accept public REST requests (default: 0)

`-rpcbind=<addr>`

Bind to given address to listen for JSON-RPC connections. Use [host]:port notation for IPv6. This option can be specified multiple times (default: bind to all interfaces)

`-rpcuser=<user>`

Username for JSON-RPC connections

`-rpcpassword=<pw>`

Password for JSON-RPC connections

`-rpcport=<port>`

Listen for JSON-RPC connections on (default: 7771 or testnet: 17771)

`-rpcallowip=<ip>`

Allow JSON-RPC connections from specified source. Valid for a single IP (e.g. 1.2.3.4), a network/netmask (e.g. 1.2.3.4/255.255.255.0) or a network/CIDR (e.g. 1.2.3.4/24). This option can be specified multiple times

`-rpcthreads=<n>`

Set the number of threads to service RPC calls (default: 4)

Metrics Options (only if `-daemon` and `-printtotoconsole` are not set):

[[↑](#)]

`-showmetrics`

Show metrics on stdout (default: 1 if running in a console, 0 otherwise)

`-metricsui`

Set to 1 for a persistent metrics screen, 0 for sequential metrics output (default: 1 if running in a console, 0 otherwise)

`-metricsrefreshime`

Number of seconds between metrics refreshes (default: 1 if running in a console, 600 otherwise)

Additional debug options:

These are options for developers to debug/test the chain or daemon. These options are **not** for general use on users daemons.

General options:

[[↑](#)]

`-enforcenodebloom`

Enforce minimum protocol version to limit use of Bloom filters (default: 1)

Debugging/Testing options:

[[↑](#)]

`-checkpoints`

Disable expensive verification for known chain history (default: 1)

`-dblogsize=<n>`

Flush database activity from memory pool to disk log every megabytes (default: 100)

`-disablesafemode`

Disable safemode, override a real safe mode event (default: 0)

`-test safemode`

Force safe mode (default: 0)

`dropmessagestest=<n>`

Randomly drop 1 of every network messages

`-fuzzmessagetest=<n>`

Randomly fuzz 1 of every network messages

`-flushwallet`

Run a thread to flush wallet periodically (default: 1)

`-stopafterblockimport`

Stop running after importing blocks from disk (default: 0)

`-nuparams=hexBranchId:activationHeight`

Use given activation height for specified network upgrade (regtest-only)

`-limitfreerelay=<n>`

Continuously rate-limit free transactions to *1000 bytes per minute (default: 15)

`-relaypriority`

Require high priority for relaying free or low-fee transactions (default: 0)

`-maxsigcachesize=<n>`

Limit size of signature cache to MiB (default: 40)

`-maxtipage=<n>`

Maximum tip age in seconds to consider node in initial block download (default: 86400)

`-printpriority`

Log transaction priority and fee per kB when mining blocks (default: 0)

`-privdb`

Sets the DB_PRIVATE flag in the wallet db environment (default: 1)

`-regtest`

Enter regression test mode, which uses a special chain in which blocks can be solved instantly. This is intended for regression testing tools and app development.

Node relay options:

[[↑](#)]

`-blockversion=<n>`

Override block version to test forking scenarios (default: 4)

RPC options:

[[↑](#)]

`-rpcworkqueue=<n>`

Set the depth of the work queue to service RPC calls (default: 16)

`-rpcservertimeout=<n>`

Timeout during HTTP requests (default: 30)

compiled by Oink.vrsc@.

Note: last revision date 2024-01-15.

Examples

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockchain", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlockchainIndex

Retrieves object containing the index and the height block of the provided

Block height to be used for the block.

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockchainIndex", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlockCount

Retrieves the number of blocks on the total valid block chain.

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockCount", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlockDetails

Retrieves information about the given block and its branches.

Arguments

1. `id` (int) (req) The block id

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockDetails", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlocksIndex

Retrieves object containing the index and the height block of the provided

Block height to be used for the block.

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlocksIndex", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlockIndex

Retrieves object containing the index and the height block of the provided

Block height to be used for the block.

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockIndex", "args": [{"id": 1, "order": 1, "type": "data", "value": "data1"}, {"id": 2, "order": 2, "type": "data", "value": "data2"}, {"id": 3, "order": 3, "type": "data", "value": "data3"}], "version": "0.1.0"}</code>
```

getBlockIndexRange

Retrieves array of blocks within the provided range provided

Arguments

1. `start` (int) (req) The block index

2. `end` (int) (req) The end block index

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockIndexRange", "args": [{"start": 1, "end": 3}], "version": "0.1.0"}</code>
```

getBlockIndexRange

Retrieves array of blocks within the provided range provided

Arguments

1. `start` (int) (req) The block index

2. `end` (int) (req) The end block index

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockIndexRange", "args": [{"start": 1, "end": 3}], "version": "0.1.0"}</code>
```

getBlockHeader

Retrieves block header for a given block id

Arguments

1. `id` (int) (req) The block header

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockHeader", "args": [{"id": 1}], "version": "0.1.0"}</code>
```

getBlockHeaderHash

Retrieves block header hash for a given block id

Arguments

1. `id` (int) (req) The block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlockHeaderHash", "args": [{"id": 1}], "version": "0.1.0"}</code>
```

getBlocksHeader

Retrieves block header hash for all blocks in the block tree, including the main chain and all explored branches.

Arguments

1. `id` (int) (req) The block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlocksHeader", "args": [{"id": 1}], "version": "0.1.0"}</code>
```

getBlocksHeaderHash

Retrieves block header hash for all blocks in the block tree, including the main chain and all explored branches.

Arguments

1. `id` (int) (req) The block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getBlocksHeaderHash", "args": [{"id": 1}], "version": "0.1.0"}</code>
```

getDifficulty

Retrieves the proof of work difficulty as a multiple of the mining difficulty

Arguments

1. `id` (int) (req) The block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getDifficulty", "args": [{"id": 1}], "version": "0.1.0"}</code>
```

getDifficultyRange

Retrieves the proof of work difficulty as a multiple of the mining difficulty

Arguments

1. `start` (int) (req) The block header hash

2. `end` (int) (req) The end block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getDifficultyRange", "args": [{"start": 1, "end": 3}], "version": "0.1.0"}</code>
```

getDifficultyRange

Retrieves the proof of work difficulty as a multiple of the mining difficulty

Arguments

1. `start` (int) (req) The block header hash

2. `end` (int) (req) The end block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getDifficultyRange", "args": [{"start": 1, "end": 3}], "version": "0.1.0"}</code>
```

getDifficultyRange

Retrieves the proof of work difficulty as a multiple of the mining difficulty

Arguments

1. `start` (int) (req) The block header hash

2. `end` (int) (req) The end block header hash

Result

```
1. <code> curl -X POST -H "Content-Type: application/json" -d '{"method": "getDifficultyRange", "args": [{"start": 1, "end": 3}], "version": "0.1.0"}</code>
```


getidentitycontent "name# || lid# (heightstart) (heightend) (toproofs) (vaddrkey)

Arguments:

- `"heightstart"` (string, required) new height by or at earliest of an identity
- `"heightend"` (string, required) definition, only return content from this height forward, inclusive
- `"toproofs"` (bool, optional) return proofs for the content, if true, then the proofs, otherwise, the content
- `"vaddrkey"` (string, optional) data key, if true, then also return the vaddrkey for this content
- `"lid#"` (string, optional) definition, more extensive search for specific content in ID

Result:

Examples:

```
[{"name": "getidentitycontent", "args": {"heightstart": "1", "heightend": "10", "toproofs": "true", "vaddrkey": "true"}, "data": "data binary: [\"Content\", \"1\", \"10\", \"true\", \"true\", \"getidentitycontent\", \"true\"]", "id": "1", "type": "getidentitycontent"}, {"name": "getidentityhistory", "args": {"height": "1 || lid# (heightstart) (heightend) (toproofs) (vaddrkey)"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"true\", \"true\", \"getidentityhistory\", \"true\"]", "id": "1", "type": "getidentityhistory"}, {"name": "getidentitytrust", "args": {"lids": "1 || 2"}, "data": "data binary: [\"Content\", \"1\", \"2\", \"getidentitytrust\", \"true\"]", "id": "1", "type": "getidentitytrust"}, {"name": "getidentity", "args": {"height": "1 || lid# (heightstart) (heightend) (toproofs) (vaddrkey)"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"true\", \"true\", \"getidentity\", \"true\"]", "id": "1", "type": "getidentity"}, {"name": "listidentities (includeSigned) (includeSignedIn) (includeWatchOnly)


Arguments:



- "includeSigned" (bool, optional, default=true) Include identities for which we can verify signatures
- "includeSignedIn" (bool, optional, default=true) Include identities that we can only sign for but not verify
- "includeWatchOnly" (bool, optional, default=false) Include identities that we can only sign for but not verify



Result:



```
[{"name": "listidentities", "args": {"includeSigned": "true", "includeSignedIn": "true", "includeWatchOnly": "false"}, "data": "data binary: [\"Content\", \"1\", \"true\", \"true\", \"false\", \"listidentities\", \"true\"]", "id": "1", "type": "listidentities"}, {"name": "recoveridentity "jnidentity" (returntos) (tokenrecover) (feesoffer) (sourcelfunds)

Arguments:

- "jnidentity" (string, required) the identity to be recovered from the transaction, if true, it will be recovered by this action, otherwise, it will be recovered by the feesoffer
- "returntos" (string, optional) definition, defines to whom to return the recovered identity, if false, the recovered identity will be used to pay the transaction fees
- "tokenrecover" (string, optional) definition, token used to pay for the recovered identity
- "feesoffer" (string, optional) definition, feesoffer used to pay for the recovered identity
- "sourcelfunds" (string, optional) definition, funds used to pay for the recovered identity

Result:

Examples:


```
[{"name": "recoveridentity", "args": {"jnidentity": "1", "returntos": "1", "tokenrecover": "1", "feesoffer": "1", "sourcelfunds": "1"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"1\", \"1\", \"recoveridentity\", \"true\"]", "id": "1", "type": "recoveridentity"}, {"name": "registeridentity "jnidentity" (returntos) (feesoffer) (sourcelfunds)


Arguments:



- "jnidentity" (string, required) the identity to be registered
- "returntos" (string, optional) definition, to whom to return the identity after creation
- "feesoffer" (string, optional) definition, feesoffer used to pay for the creation of the identity
- "sourcelfunds" (string, optional) definition, funds used to pay for the creation of the identity



Result:



Examples:



```
[{"name": "registeridentity", "args": {"jnidentity": "1", "returntos": "1", "feesoffer": "1", "sourcelfunds": "1"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"1\", \"1\", \"registeridentity\", \"true\"]", "id": "1", "type": "registeridentity"}, {"name": "registercommitment "name" ("referralidentity") ("parentnameid") ("sourcelfunds")

Arguments:

- "name" (string, required) the name to be registered, creating a new commitment
- "referralidentity" (string, optional) definition, the name of the identity to be used when registering on behalf of the name
- "parentnameid" (string, optional) definition, the name of the parent identity to be used when registering on behalf of the name
- "sourcelfunds" (string, optional) definition, funds used to pay for the creation of the commitment

Result:

Examples:


```
[{"name": "registercommitment", "args": {"name": "1", "referralidentity": "1", "parentnameid": "1", "sourcelfunds": "1"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"1\", \"1\", \"registercommitment\", \"true\"]", "id": "1", "type": "registercommitment"}, {"name": "recoveridentity "nameID" (returntos) (tokenrecover) (feesoffer) (sourcelfunds)


Arguments:



- "nameID" (string, required) the unique name to create to, creating a new commitment
- "returntos" (string, optional) definition, to whom to return the identity after creation
- "tokenrecover" (string, optional) definition, token used to pay for the creation of the commitment
- "feesoffer" (string, optional) definition, feesoffer used to pay for the creation of the commitment
- "sourcelfunds" (string, optional) definition, funds used to pay for the creation of the commitment



Result:



Examples:



```
[{"name": "recoveridentity", "args": {"nameID": "1", "returntos": "1", "tokenrecover": "1", "feesoffer": "1", "sourcelfunds": "1"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"1\", \"1\", \"recoveridentity\", \"true\"]", "id": "1", "type": "recoveridentity"}, {"name": "setidentityunlock "tag" ("unlockdelay" "absoluteunlockheight" || "returnlockdelay":numberofblocksdelayafterunlock) (returntos) (feesoffer) (sourcelfunds)

Arguments:

- "tag" (string, optional) definition, tag for an absolute unlock height, unlock starts when a tag is used
- "unlockdelay" (string, optional) definition, delay in blocks after which unlock is used
- "absoluteunlockheight" (string, optional) definition, absolute unlock height for the unlock
- "returnlockdelay" (string, optional) definition, number of blocks delay after which the unlock is used
- "returntos" (string, optional) definition, to whom to return the identity after unlock
- "feesoffer" (string, optional) definition, feesoffer used to pay for the transaction
- "sourcelfunds" (string, optional) definition, funds used to pay for the transaction

Result:

Examples:


```
[{"name": "setidentityunlock", "args": {"tag": "1", "unlockdelay": "1", "absoluteunlockheight": "1", "returnlockdelay": "1", "returntos": "1", "feesoffer": "1", "sourcelfunds": "1"}, "data": "data binary: [\"Content\", \"1\", \"1\", \"1\", \"1\", \"1\", \"1\", \"setidentityunlock\", \"true\"]", "id": "1", "type": "setidentityunlock"}, {"name": "setidentitytrust "cilverall" : bool, "setstrategies": {"id": "1": {"setminingobject": "..."}, "removeratings": {"id": "..."}, "identitytrustmode": "none"}


Arguments:



- "cilverall" (bool, optional) defines all wallet identity trust rules before a strategy, if true, then the wallet mode is set to none
- "setstrategies" (string, optional) definition, strategy for the wallet mode
- "removeratings" (string, optional) definition, remove ratings for the wallet mode
- "identitytrustmode" (string, optional) definition, identity trust mode, if true, then the wallet mode is set to none



Result:



Examples:



```
[{"name": "setidentitytrust", "args": {"cilverall": "true", "setstrategies": {"id": "1": {"setminingobject": "..."}, "removeratings": {"id": "..."}, "identitytrustmode": "none"}, "removeratings": {"id": "..."}, "identitytrustmode": "none"}, "data": "data binary: [\"Content\", \"1\", \"true\", \"1\", \"1\", \"1\", \"1\", \"setidentitytrust\", \"true\"]", "id": "1", "type": "setidentitytrust"}, {"name": "signdata "json"

Arguments:

- "json" (string, required) the json to be signed

```


```


```


```


```


```


```


```


or chain, all deposits will be under the control of that system or chain-only, not its independent currencies.

Arguments:

- 1. "currenciesname" (string, optional): full name or ID of underlying currency

Result:

1
1

Example:



Remarks on Windows command line formatting:

The CLI help shows the command format for Linux and MacOS. On the native windows command prompt ([cmd.com](#)) the formatting is different.

- For windows substitute the shown `'`-character with the `"`-character.
- For windows substitute the shown `"`-character with the `\"`-characters.

Note: this only applies to giving commands from the windows command prompt. Verus Desktop will accept commands as listed.

Created by Oink.vrsc@

Note: last revision date 2020-03-02.



How do I install the Verus CLI (command line interface) wallet on a brand new (hosted) Linux system?

Notice: Read it completely before use.

Important General Information

`wallet.dat` location on Linux: `~/.komodo/VRSC`

Necessary files:

Link 1: [Download latest Wallet](#)

Link 2: [Download Verus Bootstrap](#)

Procedure:

[↑]

1. First make sure your system is up to date:

```
sudo apt-get update && apt-get upgrade -y
```

2. I suggest not to use the root account. If you have not yet done, set up a new user

```
sudo adduser newusername  
sudo usermod -aG sudo newusername
```

3. Switch to new username

```
su - newusername
```

4. Test if your new user actually has root (SuDo) access, e.g.:

```
sudo ls -la /root
```

You should get some lines like this:

```
drwx----- 6 root root 4096 Jul 3 15:56
```

5. Download & install the wallet binaries:

```
wget https://github.com/VerusCoin/VerusCoin/releases/download/v0.9.3/Verus-CLI-Linux-v0.9.3-amd64.tgz
```

The downloaded archive contains another archive and a signature text file, enabling the archive within to be verified (You'll need a running wallet to do that)

Also: Verify the URL to the latest version from the [Download latest Wallet](#) above.

```
tar -xvf Verus-CLI-Linux-v0.9.3-amd64.tgz
```

Now extract the wallet archive:

```
tar -xvf Verus-CLI-Linux-v0.9.3-amd64.tar.gz
```

Change directory to verus-cli

```
cd verus-cli
```

Fetch parameters, takes time, more on slow Internet connection

```
./fetch-params
```

Creating the chaindata directory

```
cd ~
```

```
mkdir -p .komodo/VRSC
```

```
cd ~/.komodo/VRSC
```

Download the block-chain bootstrap, this considerably speeds up synchronisation of the block-chain from days

to minutes... (optional)

```
wget https://bootstrap.verus.io/VRSC-bootstrap.tar.gz
tar -xvf VRSC-bootstrap.tar.gz
```

Install libraries for Verus

```
sudo apt-get install libcurl3 g++-multilib -y
```

Install Tmux a terminal multiplexer with which you can run threads in the background see

<https://en.wikipedia.org/wiki/Tmux>

```
sudo apt-get install tmux -y
```

Start tmux:

```
tmux
```

Launch Verus Daemon with or without number of threads

(usually number of threads equals number of cores or double of that if the processor support hyper threading well)

```
~/verus-cli/verusd -gen -genproclimit
~/verus-cli/verusd -gen -genproclimit=24
```

Once mining is operational – again this may take some time –

you'll see: 256 mega hashes complete - working

then detach tmux

```
[ctrl]&b d
```

Disable login with Root User

(make sure your newly created user login works and has sudo rights)

```
sudo nano /etc/ssh/sshd_config
```

Find: PermitRootLogin yes

And set to

PermitRootLogin no

Apply new settings:

```
sudo systemctl restart sshd
```

(submitted by @karero, corrected by @Glennp, edited by Oink.vrsc@)

Note: last revision date 2022-08-19.



Guide to install Bootstrap for your Verus Wallet.

[Important General Information](#)

[Verus-Desktop Procedure](#)

[Verus CLI Procedure](#)

[Optional:](#)

[Manual Bootstrap Procedure](#)

Attention: Read it completely before using.

Important General Information

[[↑](#)]

Verus **Wallet.dat** , Chaindata & **VRSC.conf** standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help` , `Show Verus data folder (default)`

Tip: The easiest way is to copy the location above and paste it into your address bar of your file browser. Your operation system will accept the input, interpret where that location is and bring you there.

Verus-Desktop Procedure

[[↑](#)]

1. In Verus-Desktop, exit your profile, by selecting the exit icon at the top right.
2. Wait a minute or two, allowing the wallet to close completely in the background.
3. click `help` , `Bootstrap VRSC` . That opens up a new window.
4. Follow the instructions and when finished successfully, select your preferred profile and enter Verus Desktop.

Verus CLI Procedure

[↑]

1. Go to the folder where your daemon is extracted (standard `verus-cli`)
2. Shutdown `verusd` and wait for it to close completely
3. doubleclick `fetch-bootstrap` in your file browser.
4. Follow the instructions and when finished, start your `verusd` daemon as usual

Optional:

[↑]

Watch this video with an explanation how to accomplish the steps above: [Bootstrapping your wallet using the supplied script](#)

Manual Bootstrap Procedure

[↑]

Necessary files:

Link 1: [Download latest Wallet](#)

Link 2: [Download Verus Bootstrap](#)

Procedure:

1. Make sure your wallet is not active.
2. If you already had your wallet running, backup essential files:
 - a. Go to `VRSC Wallet location`
 - b. copy `wallet.dat` to a *SAFE* location
 - c. copy `VRSC.conf` to a *SAFE* location
 - d. Verify that both files are copied to your safe location
3. Make sure the latest version of your Wallet for Verus is installed
 - a. Download the latest Verus Wallet from link 1, supplied above.
 - b. Verify the SHA256 checksum & signature of your download, to verify you have an untampered installer.
 - c. extract the file you just downloaded to a suitable location.
On MacOS and Linux you will have extracted an **AppImage** which can be run directly. Windows users need to run the **installer**.
4. Installing the bootstrap:
 - a. Download the bootstrap from Link 2. (For Windows a zip archive is available to accommodate native extraction. Linux and MacOS users can use the tar.gz archive)
 - b. (Optional, but recommended) Verify the md5, sha256 or sha512 checksum and the signature of your download, to verify that you downloaded an untampered Bootstrap archive.
 - c. Remove all files and folders from `VRSC Wallet Location` except `wallet.dat`, `debug.log`, `VRSC.conf` and if applicable `VRSC-bootstrap.*`.
 - d. Extract the downloaded archive to `VRSC Wallet location`. Make absolutely sure the folders `blocks` and `chainstate` are extracted into the correct folder. If they end up in a different folder (eg. `VRSC-bootstrap`-folder) move them to `VRSC Wallet location`.
5. If you had a VRSC wallet running before, restore essential files:
 - a. Go to `VRSC Wallet location`
 - b. Verify that your `wallet.dat` is bigger than the one in this folder (if any is present)
 - c. copy `wallet.dat` from your *SAFE* location
6. Start your wallet

If you followed these steps, you will have installed/updated the latest version of a wallet for verus, made a backup of your wallet and installed the bootstrap. If desired you can remove the downloaded bootstrap archive to free up space on your hard drive.

Optional:

Watch this video with an explanation how to accomplish the steps above: [Bootstrapping your wallet manually](#)

Information compiled by Oink.vrsc@.

Note: revision date 2024-01-17.



Guide to verify a new wallet download.

[Important General Information](#)

[Using Verus-Desktop](#)

[Using CLI Wallet](#)

Attention: Read it completely before using.

Important General Information

[[↑](#)]

Wallet download page: <https://verus.io/wallet>

Using Verus-Desktop

[[↑](#)]

1. download the new version
2. extract the archive
3. verify the signature using the data in the `*.signature.txt` -file through your existing Verus-Desktop, *VerusID* tab, Verify Signed Data and choose to verify a file. Only continue when this verification returns True.
4. stop your Verus-Desktop wallet
5. install the verified installer (Windows).
6. start your wallet.

Using CLI Wallet

[[↑](#)]

1. download the new version
2. extract the archive
3. verify the signature using the data in the `*.signature.txt` -file with the command
`./verus verifyfile "address or identity" "signature" "filepath/filename"` command. Only continue when this verification returns True.
4. stop your verusdaemon verus stop
5. extract the verified archive to your current CLI-wallet location
6. start your wallet (verusd)

Compiled by: Oink@

Note: creation date 2020-11-11.



How-To: Backup my wallet?

Important General Information

Video:

Preferred method: Exporting your wallet

Alternate method: Backing up your wallet

Extra info for chains

Important General Information

[↑]

Video:

[↑]

Watch this video with an explanation how to accomplish the steps below: [Backup your wallet](#)

Verus `Wallet.dat`, `Chaindata` & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

Instruction Video

External YouTube link: [Backing up the Verus Desktop wallet](#)

Preferred method: Exporting your wallet

[↑]

Note: The filename you replace `<mywalletexport>` with, can only contain letters and figures, no other characters, so it **cannot** have a file-extension!*

Verus Desktop:

Go to `Settings`, `Coin Settings` and click `Export native wallet backup`.

Confirm that you want to export after reading the pop-up.

The green message will tell you where the backup is and what it's name is.

linux/MacOS CLI:

run `./verus z_exportwallet "<mywalletexport>"`

windows CLI:

run `verus z_exportwallet "<mywalletexport>"`

Attention: Pay attention to the feedback this command gives you: it will mention the location where the export file is saved.

The exported wallet should be a file called `<mywalletexport>`, standard in the same directory as your `wallet.dat`. Keep this file secure, it has your plaintext private keys. Verify that the file is there and isn't empty.

Alternate method: Backing up your wallet

[[↑](#)]

Note: The filename you replace `<DestinationFileName>` with, can only contain letters and figures, no other characters, so it **cannot** have an file-extension

Verus Desktop:

Go to `Settings`, `Coin Settings` and click the text box
type `run backupwallet "<DestinationFileName>"`

linux/MacOS CLI:

run `./verus backupwallet "<DestinationFileName>"`

windows CLI:

run `verus backupwallet "<DestinationFileName>"`

Attention: Pay attention to the feedback this command gives you: it will mention the location where the backup file is saved.

The backup wallet should be a file called `<DestinationFileName>`, standard in the same directory as your `wallet.dat`. Keep this file secure, it enables full access to all your addresses.

Verify that the file is present and that it is the same size as your `wallet.dat`.

Extra info for non-Verus chains

[[↑](#)]

Extra line in `<coin>.conf` required

Non-Verus chains like **Komodo** and its asset chains and **Zcash**, need this entry in the coins configuration file to specify the export directory, before you started your wallet.

`exportdir=<dir>`

For Komodo the base directory is `komodo`.

- Linux: `~/.Komodo`
- Mac OS: `~/Library/Application Support/Komodo`
- Windows 10: `%AppData%\Roaming\Komodo`

For Komodo asset chains it is a folder/directory in the `komodo` base directory (eg `komodo/PIRATE`) with the **official** coin designation.

For Zcash the base directory is `zcash` instead of komodo.

- Linux: `~/.Zcash`
- Mac OS: `~/Library/Application Support/Zcash`
- Windows 10: `%AppData%\Roaming\Zcash`

non-Verus chains Verus Desktop

- **before** executing the command in the `run ...` commands, select the appropriate coin in the top right corner.

Information compiled by Oink.vrsc@.

Note: revision date 2022-09-12.



How-To: Restore my wallet from a backup?

Important General Information

Procedure

Important General Information

[↑]

Verus `Wallet.dat`, Chaindata & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

`verus command "<userinput>"` needs to be entered literally, with `<userinput>` replaced by your specific userdata. So if the text directs you to use for example `"<Public Address>"`, you replace that (including the `<` and `>`) with the address, so it looks similar to this: `"RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r"`.

Instruction Video

External YouTube link: [Restoring the Verus Desktop wallet](#)

Procedure

[↑]

Using a backup of your `wallet.dat`.

1. Stop verusd. For Windows-Desktop or Agama, just exit and wait for it to close completely. For the linux cli run `./verus stop`, or for the windows cli run `verus stop`.
2. Once your wallet is finished closing copy the backup of your `wallet.dat` file from your backup location to the directory listed in the start of this document (see above).
3. Now restart your wallet by launching Verus Desktop, Agama or running verusd for the CLI.

Using a `walletexport` file.

Note: The filename you replace `<mywalletimport>` with, can only contain letters and figures, no other characters, so it **cannot** have an file-extension

Attention: The command `z_importwallet` triggers the wallet to rescan in order to make all transactions to the freshly imported wallet addresses visible.

Rescanning your wallet may take a considerable time, during which your wallet may not respond to other commands. Please be patient.

The `<PATH>` in the `z_importwallet` command needs to be the **full absolute** path to the file. replace `LOGINNAME` with the actual loginname.

Verus Desktop:

Go to `Settings`, `Coin Settings` and click `Import native wallet backup`.

Click `Choose file`, browse to your backup file, select it and click `Open`

Click `Import` to start the import process

Agama:

Go to settings, scroll to the bottom and click CLI, select VRSC in that section.

Then below type `z_importwallet "<PATH><mywalletimport>"` and click the button below to run it.

linux/MacOS CLI:

run `./verus z_importwallet "<PATH><mywalletimport>"`

Windows CLI:

run `verus z_importwallet "<PATH><mywalletimport>\\"`

Using individual seeds / WIF-keys

Importing individual keys is explained in detail in this wiki: [import your Lite wallet address into your native Verus Desktop](#).

Information compiled by Oink.vrsc@.

Note: revision date 2022-09-12.



Guide to change Verus-Desktop from Lite Mode to Native Mode.

Attention: Read completely before use.

Important General Information

Verus `Wallet.dat`, `Chaindata` & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

Necessary files & links:

Link 1: [Download Verus Bootstrap](#)

Link 2: [Import Lite wallet address in Verus Desktop native](#)

Link 3: [Checking the signature](#)

Procedure:

[[↑](#)]

1. Make sure you have your seed phrase and password you use to log into your Lite mode wallet available.
2. First of all make a notion of your address and balance of VRSC you have in your wallet, before closing Verus Desktop.
3. Make sure the latest version of Verus-Desktop is installed.
 1. Download the latest Verus-Desktop.
 2. Verify the signature of your download, so you have an untampered installer. [Link 3](#) or [Video](#)
 3. Run the file you just downloaded to install it.
4. Getting Verus-Desktop ready for Native mode:
 1. Start Verus-Desktop and enter your profile (if not loaded automatically).
 1. If present in your profile, `deactivate` Verus Lite.
 2. Click `+ Add Coin`, select **Verus** from the dropdown list and continue.
 3. Select **Native**, tick the option `bootstrap` and optionally tick the options `Start staking`, `Start Mining` and fill in the amount of threads to mine with.
 4. Click `Add Coin`. Verus-Desktop will add Verus as Native chain to your screens.
 2. You may get a red warning message about Zcash params. (Verus Desktop will detect if you have the necessary ZCash parameter files and download them if needed)
 3. As soon as the download is finished, Verus-Desktop will continue and bring you into your wallet. It will automatically start to synchronize the blockchain. Since we already put the majority of the chain in place, this will take just a few minutes.

5. Importing your existing Address:

- This procedure is described in detail in: [Import Lite wallet address in Verus Desktop native](#).

If you followed these steps, installed the bootstrap, switched from Lite to Native mode and imported your existing address into Verus-Desktop. You can now stake your balance and use Private (sapling) addresses.

Created by Oink.vrsc@

Note: last revision date 2023-06-03.



How to import your Lite wallet address into your native Verus Desktop?

Prerequisites

Converting Seed to WIF

Importing a single WIF for a public (transparent) address

Importing multiple WIFs in one batch for public (transparent) addresses

Importing a single WIF for a private address

Importing multiple WIFs in one batch for private addresses

Attention: Read it completely before using.

Important General Information

`verus command "<userinput>"` needs to be entered literally, with `<userinput>` replaced by your specific userdata. So if the text directs you to use for example `"<Public Address>"`, you replace that (including the `<` and `>`) with the address, so it looks similar to this: `"RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r"`.

This method is confirmed to work on Verus and all Komodo assetchains.

Prerequisites

[[↑](#)]

In order to convert your seedphrase into a Private key (WIF), you need to have a running native wallet first, that is fully synchronized. The earliest wallet that supports these functions is **Verus Desktop v0.6.4-beta-1**.

If needed, use this guide to quickly synchronize your wallet: https://wiki.verus.io/#!how-to/how-to_bootstrap.md

Converting Seed to WIF

[[↑](#)]

If you have a seed, you can retrieve your Private key (WIF) by having the Verus Desktop wallet convert it for you.

To convert your *seed phrase* in Verus Desktop, go to `settings` --> `Coin Settings`, select the chain you want to use in the top-right corner and enter the following command:

```
run convert passphrase "word_1 word_2 word_3 ... word_n"
```

Note: Make sure you replace `Word1 word2 word3 ... word_n` with the actual seedphrase (12 or 24 words) of the address you want to import!

You will receive a response **similar** to this:

```
{
  "walletpassphrase": "seedphrase",
  "address": "RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r",
  "pubkey": "02ffc2f4b071afdec631e3fb7d435a0047be14a81ea1a269e4206b0068c0c1fa6f",
  "privkey": "d899ed88e9ee2e90c2cf51cb47e7b4495ec1e1cb10763bb1c111b0bde48bf86c",
  "wif": "UwGb5KvGPfMUr1tu74Desjh87ZeJM4wq5goLyThcogeLifc5aJqT"
}
```

Copy that information and store it somewhere **SAFE**. With this information anyone having access to it will have full control over that address.

The 52-character string after "**wif**": that is shown, is what you want to import in the next step.

Importing a single WIF for a public (transparent) address

[[↑](#)]

To import your address, go to `settings` --> `Coin Settings`, select the chain you want to use in the top-right corner and enter the following command:

```
run importprivkey "<wif>" "" true
```

Replace `<wif>` with the actual **wif** you got from the `convert passphrase` command earlier.

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

Note: The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Importing multiple WIFs in one batch for public (transparent) addresses

[[↑](#)]

To import your address, go to `settings` --> `Coin Settings`, select the chain you want to use in the top-right corner and enter the following command for every WIF except for the final one:

```
run importprivkey "<wif>" "" false
```

Import the final WIF with this command:

```
run importprivkey "<wif>" "" true
```

The last command triggers the chain to rescan all addresses in your wallet, including all the addresses you just imported.

Replace `<wif>` with the actual **wif** (like the one you got from the `convert passphrase` command earlier).

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Importing a single WIF for a private address

[[↑](#)]

To import your address, go to `settings` --> `Coin Settings`, select the chain you want to use in the top-right corner and enter the following command:

```
run z_importkey "<wif>" "yes" 1
```

Replace `<wif>` with the actual **wif** you got from the `convert passphrase` command earlier.

Note: Don't use the WIF from the example above, but use the one from the CLI-interface in Verus Desktop.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Importing multiple WIFs in one batch for private addresses

[[↑](#)]

To import your address, go to `settings` --> `Coin Settings`, select the chain you want to use in the top-right corner and enter the following command for every WIF except for the final one:

```
run z_importkey "<wif>" "no"
```

Import the final WIF with this command:

```
run z_importkey "<wif>" "yes" 1
```

The last command triggers the chain to rescan all addresses in your wallet, including all the addresses you just imported.

Replace `<wif>` with the actual **wif** (like the one you got from the `convertpassphrase` command earlier).

Note: Don't use the **WIF** from the example above, but use the one from the **CLI-interface in Verus Desktop**.

The GUI wallet will not show any progress on the import and may give messages that the RPC daemon is not reacting. It will take quite some time for the process to finish in the background, especially if the address has many transaction on it.

Information compiled by Oink.vrsc@.

Note: revision date 2021-05-02.

Shielding Verus Coin via the CLI.

Attention: Read it completely before use.

Note: Shielding is no longer required for coinbase rewards after block 800200. Earlier timelocked coins will still need to be shielded in order to use them.

Important General Information

`verus command "<userinput>"` needs to be entered literally, with `<userinput>` replaced by your specific userdata. So if the text directs you to use for example "`<Public Address>`", you replace that (including the `<` and `>`) with the address, so it looks similar to this: `"RYX6RYU3AAvwVCNyNM4cVyGUhSMUPvKs3r"`.

General remarks on CLI wallet:

On Windows command line enter the commands as shown without the surrounding quotation marks

In Linux shell preceed the commands without surrounding quotation marks with `./`

In MacOS shell preceed the commands without surrounding quotation marks with `./`

Example: the windows version `verus listtransactions` transforms in Linux or MacOS to `./verus listtransactions`.

General remarks on Windows command line formatting:

The CLI help shows the command format for Linux and MacOS.

For windows substitute the shown `'`-character with the `"`-character.

For windows substitute the shown `\`-character with the `\\"`-characters.

Procedure:

You must first "shield coins" (send from a transparent R-addr to a shielded zaddr) to be able to use them in staking, when they first unlock. This is part of the Zcash protocol itself. The commands below assume you are in the Verus source code directory

`verus z_shieldcoinbase`

You can either use an existing zaddr or use a new zaddr. To make a new zaddr:

`verus z_getnewaddress`

Use the address the above command outputs in the `z_shieldcoinbase` command

To shield all coinbase in your wallet, you can use `"*"` (quotes are important) and zaddr that is getting the funds:

`verus z_shieldcoinbase "*" <YOUR zs-ADDRESS>`

To just shield a single address, specify that as the first argument:

`verus z_shieldcoinbase <YOUR R-ADDRESS> <YOUR zs-ADDRESS>`

Once the funds have moved to the zaddr and are confirmed, you can freely send them to any address, They will be eligible for staking id sent to a transparent address (R-address).

To send from a certain zaddr to a transparent address, use `z_sendmany`. The following command sends 10 Verus to a given transparent address, ID address or ID-name.

Example:

`verus z_sendmany zcZpfuzzJqmN3fUJekvbnyuxuJe9eAURAHRMCvN2Nr7VuWjakb1LEw6j2etPcCnr45BRot7MaMbipuS5da162BfuUkFGxx '[{"amount":10,"address":"Verus Coin Foundation@"}]'`

Note: revision date 2020-02-12.



How-To Join VRSC testnet.

Attention: Read it completely before using.

Important General Information

VRSCTEST data location :

Linux GUI: `~/.komodo/VRSCTEST`

Mac OS: `/Users//Library/Application Support/Komodo/VRSCTEST`

Windows 10: `%AppData%\Roaming\Komodo\VRSCTEST\`

General remarks on CLI wallet:

On Windows command line enter the commands as shown without the surrounding quotation marks

In Linux shell preceed the commands without surrounding quotation marks with `./`

In MacOS shell preceed the commands without surrounding quotation marks with `./`

Example: the windows version `verus listtransactions` transforms in Linux or MacOS to

`./verus listtransactions` .

General remarks on Windows command line formatting:

The CLI help shows the command format for Linux and MacOS.

For windows substitute the shown `'` -character with the `"` -character.

For windows substitute the shown `"` -character with the `\"` -characters.

Necessary files:

Link 1: [Download latest Wallet](#)

Procedure:

[]

Joining Verus testnet to test the latest capabilities before they are released to mainnet or simply test if your goals are possible without spending VRSC (testnet coins hold no value) is easy.

Download a wallet

The first thing you need is a VRSC wallet. The CLI-wallet and Verus Desktop GUI wallet are available on the link above for Windows,

Linux and MacOS. If you already have a wallet verify that the wallet is the most recent version and update if needed.

Verus Desktop Wallet

1. Start your Verus Desktop wallet.

2. If you have never run Verus testnet on your system before:

1. Go to `settings` (cogwheel icon) and select `General Settings`.

2. Select `Enable VRSCTEST`.

3. Click `Save Changes`

4. Restart Verus desktop

3. When logged in, click `Add Coin`, select `Verus Testnet` and click `Continue`.

4. Select the startup parameters you desire (Native (Lite mode is not available on testnet), Staking, mining (specify number of threads), reindex blockchain and/or rescan wallet) and click `Add Coin`.

CLI wallet

1. start CLI walletdaemon using these parameters:

```
verusd -chain=VRSCTEST
```

Any extra parameter that you are used to for VRSC (like `-mint` or `-pubkey=`) can be appended as well.

2. commands through the CLI are in the following format:

```
verus -chain=VRSCTEST
```

The only difference with the normal VRSC chain is the `-chain=VRSCTEST` option, that is added.

Created by Oink.vrsc@, inspired by 0x03.vrsc@.

Note: revision date 2020-11-11.

Consensus 2024 — Verus Showcases Fully Completed PBaaS Blockchain Technology

Verus showcases its fully completed Public Blockchain as a Service (PBaaS) technology at Consensus 2024, May 29–31, Austin, USA (booth 1245). The global event for cryptocurrency and blockchain innovations this year. Here, tens of thousands of cryptocurrency enthusiasts gather to explore the newest advancements in the industry.



Max Theyse · Follow

Published in Verus Coin · 6 min read · Jan 31, 2024



The L0/1 Verus Protocol Solves Major Issues with Ethereum and

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

[more](#) are much easier to use (no programming needed) in practice to achieve most capabilities, if not all, compared to VM-protocols that use smart contracts. Yet Verus is still fully interoperable with Ethereum through the non-custodial, trustless and decentralized Verus-Ethereum Bridge with liquidity pool currency [\[more on the Bridge\]](#).

Showcasing Verus App-Development Opportunities at Consensus 2024

Get ready for a new era of decentralized application development. Unlike today's complex smart contract applications, requiring VM-based smart contracts that talk to user facing applications through server-side APIs and phishing-prone web-interfaces using public key hashes or rented names as identity, the Verus network is a Layer 1 (L1) and Layer 0 (L0) protocol with composable and powerful multi-chain core primitives, including identities, currencies, MEV-resistant liquidity baskets, updateable, even transferable ID-bound databases that can comprise NFTs and real world assets, all on a 100% fully decentralized, unlimited scale backbone, secured by the 50% proof-of-stake / 50% proof-of-work, 51% hash attack resistant Verus Proof of

Power (PoP) consensus and accessible securely from mobile clients or cloud-based web-applications.

Build applications in your favorite framework for clients and access the Verus network either through QR-codes and deep-links to a client-side wallet under user control or talk directly to data-indexed, blockchain nodes, capable of providing fast access to an unlimited number of user identities and their unlimited Verus Data Exchange Format (VDXF) indexable key-value database providing provable, optionally private, worldwide resolvable data at unlimited scale over an unlimited number of blockchains. All of this power, including inherent eCommerce and markets of all types are usable by applications, and unlike yesterday's Internet, application-owned data stays under application control, while user-owned data remains under user control and subject to voluntary, provable disclosure, protected by zero knowledge privacy, as it should always have been.

Write applications your way, your client frameworks, your languages, with links and RPC calls to the decentralized Internet of Value. Connect through one or more blockchains. Use, create, sell, and provision your application's permanent IDs for secure login, KYC, commerce, and provable data at scale, all on the rent-free, credibly neutral and fully decentralized Internet of Value.

Builders of applications, organizations, and businesses can use the Verus Protocol to provision IDs, manage and interact with users and user state at any scale, launch currencies (e.g. tokens, liquidity baskets [\[see docs\]](#)), even fully interoperable, independent, customizable worldwide blockchains with their own economics and fee structure, while sharing mining and economic power across the entire Verus Protocol network of unlimited PBaaS chains. All currencies, whether native currencies of a new blockchain, easily created and minted tokens, or MEV-resistant liquidity baskets are easy to launch and access with the protocol on Verus or any connected PBaaS chain. All of them, from the moment they are launched can be exported to Ethereum as ERC20s [\[see docs\]](#) or ERC721 currencies or any chain across the Verus ecosystem. All currencies on the network, including tokens, cross-chain currencies, and even liquidity baskets and their supplies are secured by miners and stakers according to protocol rules. On other blockchain networks, this level of security is applied to single native currencies, on Verus, all currencies on a chain are primitives known to and accounted for by the protocol, eliminating entire classes of smart contract risks that result when every contract reinvents a new way to account for its currency but has no systemic control.

With Verus, currencies are compatible with ERC20s. Verus applications can leverage, but do not require Solidity developers or VM-based programming. Additionally all ERC-20s can be permissionlessly bridged over from Ethereum to Verus [\[see docs\]](#), giving users and application builders the freedom to switch seamlessly between the Verus and Ethereum networks.

All currencies on the Verus network (also the ones bridged over from Ethereum) can be used with Verus DeFi, another primitive inherent to the protocol. Verus DeFi enables liquidity baskets of up to 10 underlying reserve currencies and can serve as a unique, self-balancing, MEV-resistant portfolio, a permissionless or permissioned ID registrar where fees go to the LPs of the basket, or an easy way to send from any reserve currency or the basket currency itself to any other without worrying about block builder reordering or MEV, as all conversions of a liquidity basket in any direction that are processed within one or more blocks are always solved simultaneously, no front and not back. Every conversion gets the same price in each direction with no spread and an ultra low fee (max. 0.05%), again, all part of the L1 consensus.

Whether you plan to build eCommerce enabled supply chains, massively scalable ownership-based games, real world asset eCommerce enabled networks, voting systems, identity management networks, communication networks, financial networks, 3D metaverses with identities, brands, and

ownership or just about any application for a better Web3 on the Internet of Value, you'll be left behind if you don't deep dive on Verus first.

Join us in the next crypto revolution. Verus, with truth and privacy for all!

Come to the Verus booth on May 29–31 at Consensus 2024. The worldwide community is happy to get you started with a free VerusID, the easy to use Verus Mobile or any of the wide range of protocol features.

Amazing Verus Community

The Verus community united to facilitate participation at Consensus 2024 by generously contributing VRSC, ETH, DAI, MKR, Bridge.vETH and BTC. A heartfelt thank you to all the contributors! ❤

The Verus community used the funds to secure a two-block booth at Consensus, along with other facilities that we can not disclose yet.

Numerous members of the Verus community are participating in Consensus 2024, and we invite you to be a part of it too! Let's create a significant presence and showcase to the world the wide range of capabilities of Verus. We're excited and believe that it will be an excellent opportunity for the growth and exposure of the Verus ecosystem!

...

Please consider donating for the Consensus 2024 effort to make it even more unforgettable.

BTC:

[1JrgohmxB618J13bDF1V71STavrqeaghSU](#)

ETH (& tokens):

[0xdC415012eA218402E58E0221E4F8EA1544973A63](#)

VRSC (or DAI.vETH, MKR.vETH, vETH, Bridge.vETH, or any other convertible currency on the Verus network):

[Consensus2024@](#)

...

[Grab your ticket here](#). 🚨 Get a 20% discount on your Consensus 2024 ticket if you use the "VERUSC24" code.

...

Try Yourself!

Look up the [complete command list here](#). Or look up [docs.verus.io](#) to use many API commands (e.g. [launching currencies, tokens & liquidity pools](#)).

Join the community. Learn about the protocol. Use Verus & build.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)



Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow



More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196

Q

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304

Q



Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346

Q

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259

Q



See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Benedict Neo in bitgrit Data Science Publication

Roadmap to Learn AI in 2024

A free curriculum for hackers and programmers to learn AI

Kallol Mazumdar in ILLUMINATION

I Went on the Dark Web and Instantly Regretted It

Accessing the forbidden parts of the World Wide Web, only to realize the depravity of...

11 min read · Mar 11, 2024

9.1K 99

8 min read · Mar 13, 2024

9.2K 184

Lists

Generative AI Recommended Reading

52 stories · 894 saves

Modern Marketing

103 stories · 523 saves

Alexander Parks

Multiplewindow3dscene: Cool Front-End Quantum Entanglemen...

Using Three.js to Create Interactive 3D Scenes

2 min read · Jan 8, 2024

65 1

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538 13

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2K 29

Artturi Jalli

I Built an App in 6 Hours that Makes \$1,500/Mo

Copy my strategy!

3 min read · Jan 23, 2024

15.5K 180

See more recommendations

Discord's Session Hijack Vulnerability and Verus Community's Response

Join new server:
verus.io/discord

verus

Discord's Session Hijack Vulnerability and Verus Community's Response



Michael Toutonghi · Follow

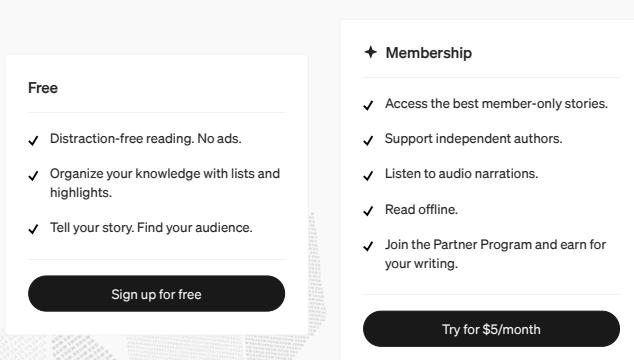
Published in Verus Coin · 4 min read · Dec 5, 2023

241



COPY FROM BELOW THIS LINE TO VERIFY SIGNATURE BELOW

One of the first and most important things to announce is that no funds on the Verus network or in Verus wallets were compromised due to this attack, as they were not subject to the vulnerabilities this hacker group exploited. We hope that the hackers were unable to fool anyone into giving up the funds from their Metamask or other Web wallets and that the criminals'



The image shows a modal window from Medium. It has two main sections: 'Free' on the left and 'Membership' on the right. The 'Free' section lists three benefits: 'Distraction-free reading. No ads.', 'Organize your knowledge with lists and highlights.', and 'Tell your story. Find your audience.' Below these is a 'Sign up for free' button. The 'Membership' section lists five benefits: 'Access the best member-only stories.', 'Support independent authors.', 'Listen to audio narrations.', 'Read offline.', and 'Join the Partner Program and earn for your writing.' Below these is a 'Try for \$5/month' button. The background of the modal features a faint, abstract geometric pattern of triangles and lines.

What we have determined so far:

1. The group targeted me and worked to establish trust as journalists for weeks, always persistent, never pushy. After working with me on an article, where I contributed the following content ([Verus Article from Hack](#)).
2. After they said they had completed their article, they worked for 2 days on getting me to accept a content release form, even rewriting it after I refused to agree to their first version.
3. Once they rewrote it, I clicked a link from their Discord server, and although they weren't able to get login credentials, they exploited a session hijack link, and immediately went to work.
4. After they had my account, which had setup the Discord server so many years ago, under their control, they kicked off all admins, installed their own

admins, and did something that Discord would have to help explain that got Discord to instantly ban my account, the only one besides Discord that could have kicked them off.

5. Since then, they started announcing scam links, claiming fraudulent giveaways with Ethereum tokens backed by Verus on websites that if someone with a Metamask or similar wallet engages with, there is a risk of taking assets from them on Ethereum or possibly other networks, not Verus.

6. We still have not heard from or gotten help from Discord Support since this event began, except that they banned my account, and to our knowledge the scammers are still in the old Verus Discord server attempting to scam an ever dwindling group of members who haven't left by now.

7. If you know anyone in the old Discord, please inform them to leave the Discord asap and not engage in any of the announcements, as they are made by impersonators, while people trying to warn are banned and the original admins are locked out of the server.

8. My old Discord ID is now disabled by Discord, and I no longer have access to our conversations. My new account is miketout.vrsc (this ID), which is not the owner of this Discord, and we have implemented a set of processes for all Discord admins to prevent any such event from occurring again.

Since the Verus Community is much more than a Discord Server, though we still haven't found a more suitable engagement platform for the way the community works, a number of people have worked to create and secure a new community Discord Server, and we welcome all Verus Community members and well meaning interested people to join.

Though we are still hoping to have a conversation with Discord about the hack and how they could improve their security to prevent this for all of their user communities in the future, we have taken steps independently to harden the Verus Discord community in a way that would prevent such an attack in the future, even if the link click session hijack exploit is not resolved. We also have even better security thoughts for the future around leveraging VerusID in easy, meaningful ways. If you have experience writing Discord bots and especially secure servers, please get in touch. Until then, we hope you will rejoin or join the Verus Community's Discord server and the worldwide, unlimited scale, fully decentralized blockchain protocol conversation.

Get an invite for the new Verus Discord Community Server:
<https://verus.io/discord>

COPY UNTIL THE END OF THE LINE ABOVE THIS ONE

This article is signed by my ID, mike@ on the Verus network with the following signature:

AX8OKwABQSBhYU0ShQsu6LSRVay8IY7avMhoy3ub8jhznZnlhD5nGRE4h4U5
MEWiPg+DXGJRSXd3bWou5TfNHn/zEPfbMA7d

(if you are using Safari or possibly another browser, you may need this signature, as the copy comes out a bit different):

AZAOKwABQSCaekvVN3CBbneot9AHby8S0+YCYevJDSfps3iaML8BF0j9PMdO
/37JYxHbuDy/R11hWe9tCI5AIjppcQhQ3hDy

You can verify my signature on any Verus node or lite mode API, by copying the article between the instructions above and entering the ID as mike@ and the signature above to check. You can also use this interface on the Verus website to verify:

Verify Signed Files, Messages and Hashes

Verify files, messages and hashes for their authenticity. Make sure they are signed by the correct VerusID.

241

+

↑



Written by Michael Toutonghi

16 Followers · Writer for Verus Coin

Follow



More from Michael Toutonghi and Verus Coin



Michael Toutonghi

AI for Us All or Just the Privileged Elite?

Humanity finds itself hurtling towards a future inevitably shaped by the evolution of the...

3 min read · Apr 11, 2023

411



196



Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304



45



Michael Toutonghi

What inspired the creation of Verus Coin, and how long has it been in...

Late 2017, I was working on a machine learning project...

3 min read · Dec 5, 2022

See all from Michael Toutonghi

See all from Verus Coin

Recommended from Medium

Angelgarcia

Headless HTB-Walkthrough

Artturi Jalli

I Built an App in 6 Hours that

Season4

Name: Headless

4 min read · Mar 25, 2024

4 1

Makes \$1,500/Mo

Copy my strategy!

3 min read · Jan 23, 2024

15.5K 180

+

Lists**Staff Picks**

609 stories · 873 saves

Stories to Help You Level-Up at Work

19 stories · 546 saves

Self-Improvement 101

20 stories · 1545 saves

Productivity 101

20 stories · 1437 saves

Khaledyassien in InfoSec Write-ups

How I Found Multiple XSS Vulnerabilities Using Unknown...

Hello, everyone. I hope you are well.

12 min read · Mar 5, 2024

1.7K 20

Henry N. Caga (hncaga)

Hacking the Giant: How I Discovered Google's Vulnerability...

Introduction:

7 min read · Mar 23, 2024

441 6

+

Mil Hoorna... in Change Your Mind Change Your ...

Stop Listening to Music, It Will Change Your Life

I stopped listening to music, find out what the results and benefits are in this post!

5 min read · Nov 23, 2023

14K 529

Kallol Mazumdar in ILLUMINATION

I Went on the Dark Web and Instantly Regretted It

Accessing the forbidden parts of the World Wide Web, only to realize the depravity of...

8 min read · Mar 13, 2024

9.2K 184

+

[See more recommendations](#)

Introducing Switch — The Gateway Currency to Switch Between Stablecoins on Verus

Switch is a decentralized currency fully backed by VRSC, DAI, USDC & EURC. Use it to switch between Verus and the different stablecoins. Or hold it while it accrues value from conversion fees.



Max Theyse · Follow

Published in Verus Coin · 7 min read · Mar 25, 2024

280



⌚ Switch launch block: 2,984,851 (Saturday 30 March, 2024).

Start preconverting to Switch now for a 7 day period. Read further to learn how.

Medium

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

PRECONVERSION DRAFT WHEN YOU PUT VRSC IN.

- After the preconversion period you can use Switch to convert between any of the reserves for a max. fee of 0.05%. Switch then becomes a MEV-resistant AMM. [Learn more about Verus DeFi](#).
- Switch accrues 50% of the conversion fees — the conversion fees stay in the reserves, increasing the value of Switch.
- The other 50% of the conversion fees go to the miners and stakers of Verus.
- There is a risk to holding Switch — USDC & EURC are distributed by the centralized company Circle.
- After the preconversion period the supply of Switch is dynamic. Switch gets minted when people convert to it from VRSC, DAI, USDC or EURC, and get burned when people convert back to the reserves.

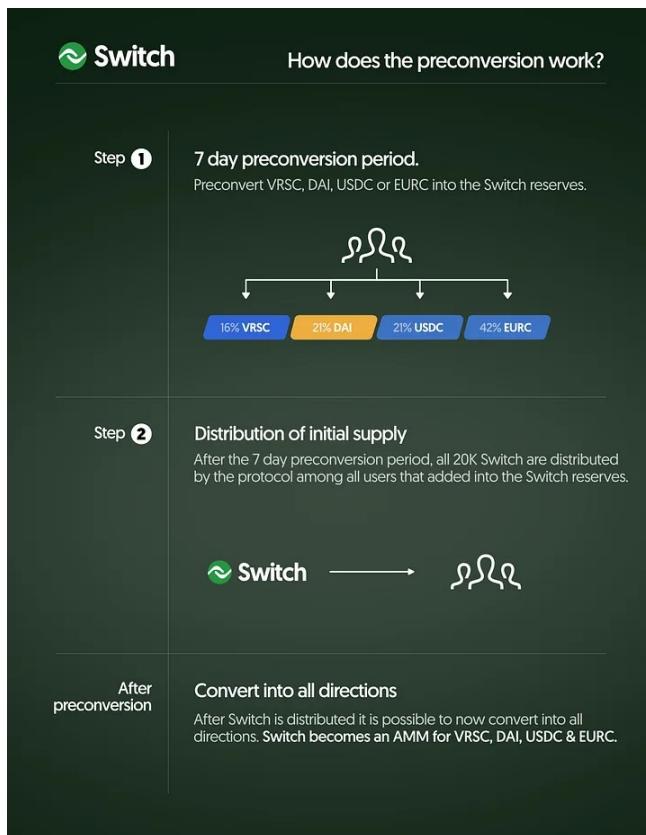
Decentralized currency backed by VRSC, DAI, USDC & EURC

Initial supply	20,000 Switch
Launch (after preconversion)	March 30, 2024
subID registration fees	10 EURC (in Switch)
subID registration fees are burned.	
CONVERSION FEES	
Switch \longleftrightarrow VRSC / DAI / USDC / EURC	0.025%
VRSC / DAI / USDC / EURC	0.05%
50% of the fees stay in the reserves, 50% go to block producers.	

- Register subIDs— [*.Switch@](#) for a 10 EURC (in Switch) fee. Those fees get burned. [Learn more about subIDs \(VerusIDs\)](#).
- Switch is perfect to use with [VerusPay](#)— Get your local businesses to take DAI, USDC & EURC on Verus. If you hold VRSC, DAI, or USDC and need to buy coffee in EURC, just scan a QR code! Verus scales to everyone worldwide and auto-converts payments from your favorite currency for always low transaction fees & a 0.05% conversion fee.

What is the preconversion period?

The preconversion period is a 7 day period in which users can fund the reserves of Switch with VRSC, DAI [DAI.vETH](#), USDC [vUSDC.vETH](#) & EURC [EURC.vETH](#). Depending on how much they added they will get their fair share of the 20,000 Switch after the 7 day period.



How to bridge USDC & EURC over to Verus

USDC and EURC are ERC-20 token stablecoins issued by Circle. Get them on [Coinbase](#) or other exchanges. On the Verus blockchain they are called [vUSDC.vETH](#) & [EURC.vETH](#).

DAI [DAI.vETH](#) is readily available on the Verus network. Below are the ERC-20

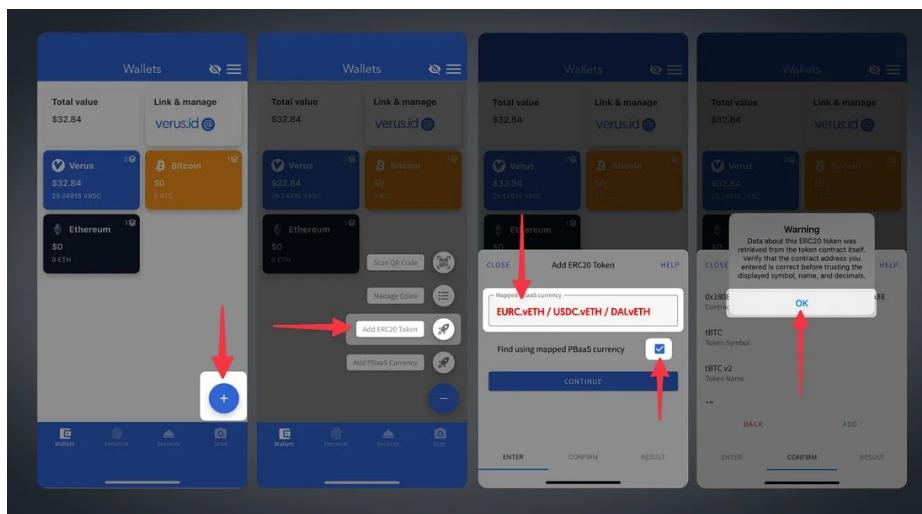
token contracts for the tokens:

- [USDC token contract](#)
- [EURC token contract](#)
- [DAI token contract](#)

To be able to use those ERC-20 tokens to preconvert into Switch they need to be bridged to the Verus blockchain. The bridging is done through the non-custodial and trustless Verus-Ethereum Bridge. Bridging and preconverting can be done with [Verus Mobile](#), [Verus Desktop](#) (full node) & MetaMask with [the bridge website](#).

Verus Mobile bridging

Let's start by bridging with Verus Mobile. First thing to do is to add the ERC-20s in Verus Mobile. This is how to do it:

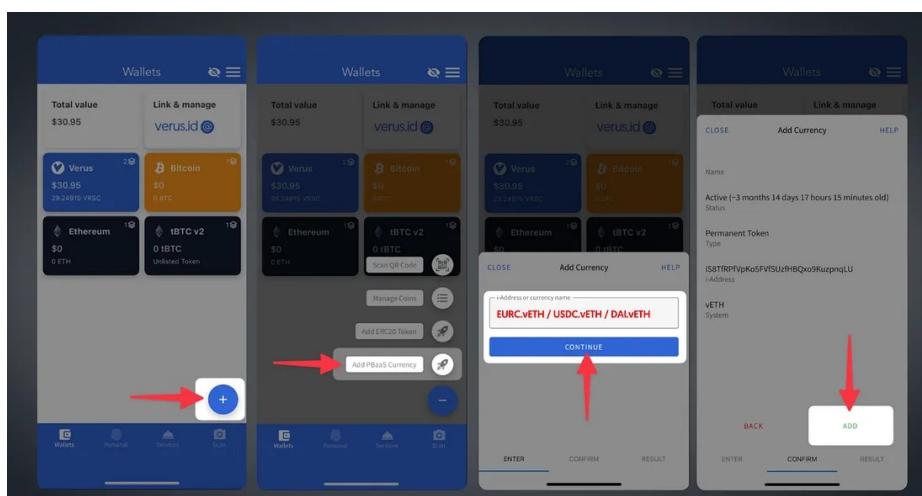


Adding ERC-20s to Verus Mobile.

- Press the plus-button bottom-right corner
- Press “Add ERC-20 Token”
- Select checkmark to “Find using mapped PBaaS currency”, and fill in `eurc.veth`, `usdc.veth` or `dai.veth`. Then press Continue.
- Press OK when a warning screen pops up.

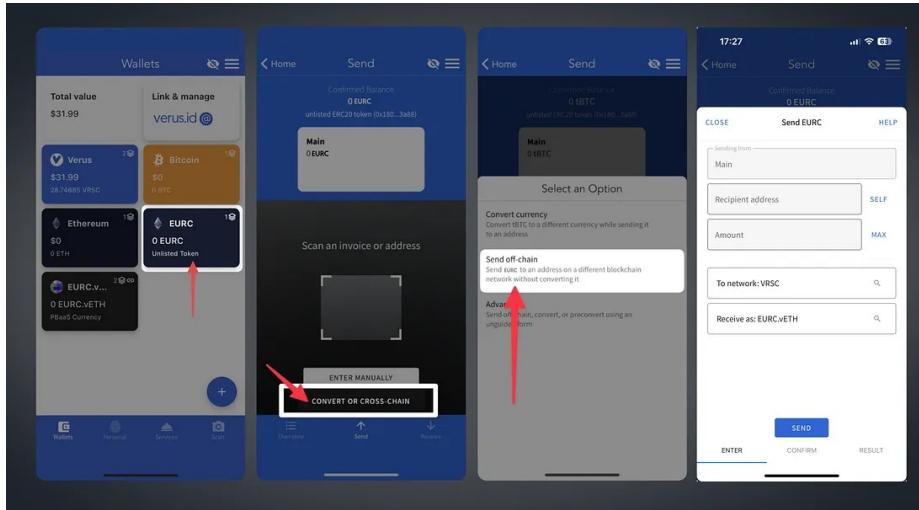
Now you are ready to receive USDC, EURC or DAI (as ERC-20s) on Verus Mobile.

Before bridging the tokens over to Verus let's add the currencies on the Verus blockchain to the wallet:



- Press the plus-button bottom-right corner.
- Press “Add PBaaS currency”.
- Fill in **eurc.veth**, **vusdc.veth** or **dai.veth** and press continue.
- Press “ADD”.

Now we are ready to bridge the ERC-20s over to the Verus blockchain.



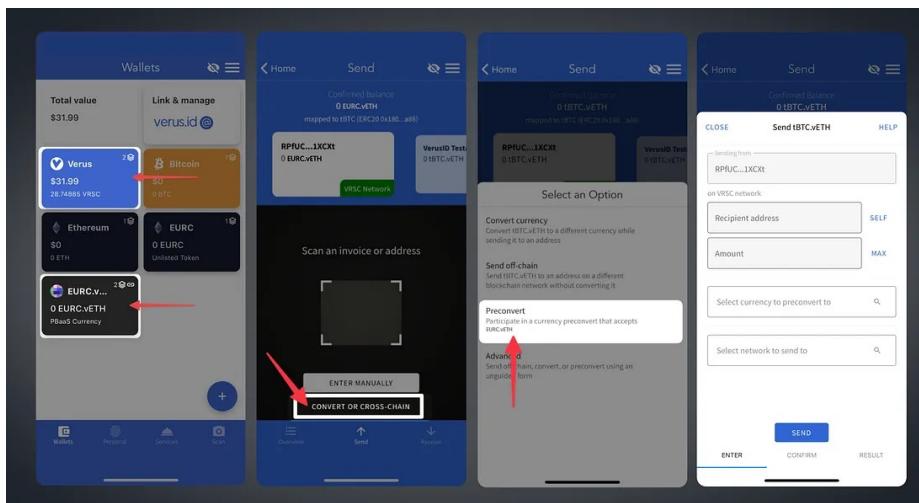
Bridging EURC to Verus. It will come out as EURC.vETH.

- Select the ERC-20 you want to bridge. (In the example we use EURC).
- Press “Send” and then “Convert or cross-chain”.
- Press “Send off-chain”.
- As the recipient address fill in a Verus address you own (R- or i-address, or VerusID).
- Fill in the amount you want to bridge over.
- Then as “Select network to send to” choose Verus
- Then as “Select currency to receive as” choose EURC.vETH

That's it. It can take up to 90 minutes before the ERC-20 arrives as a currency on Verus.

Verus Mobile preconverting

Choose any of the currencies you want to preconvert: **VRSC**, **DAI.vETH**, **vUSDC.vETH** OR **EURC.vETH**.



- Press **Verus**, **DAI.vETH**, **vUSDC.vETH** OR **EURC.vETH**
- Press “Send” and then “Convert or cross-chain”.
- Press “Preconvert”.
- As the recipient address fill in a Verus address you own (R- or i-address, or VerusID).
- Fill in the amount you want to preconvert.
- Then as “Select currency to preconvert to” choose Switch.
- The network to send to is Verus.
- Now press “Send” and confirm, and that’s it! You have now added a currency into the reserves of Switch and have to wait until Switch is out of the preconversion period.
- To show Switch in your wallet after it has been distributed by the protocol do the following: press the “plus”-button, “Add PBaaS Currency”, fill in “Switch”.

Bridging with MetaMask

Let’s bridge with MetaMask and the [Verus-Ethereum Bridge website](https://eth.verusbridge.io/):

Bridging with MetaMask to a Verus-address, EURC used as example.

- Connect MetaMask with this website: <https://eth.verusbridge.io/>
- Address: where you want the DAI, USDC or EURC to end up on the Verus blockchain (as **DAI.vETH**, **vUSDC.vETH** OR **EURC.vETH**)
- Select token: **[Euro Coin] as EURC.vETH**, **[0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48] as vUSDC.vETH** OR **DAI.vETH**
- Destination: select the same **... on VRSC**
- Choose the amount you want to bridge over
- It may take up to 90 minutes before the funds arrive on Verus

Preconverting with Verus Desktop

If you bridged currencies into your wallet on Verus Desktop don’t forget to add **DAI.vETH**, **vUSDC.vETH** OR **EURC.vETH** in the “Multiverse”-tab!

On Verus Desktop click the “Convert Currencies”-button and then go to the

“Advanced”-tab.

- Choose amount to send.
- “From Currency”: fill in `VRSC`, `DAI.vETH`, `vUSDC.vETH` OR `EURC.vETH`
- “To Currency”: fill in `Switch`
- “Destination Address”: fill in a R/i-address or VerusID where you want to receive Switch
- Check the box: “Send as pre-convert”
- “Refund Address”: Sometimes a currency has a minimum or maximum amount to preconvert before it launches. For example, when the minimum preconversion amount is not met, and thus the currency not launched, you will receive your funds back. In this case there isn’t a minimum or maximum but you still need to fill it in. Can be the same as your Destination Address.
- Now click “Convert Currencies”, and that’s it! You have now added a currency into the reserves of Switch and have to wait until Switch is out of the preconversion period.

... .

Join the crypto revolution with Verus & Switch

Switch is a unique currency used as a gateway to switch between stablecoins, launched on the Verus Protocol. [Join the Verus Discord](#) and explore what Verus has to offer. It is one of the most, if not the most, far ahead cryptocurrency protocols.

- [Verus solves scalability by neither sacrificing decentralization or security](#)
- [The Verus-Ethereum Bridge is non-custodial & trustless](#)
- [Launch currencies \(like Switch!\) without any coding needed](#)
- [Launch fully interoperable, independent and customizable PBaaS-blockchains](#)
- [VerusID: self-sovereign identities](#)
- [Verus DeFi: simple, low-cost, MEV-resistant and without any middleman](#)
- [Verus is the protocol for builders: join the community at Consensus 2024](#)

And so much more.

Join the community. Learn about the protocol. Use Verus. Get ahead of the game.

[!\[\]\(021ac0545a4a5d4616991eb0e5c5bdf9_img.jpg\) Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Development Blockchain Technology Defi Decentralized Finance
Cryptocurrency News

280



Follow



Written by Max Theyse

149 Followers · Editor for Verus Coin

More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196



304



Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346



Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259



See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538

13

Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781

4

+

Lists



Staff Picks

609 stories · 873 saves

Stories to Help You Level-Up at Work

19 stories · 546 saves

Self-Improvement 101

20 stories · 1545 saves

Productivity 101

20 stories · 1437 saves

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196

Q

Tony Aubé

Top 10 Ways You Will Lose All Your Money in Crypto

Crypto is popping these days. Everyone is looking for the next shitcoin to make million...

12 min read · Mar 8, 2024

492

7

+

Ahmad Rizwan

12 Lucrative Free Mining Apps To Make Some Serious Money in 2024

Discover the Top Cryptocurrency Mining Apps That Can Generate Profits Without...

5 min read · Mar 19, 2024

180

Q

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2K

29

+

See more recommendations

Bridging Done Right — Verus-Ethereum Bridge Launches Now!

The Verus-Ethereum Bridge is the first non-custodial and trustless bridge in the industry, and where the accounting of funds is proven and verified by consensus. And you can participate in its launch right now!



Max Theyse · Follow

Published in Verus Coin · 8 min read · Oct 12, 2023

359



Bridging Done Right

Non-custodial & consensus proven.



⌚ Verus-Ethereum Bridge launch: block height 2,758,800 (Fri 20 Oct, 2023).

Start preconverting to the bridge currency Bridge.vETH (with an initial supply of 100,000) NOW for a 10 day period. [➡️ Connect to the bridge](#)

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

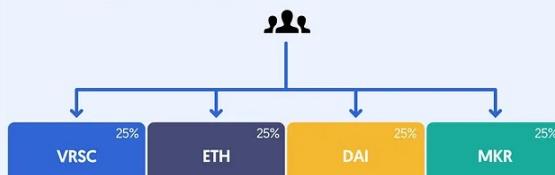
Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

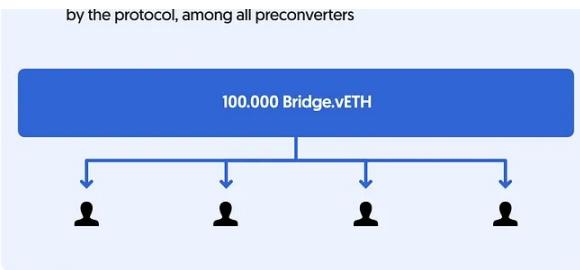
[Try for \\$5/month](#)

When the bridge currency launches on 20 Oct the initial supply of 100,000 will be distributed by the protocol among all preconverters.

- 10 day preconversion period
Preconvert VRSC and bridged ETH, DAI, MKR into the Bridge.vETH reserves



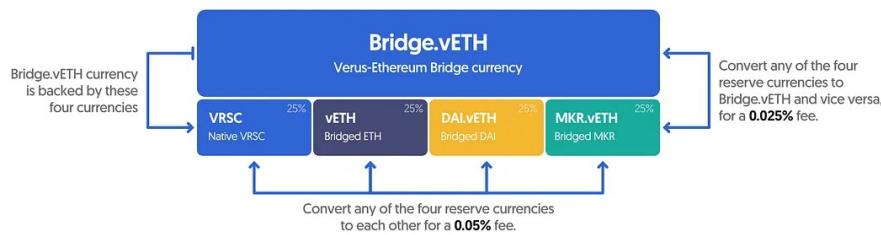
- Initial supply distributed
After the 10 days, the Bridge.vETH are distributed,



If you move ETH, DAI and MKR over to the Verus blockchain before the bridge is launched your cross-chain fees are subsidized. Around 5000 VRSC is used to subsidize the fees on the Verus side. When the bridge currency gets launched, the VRSC that is left will be put into the reserves.

Dynamic Supply

The Bridge.vETH supply is dynamic (after the preconversion has finished and the bridge currency is launched). The currency gets minted when people convert to it from VRSC, vETH, DAI.vETH and MKR.vETH, or gets burned when people convert out of it.



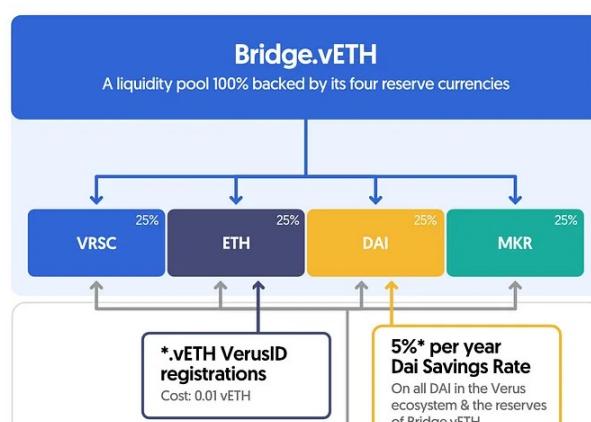
Worth noting: Verus DeFi is MEV-resistant. There is no front/back-running since all conversions are simultaneously solved within a block. [More details on docs.verus.io](#)

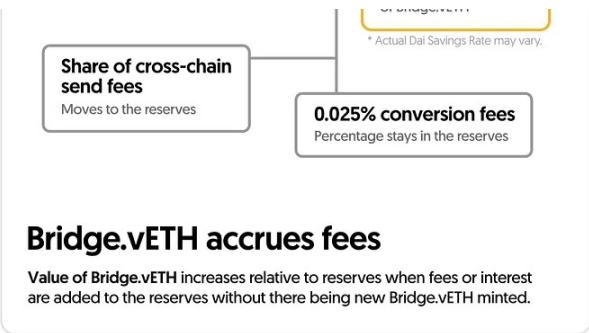
The Bridge.vETH currency function is to make the bridging of assets simple. From wherever you send it converts the fees that you need seamlessly.

The Bridge Accrues Fees

The bridge currency accrues fees with the cross-chain functions, conversions and .vETH VerusID registrations, making it more valuable the more it's used.

Dai holders get 5% interest (at the time of writing, the rate is subject to change by MakerDAO) automatically when holding in the DSR (Dai Savings Rate) contract. DAI in Bridge.vETH and in the complete Verus ecosystem get this savings rate. The DSR is being passed through 100% to the DAI reserves of Bridge.vETH.





Bridge.vETH accrues fees

Value of Bridge.vETH increases relative to reserves when fees or interest are added to the reserves without there being new Bridge.vETH minted.

Use the Bridge

After Bridge.vETH is launched and the initial supply is distributed you can now use the bridge as follows:

- Convert into any direction after the currency is launched and out of preconversion mode.
- Export launched currencies on Verus as an ERC-20. [Learn how](#)
- Launch currencies on Verus that are 1:1 mapped to any ERC-20. [Learn how](#)
- Send currencies, tokens and liquidity pools cross-chain.
- Send [ERC-721](#) & [ERC-1155](#) NFTs to Verus, and export tokenized ID currencies from Verus to Ethereum as ERC-721 & ERC-1155.

Present Issues with Existing Cryptocurrency Bridges

Let's first delve into the challenges associated with the prevailing cryptocurrency bridges in the industry.

- The custody of bridged assets is typically maintained by a single or a small number of addresses, effectively creating a central authority for holding the funds, which contradicts the decentralized nature of blockchain.
- The smart contracts that secure these bridged assets can be readily modified by a singular or limited number of authorities, thus presenting potential security issues.
- Smart contracts might contain bugs or malicious codes, making them susceptible to exploitation and undermining the safety of the bridged assets, especially if those assets are not accounted for by the block producers.

These problems have been demonstrated by numerous bridge hacks that have occurred and are likely to continue in the future. Verus does things, yet again, very different.

Why the Verus-Ethereum Bridge is Different

The Verus-Ethereum Bridge is different because the assets are never in anyone's custody. This is done through the seamless cooperation between the block producers, community notary witnesses, the Bridgekeeper software and the Ethereum smart contract. At each step during cross-chain transactions the assets are verified and proven by consensus rules, with safeguards in place to prevent hacks.

Every 10 blocks validators create a notarization. They create these digital receipts for both Verus and Ethereum. The digital receipts, called "notarizations", contain, among other things: the "stateroot" ([Merkle Mountain Range](#) for Verus, [Merkle Patricia Trie](#) for Ethereum), the blockheight, blockhash and the gas price for Ethereum. The notarizations have to be agreed to by the block producers (miners and stakers) and are then mined into the Verus blockchain.

[Read more here](#) on how the Verus Protocol handles cross-chain

Safeguards Against Bridge Hacks

Threats caused by malicious notary witnesses, or stolen keys to drain funds are not viable against the Verus-Ethereum bridge. To successfully mount an attack on the bridge, if a majority of witnesses were colluding or got their private keys stolen the following would need to happen:

1. Colluding, malicious witnesses.
2. Fake block producers with more combined hash and stake power than the publicly validated Verus blockchain.
3. Developers helping them by creating an alternate protocol for the shadow chain.

These requirements are very close to the requirements of attacking any blockchain. The bridge even provides a way to defend against such an unlikely scenario.

The notary witnesses are also monitoring notarizations, and if they were to sign for something that they themselves do not agree with, they can auto-revoke their identities, using the VerusID protocol, which cannot be stopped by an attacker unless they have stolen both the keys for the notary ID and those for its revocation ID as well. This serves as a prevention for stolen key attacks, ensuring that notaries are extremely hard targets to compromise.

Bridgekeeper

The two blockchains, Verus and Ethereum, are fundamentally different from each other, but still need to be able to communicate. A translator is needed. This translator is a program called Bridgekeeper. Notary witnesses have to run the program. Miners and stakers are also encouraged to run it, and for doing so they receive fees.

[Bridgekeeper for CLI](#) / [How to become a Bridgekeeper – video](#)

Verus Ethereum

Let's consider a scenario where a user wants to move or convert assets from Verus to Ethereum. This transaction gets recorded on the Verus blockchain. When there are enough, or a certain time period has passed, these cross-chain transactions are rolled up into something called a 'CCE' (cross-chain export). This CCE is then added to a block by a miner or staker and they are rewarded a fee for doing so. The CCE is now officially part of the blockchain, and is linked to the "stateroot" ([Merkle Mountain Range](#), which is an unchangeable overall summary of the blockchain at that block, and anything preceding it).

Meanwhile, as the transactions (CCE) are being mined (or staked) in, the notarizations are also being mined or staked in, confirmed by miners and stakers, and only then, signed by the notary witnesses. The notarizations can occur as often as every 10 blocks, depending on how many miners and stakers are running Bridgekeeper. Before a notarization is used as proof, it has to be on the blockchain for 10 blocks, confirmed by another notarization, and also signed by at least 8 of 15 notary witnesses worldwide.

Then when the notarization is confirmed on Verus, it's sent over to Ethereum and becomes pending on Ethereum. Once confirmed on Ethereum by receiving another notarization, the transactions (CCE) in that block are sent over to the Ethereum smart contract, along with the [Merkle Mountain Range](#) proof. If valid the assets are sent to the users Ethereum wallet.

Anyone can roll up the transactions into a CCE and mine it in to get a reward, but also the notary witnesses paying the ETH fees to send it to Ethereum get a share of the fees to subsidize their transaction costs.

If there is no traffic from Verus to Ethereum, the notarizations are continued to be mined in on Verus but do not need to be sent to Ethereum, so when the

traffic is light, there is minimal cost.

Ethereum Verus

The above part explains going from Verus to Ethereum, but the other way around is actually a little bit different. It is not the notary witnesses that send the group of transactions over, but any miner and staker can, and thus also get part of the fees.

Notarizations made on Verus always contain a stateroot from Ethereum that is a minimum of 30 blocks old (blocks on Ethereum take ~15 seconds) to make sure it is stable. Then after Verus has confirmed the notarization with a new one, Verus can accept transactions (CCE) from Ethereum, along with the [Ethereum Merkle Patricia Trie](#) proof that relates to the new confirmed data.

A Giant Leap Forward

The delayed sending of transactions is an important security mechanism that is not present in any other cross-chain bridge. To recap: before any real transactions happen, the Verus network sends proofs to the Ethereum network and vice versa. These proofs are checked and agreed upon by everyone in the network. Only then the actual moving of assets happens. Because of this delayed system, no single entity ever gets control over the assets. Also there are safeguards in place for the community notaries that protect users from hacks.

This system is decentralized, which means it's controlled by everyone participating in the network, rather than a single central authority. It's a giant leap forward for the entire cryptocurrency industry, and everyone can be a part of it!

...

Join the community. Learn about the protocol. Use Verus & build.

 [Connect to the bridge](#)

[Verus-Ethereum Bridge webpage](#)

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Try Yourself! 

Look up [docs.verus.io](#) to use many API commands (e.g. [launching currencies](#), [tokens & liquidity pools](#)).

Or look up the [complete command list here](#).

[Cryptocurrency News](#) [Blockchain Development](#) [Blockchain Technology](#)

[Cryptobridge](#) [Cryptocurrency](#)

 359





Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow



More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196



Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304



Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346



Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259



See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Desktop...

5 min read · Oct 12, 2023

146



Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538



Lists

Generative AI Recommended Reading

52 stories · 894 saves

Modern Marketing

103 stories · 523 saves

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4 4

Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781 4

4

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

143 4

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2K 29

4

See more recommendations

Verus Internet Protocol (VIP) — Provable, Decentralized Cross-chain Communication

Written by Mike Toutonghi, lead developer of the Verus Protocol.



Max Theyse · Follow

Published in Verus Coin · 13 min read · Aug 24, 2023

142



How it started

Bitcoin proved that decentralized, secure, permissionless and provable agreement on a worldwide blockchain network of participants is possible.

How it's going

Medium

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

companies and app designers, basically writing centralized apps. Instead of a secure set of underlying rules for currencies, we get best practices and the evolutionary legacy of thousands of “dApps”, which ironically, are mostly just partial or full re-implementations of the secure primitives missing in the consensus-based L1 protocols themselves.

As a result of the focus on pumping and market capture vs. evolving what we could learn from Bitcoin with sustainable, decentralized, more secure primitives and better protocol design, billions of dollars of value has been lost or transferred from naive early users who bought into the pumped crypto hype to hackers, scammers, VCs, centralized exits, and MEV-exploiters who all leveraged the mass of weaknesses they either failed to protect against or in some cases created and obscured themselves on systems that don’t define or enforce even the most common core primitives, currencies or identities. A great deal of loss has been due to obviously poor cross-chain “bridge” design, with the most common problem being bugs in complex smart contract workflow, followed by stolen or compromised private keys of contract owners / controllers, and in some cases, outright

theft by contract controllers.

The lack of agreement on basic primitives and how to represent them in a globally resolvable way has held back general, robust solutions to the fundamental problem of interoperability between blockchains and other financial networks today. To complicate our collective challenges, adding full primitive support in the protocols is hard work, especially in an industry where just a year of development is considered a massive investment that must be rewarded by timely exit liquidity. Consequently, an abundance of shortcuts and heavily marketed yet hastily built solutions still dominate the popular narrative, as hacks, rugpulls, and losses continue to pile up.

The solution

Verus was founded by technology veterans and others who believe humanity can do better than being exploited by the technology these same veterans had been building for corporations over decades. The Verus PBaaS protocol was designed, built, and tested over more than 5 years by a self-selected and often volunteer group of technology, finance, legal, operations, or just life veterans and newbies alike. Early Verusians shared a common vision and made it reality, achieving a fairly launched, community driven project without VCs or ICO participants who may want to exit and a truly decentralized unlimited scale network that all humanity can use and build upon without paying rent to anyone.

It has taken exactly as long as it has to realize the vision laid out 5 years ago, which is more clearly relevant today than ever. Along the way, that vision has grown and evolved as new community members contributed their own vision to help us improve, making the Verus PBaaS protocol, of which VIP is a small part, a major milestone for the crypto industry and a move back to the original intent of decentralized networks that was pioneered, if not fully realized by Satoshi Nakamoto in Bitcoin. Verus has taken Nakamoto consensus to the next level with consensus driven functional protocols for network primitives that are needed in any network hoping to securely enable financial applications.

To ensure maximum decentralization, the Verus Internet Protocol (VIP) takes a maximally decentralized, validator driven approach to its multi-network bridging by incorporating and accounting for currencies, identities, namespaces, and the concept of connected networks themselves in the L0 and L1 consensus protocols. The Verus Internet Protocol provides just enough structure to interoperate with any system that can control funds based on logic and exposes a provable representation of currencies and/or identities.

By grounding interoperability in a cryptographically provable, hierarchical namespace of self-sovereign, worldwide resolvable identities, VIP enables the first high-throughput, unlimited scale, decentralized, multi-protocol, non-custodial, auditable and transparent, currency and identity aware cross-blockchain communications protocol.

A cross-chain protocol with same chain properties

VIP can be seen as a cross-chain protocol that operates on proof of proof of consensus (PoPoC). VIP has the following properties:

1. Transactions that include cross-chain operations, cross-system currency transfers, or new currency and identity definitions, are standardized in the protocol, as are the semantics of their cross-chain use, fee models, cross-system, name-centric provable routing, and MEV-resistant bundling and rollup properties. These characteristics can be and have been implemented on UTXO, VM-based, and transparently centralized networks.
2. VIP supports direct or indirect system<->system communication, including cross-system export and import of currency and identity definitions, as well as revocation, recovery, and controlling keys. All cross-chain or DeFi functions in the protocol are accessible to full or lite node clients and applications by simply creating standard transaction

inputs and outputs that specify the operations. Available APIs on every node or FOSS API server simplify the creation of transactions with cross-chain outputs in response to command requests or JSON RPC calls.

3. Transactions with Identity, DeFi, or cross-chain operations, including all inputs and outputs, even exports and imports of all currencies, are checked for validity by miners and stakers with network consensus, just as single currency inputs and outputs are in today's single currency blockchain networks.
4. Cross-chain protocols operate in the same way as on-chain DeFi protocols and are processed on chain by miners and stakers with bundle->export->import workflow, enabling high-throughput DeFi and cross-chain operations.
5. In PBaaS, exports, imports, and DeFi operations are all solved simultaneously and processed with at per-block or larger granularity in an MEV-resistant manner as part of the core protocol, recognizing and enabling the need for beneficial arbitrage, while eliminating toxic MEV such as front-running and sandwiching once and for all.
6. All cross-chain operations are proven based on one of the following proof models, which are configurable on a per system/chain basis:
 - Decentralized consensus-achieved, witness enhanced, cross-chain cryptographic proofs. (auto-notarization)
 - Decentralized consensus-achieved, witness-dependent, cross-chain cryptographic proofs. (notary dependent notarization)
 - Semi-centralized or centrally controlled gateway proofs

Full auto-notarization is only available between Verus and PBaaS chains, and operates faster with but is not dependent on notary witnesses. When auto-notarization is applied to other gateway connections, the protocol will fall back to the closest it can come between auto-notarization and notary dependent, which may evolve towards less dependence on the notaries in future versions. Whether you auto-notarize or depend on notary witness confirmation, all forms of notarization operate similarly. We'll start with the basic workflow, then highlight differences:

1. “Earned” and “Accepted” notarizations – VIP broadly defines two types of cross-system notarizations, earned and accepted.
 - Generally, one side of a connection will create “earned” notarizations, which on PBaaS chains or for the Ethereum bridge, are created by miners and stakers adding a notarization output to the coinbase. Earned notarizations, which can only be created by Verus or PBaaS chain miners and stakers, include a way to agree or disagree with specific past notarizations, and are subject to further “confirmation” rules, which may vary depending on if the rules are “auto”, centralized, or “notary confirm”.
 - Once a new, earned notarization is confirmed, the proof of confirmation is sent to the “notary” system, which is typically a launch chain, and in the case of the ETH bridge, is Ethereum, along with any notary signatures necessary to prove confirmation, which are typically present. The new, confirmed, earned notarization is entered on the notary chain as an “accepted” notarization, and is entered as “agreeing” with a prior accepted notarization along with proof of the agreement. If the prior accepted notarization it agrees with is not yet confirmed, then it may be considered newly confirmed.
2. Once an earned notarization has been agreed to by a new earned notarization, alternating between PoW and PoS blocks to help further decentralized requirements for agreement, and by following all notarization rules, notary witnesses may sign signifying they agree with the notarization as well. Although they can disagree with a signature, no signature is also a valid way to disagree.

3. After validators and notary witnesses have agreed, or in the case of auto-notarization, a longer validation period with more agreement has passed, the prior notarization agreed with by a new notarization that has passed all necessary agreement, is confirmed. Once that happens, the blockchain is considered final up to the point of that confirmation and will resist reorganizations to any point behind.

The primary difference between notary confirm notarization and auto-notarization is the amount and type of evidence required to enter a potentially confirmed notarization. Most notably, “notary confirm” always requires a quorum of signatures from the specified notaries, whereas auto-notarization requires more multi-stage evidence in the face of randomized proof selection, potential fraud proofs and disagreements. Fully centralized gateways still operate on cryptographic proofs, but they only need notaries to agree on the cryptographic roots in order to consider them true and do not need cryptographic proof to establish the notarization proof root.

Once a proof root has been established by notarization, this root is used to cryptographically prove bundles of transactions exported from one chain or system and imported to another. Even Ethereum transactions are proven in both directions using the same workflow, although Ethereum transactions are validated using [PATRICIA Trie](#) proofs, while PBaaS chains use the Verus [Merkle Mountain Range \(MMR\)](#) method, both being cryptographically sound. Cross-chain bundles of transactions follow the same bundling protocol when sourced from Verus or PBaaS chains as is followed in DeFi to prevent MEV.

Transactions are always delivered to the destination network with bundles in the order they were created on the source network and according to the enforceable rules of the source network. Since a relayer may add arbitrage transactions when the destination is a PBaaS chain, but may not affect the provable source contents of a relayed bundle from one network to another, no restriction is placed on who relays a valid bundle of transactions. Since validators who relay and mine or stake transaction bundles into a block may earn a validation fee from the import transaction itself, validators or notaries will generally shuttle transaction bundles between chains and networks out of self interest, while having no ability whatsoever to censor or modify protocol level bundles.

What witness / notaries do NOT do

Witnesses / notaries have no control over what transactions are or are not included or the order in which they arrive from one system to another, as all of that is dictated by the protocol. By default, notaries cannot create notarizations and can only witness the validity of notarizations already created and agreed to by both proof of work and proof of stake validators.

Witness notaries’ primary role is to agree or disagree with the network validators’ earned notarizations, which are earned on one blockchain and accepted on another chain or system. As long as validators enter valid notarizations as part of the merge mining or staking process, confirm other notarizations entered by past validators, and a majority of notaries sign and witness their recognition of those events, a proof root that can be used by anyone to prove committed bundles of cross-chain transactions is established on both systems for the other. As the protocol ensures via cryptographic proof validation that the cross-chain bundles are the exact bundles in the order they were packaged on the source system, importing cross-chain transaction bundles is a permissionless operation that anyone, usually a validator, can perform, which relies on the proof established in relation to a confirmed, notarized proof root.

Do auto-notarization and notary-confirm notarization options have different security considerations?

Yes. Proper planning and selection of options and witness ID structure during the launch of a blockchain network or gateway can ensure that security levels meet network demands. There are security tradeoffs between auto-notarization and notary-confirm options, which basically are a tradeoff between a permissionless model where proof root determination is

enhanced, but not dependent upon multisig agreement of notary witnesses, auto-notarization vs. a model where establishing any cross-network cryptographic proof root requires the honest participation of a majority of the notary witnesses, which can be represented by revocable and recoverable IDs, allowing for further decentralization, even while requiring participation of a quorum.

Determining which option to choose, how many and what notaries to choose, what other safeguards can further enhance security, and then understanding the resulting security profile from each set of choices, is important when launching a blockchain or gateway. The following outlines the current high-level security considerations when configuring a new chain or gateway.

Currently, Verus VIP supports 2 proof models for interchain communication, the Verus PBaaS Merkle Mountain Range proofs, which also assume the Verus Proof of Power protocol, and Ethereum PATRICIA Trie proofs that enable interfacing with VIP between any PBaaS chain, and any Ethereum VM + PATRICIA Trie compatible network.

Since Verus PBaaS is the only interface to fully support notarization finalization using full auto-notarization, it is important to understand the differences between PBaaS and Ethereum proof protocols. At present, there is no objective proof protocol implementation for Ethereum that is known and implemented, which can operate without dependence on some set of notary witnesses. At this time, auto-notarization, when applied to Ethereum, is more similar to notary-confirm, with additional revocation safeguards that could address more edge-case threat models.

As new proof algorithms evolve for Ethereum, enabling trustworthy cross-chain proof models in both directions that are both timely and do not need to be backed by witness signatures, it will be possible to update the protocols to remove dependence on witnesses over time. To ensure maximum decentralization on the Ethereum bridge, the network itself can permissionlessly propose contract upgrades, and if affirmatively voted on by Verus validators, an upgrade and new contracts can eventually change the cross-chain proof conditions.

Auto-notarization of one PBaaS chain on another (PBAAS_AN) differs from and Ethereum interface or notary-confirm (NC) in that, if there are enough validated blocks, both mined and staked, as well as validator agreed notarizations, PBAAS_AN allows validators to finalize a notarization and submit it to the alternate chain. If there are disagreements about the correct notarization, the process becomes an on-chain proof competition, requiring each party to represent a growing, actual chain with both proof of work and proof of stake with an eventual winner. NC still has the miners and stakers create and earn from earned notarizations, but it also must have agreement from specified notaries to finalize any notarization and cannot finalize cross-chain proof roots without that agreement.

Improvement over common cross-chain protocols

Since Verus VIP does not ever have any identity, multisig or otherwise, hold custody over any funds when crossing from one decentralized network to another, even in the context of the Verus Ethereum contracts, threats caused by malicious notaries or stolen keys to drain funds on decentralized bridges or currencies are simply not viable against the Verus VIP protocol.

Cross-chain notarizations must first be entered by an earning validator, either PoW or PoS, and must also be agreed to by another validator, alternating between work and stake in eligibility. Only after that happens can notary witnesses apply their signature and agree to make that a confirmed notarization candidate. Still, what happens if a majority of notaries are malicious and colluding, or if some entity was able to somehow steal a majority of the notary private keys?

If the majority of notary witnesses are honest and operate their witnessing

nodes according to the protocol, the security of such a model should be sufficient for any level of cross-chain transaction proof. If, however, a majority of the notaries were colluding and malicious or if their keys were stolen, Verus VIP still provides ways to ensure that the protocol responds to such a situation without risk of loss. Although there are ways to mitigate such a form of attack, even if no mitigation were available, such malicious or fraudulent notary witnesses would need the following in order to attack a PBaaS cross-chain connection:

1. Colluding, malicious notaries
2. Fake validators with more combined hash + stake power than the publicly validated chain making earned notarizations
3. Developers helping them by creating an alternate protocol for the shadow chain

While you may recognize these requirements as very close to the requirements of attacking any blockchain, the Verus VIP protocol provides a way to even defend against such an unlikely scenario. VIP provides for revocation of all VerusIDs, including notary IDs, even those that are used for Ethereum protocol bridges. Furthermore, each PBaaS chain, including Verus, has a network agreed multisig chain oracle that can separately pause cross-chain notarizations or transactions temporarily, which also can be overridden by network validators.

To enable yet another layer of safeguards, the Verus VIP protocol does not confirm any notarization that has first been agreed to by validators, then agreed to and signed by all notaries on either the earned or the accepted chain, until a second such notarization confirms the first. This ensures that every notarization on any chain is first made public, given a period of time before confirmation, and finally, if nothing stops it, confirmed.

Meanwhile, with basic monitoring by witnesses to see if their own identities have signed for anything they do not actually agree with, meaning their keys were stolen, notaries can auto-revoke their own identities and prevent stolen key attacks from being anything more than an inconvenience. Additional monitoring by chain oracle controllers to look for clear discrepancies between soon to be confirmed notarizations and chains that they monitor can also use the oracle notification network to pause cross-chain transactions, providing yet another line of defense for each chain on the network.

While there are numerous capabilities and safeguards built into the Verus VIP protocol beyond those described in this document, by separating the proof root notarization process and using cryptographic proof as the foundation of a provable, high throughput, cross-chain export/import protocol, VIP enables the most secure, scalable, decentralized and multi-modal cross-chain protocol ever designed for blockchains and the Internet of Value.

...

Try the Verus Protocol Yourself!

Look up docs.verus.io to use many API commands (e.g. [launching currencies](#), [tokens & liquidity pools](#)).

Or look up the [complete command list here](#).

...

Join the community. Learn about the protocol. Use Verus & build.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Technology Blockchain Development Crosschain Cross Platform
Cryptocurrency

142   



Written by **Max Theyse**

149 Followers · Editor for Verus Coin

 Follow 

More from Max Theyse and Verus Coin

 Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196 

 Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304 



 Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346 

 Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259 



[See all from Max Theyse](#)

[See all from Verus Coin](#)

Recommended from Medium

👤 PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

💬 143



👤 Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Deskt...

5 min read · Oct 12, 2023

💬 146



🔖

Lists

Generative AI Recommended Reading

👤 Perzibal

52 stories · 894 saves

Modern Marketing

👤 MXS Games in MXS Games

103 stories · 523 saves

👤 Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

💬 4



👤 MXS Games in MXS Games

MXS Games Public Node Licence Sale Is Now LIVE!

In the true spirit of a fair launch, we are happy to announce that we will make a massive 95...

5 min read · Jan 29, 2024

💬 1



🔖

👤 Ilya Ermilov

HOT (Near Protocol) Mining and some hints

Hello friends!

8 min read · Feb 19, 2024

💬 41



👤 Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

💬 538



🔖

[See more recommendations](#)

Verus Smart Transactions vs. Smart Contracts

Verus Protocol lead developer Mike Toutonghi explains the differences between Verus smart transactions and smart contracts.



Max Theyse · Follow

Published in Verus Coin · 5 min read · Jul 25, 2023

121



The following was written by Mike Toutonghi, lead developer on the Verus Protocol, on [Discord](#)

Application Programming Model

The way to understand smart transactions vs. smart contracts is to think about the application programming model, and how each of them work in

Medium

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

or accounting rules whatsoever.

2. The differentiation of these contracts, which are akin to stored procedures in a database, but carry with them the DB schema as well as the interface, is in the way they enable differences of implementation of those accepted conventions.
3. Consensus across all of Ethereum is applied to ensuring that the shared, serialized, worldwide computer that is the EVM executes its low level instructions accurately, with no accounting of currencies, exchanges, enablement of efficient, actual IDs, zk-privacy, etc. The currencies that run above the single native security currency, run by arbitrary rules, often opaque or containing unexpected behaviors, and everyone must create their version of these contracts to build a dApp.

Verus Smart Transactions

In Verus, it's a new model as follows:

1. Your dApp can run on a client or on a broad range of resource types, and

it communicates via the same RPC API as the CLI commands (optionally without wallet or control functions) with a fully functional, decentralized backend, which is the Internet of Value, including all its unlimited scale, connected systems and chains.

2. Unlike a single currency blockchain, Verus smart transactions natively support an unlimited number of friendly name currencies directly via protocol and an unlimited number of blockchains for currency launches, simple sends, currency conversions, or cross-chain operations. All inputs, outputs, cross-chain imports, and even conversions are validated and in/out values of all currencies accounted for by consensus, just as native currency ins and outs are on single-currency blockchains.
3. Identities are a first class concept in the primitives available for dApps. They all support fully available, open, decentralized, user controlled authentication and authorization protocols, enabled by QR code or deep links and supportive of permission granting, ID provisioning/sale by the dApp, private/attested KYC and other services. These protocols can easily be bridged to OAuth or OpenID Connect via servers like Hydra, but used natively, they require no service provider between the service actually authenticating the user and the user themselves.
4. Every identity is multisig and has revocation and recovery capabilities. They are also transferable and can have rights bound into them in a provable manner via the contentmultimap. All of this is supported as primitives in the core RPC/CLI api/command line and in the core protocol.
5. IDs enable their holders to launch like-named currencies, blockchains, and single NFT-like tokens that have a super power over the ID, which is very useful for an ID that may live across multiple chains.
6. IDs and currencies may be created and launched on one chain, leveraging all of that chains capabilities (Verus includes many variations in currency, including conditional crowdfunding, liquidity pools, blockchains, etc.), and exported to other chains to be used everywhere across the network. For example, if you launch a currency on Verus or any PBaaS chain, including the native PBaaS chain's currency. All of those currency definitions can be exported to Ethereum, and they will emerge on Ethereum as an automatically created and functional ERC20 contract, enabling the currency to be sent back and forth via protocol from then on.
7. All identities and currencies are resolvable worldwide, meaning that you can determine a network path to the nodes of the blockchain where they were defined via their friendly name, or usually via their i-address, if they have been exported cross-chain. This enables apps to scale over any number of decentralized blockchains worldwide and leave the management of user databases and user data, as well as settings, signals, endpoints, etc. to the users themselves and the client apps they use to help them manage.
8. Because the currency and liquidity pool support is in the L1 protocol, we have been able to actually solve for and provide MEV-resistant (both intra and inter-block) AMMs as a primitive, available to all applications and users.

People don't have to write basic liquidity and conversion protocols, which are just primitives, not apps. Real apps can use all of these primitives to deliver the next level of function. Payments, currency conversions, earning systems, polls, voting, multi-chain world scale social networks using provable streams in provable IDs, crowdfunding, independent economies. All primitives, such as IDs, currencies, blockchains, and liquidity baskets have very rich capabilities from launch to continued operation.

Extending Verus Smart Transactions

If someone wants to extend smart transactions, that is an option, but they either don't have to, or if they really have a reason to do so, they can do it on their PBaaS chain, share or not share back (lots of reasons to do so), and now that we have the core primitive framework as a foundation, we can certainly

collaborate on a kind of protocol extension that is likely to include forms of VM and/or ZKVM extension points in the current protocol, to work within the framework of the core primitives that are fundamental to Verus and that we believe are crucial for safe financial infrastructure, yet generally missing in other systems.

Try Yourself!

Look up docs.verus.io to use many API commands (e.g. launching currencies, tokens & liquidity pools).

Or look up the [complete command list here](#).

Join the community. Learn about the protocol. Use Verus & build.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Technology Blockchain Development Cryptocurrency News
Cryptocurrency Blockchain

 121   



Written by Max Theyse

149 Followers · Editor for Verus Coin

[Follow](#)



[More from Max Theyse and Verus Coin](#)

 Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

 196 

 Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

 304 

 Oliver in Verus Coin

 Max Theyse

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346  1 

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259  

[See all from Max Theyse](#)

[See all from Verus Coin](#)

Recommended from Medium

 Yash Agarwal

State of Solana DePIN 2024

Dive deep into Decentralised Physical Infrastructure (DePIN) and why Solana is the...

24 min read · Feb 16, 2024

150  

 PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

143  

Lists

Modern Marketing

103 stories · 523 saves

Generative AI Recommended Reading

52 stories · 894 saves

My Kind Of Medium (All-Time Faves)

71 stories · 261 saves

 Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196  

 Trent McConaghay in Ocean Protocol

The Web3 Sustainability Loop

A system design for long-term growth of Web3 projects, with application to Ocean...

18 min read · Sep 1, 2020

1K  3 

 Sasha Shilina in Paradigm

The future of social networking: Decentralization for user...

 Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

This paper delves into the transformative potential of decentralized social networks a...

35 min read · Nov 6, 2023

74  1 

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4  

4  

[See more recommendations](#)

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)

Ultimate Guide for Launching Currencies on Verus

Launching currencies on Verus, and any other PBaaS-chain (Public Blockchains as a Service), is better, faster, cheaper and more secure than any EVM-like protocol out there. This article explains why, gives an in-depth view of all the options and parameters, and teaches how to launch these currencies. Welcome to Verus, a fundamentally different protocol, and the future of currency launches and value creation.



Max Theyse · Follow

Published in Verus Coin · 16 min read · Jun 16, 2023

532



What are Currencies?

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

involved. It is just creating a transaction, and then sending that transaction to the blockchain making the currency a reality. Anyone can do it.

Better, Faster, Cheaper and More Secure

Why is it smarter to launch currencies on the Verus Protocol? The answer is simple, because all launched currencies are part of the consensus mechanism. On EVM-like and account-based protocols (Ethereum, BNB Chain, Polkadot, Cosmos, Avalanche) currencies are not part of consensus and thus easily exploitable by smart contract bugs and malicious developers. With Verus, a UTXO model, all currencies are accounted for, and verified by the worldwide miners and stakers of the protocol, making it as secure as sending BTC on Bitcoin. To make it simple — all features on the Verus Protocol are on the first layer, the consensus layer.

Verus fundamentally revolutionizes the preexisting cryptocurrency landscape

Also, where other protocols need L2 solutions which are mostly centralized, Verus includes self-sovereign identities, namespaces, privacy, DeFi, bridges, and currency and blockchain launches on the first layer. An innovation that fundamentally revolutionizes the preexisting cryptocurrency landscape.

Launching currencies with Verus is also much faster and cheaper than with the alternatives. The magic here is that there is no coding involved. Anyone can launch a currency by just choosing the options and parameters they would like. It is as easy as typing in a command and the currency is ready to be used by the world within minutes. This process does not involve expensive Solidity developers or security audits, saving a lot of development costs.

Another big advantage is being able to export launched currencies to Ethereum as ERC-20. These currencies can then partake in the Ethereum ecosystem, yet have better security and cheaper DeFi transactions than when they are created on Ethereum itself. Each ERC-20 token can also be bridged to Verus. It's all super easy thanks to the non-custodial Verus-Ethereum Bridge.

This is not all some fantasy, this is all functional (the Verus-Ethereum Bridge is in testing), right now, on mainnet, on the Verus Protocol. As you now know, Verus is built fundamentally differently than any other protocol out there, and ushers in a new era for brands, communities, businesses, entrepreneurs and organizations to create value in a better and more secure way than ever before.

The next chapter explains the options to choose from when launching currencies.

Options to Launch Currencies

For ease of explaining, all PBaaS-chain launch options are omitted. Good to know — with Verus anyone can also launch fully interoperable, independent and customizable blockchains with all the same L1 features as the Verus blockchain. The focus of this article is on launching currencies only, on Verus or any other PBaaS-chain.

To launch a currency on Verus (and any other PBaaS-chain) a namespace is needed. That namespace is a VerusID. The namespace is the name of the currency (e.g. `MyBrand@`). That namespace can also have subIDs (e.g. `product.MyBrand@`, `user.MyBrand@`, more on this later).

On Verus a VerusID registration costs between 20 and 100 VRSC, and the launch of a currency costs 200 VRSC. These costs are always paid in the chain's native coin (to the worldwide miners and stakers). On other PBaaS-chains these costs can differ since the chain launcher can define its own costs for VerusID registrations and currency launches.

To define a currency, choose options. Then combine these options to whatever the currency needs to be. Just add the numbers together in the currency definition. The currency options are listed below.

- `options:1` — The currency has reserves, and can be converted to and from the reserves (option 32 needs to be added). Can have one currency as its reserves, or multiple with up to 10 currencies. Let's call it a "Basket currency" — a currency with a basket of reserves. Such a currency can be launched centralized or decentralized.
- `options:2` — Only the controlling VerusID (the namespace of the currency, the rootID) can create subIDs.
- `options:8` — Referrals and discounts are enabled for subID creation.
- `options:16` — Referrals are required for subID creation.
- `options:32` — The currency is a simple token currency without any reserves. Such a currency can be launched centralized or decentralized. This option is also used for Ethereum ERC-20 mapped tokens.

- `options:2048` — A single satoshi (0.00000001) NFT token is created & has tokenized control of the root VerusID. Which means you can send the single satoshi token to other addresses and then they have control of the root VerusID.

Let's give some examples of combined options:

- `"options":33` — this launches a **basket currency**. It is options 1 + 32 combined.
- `"options":2080` — this launches a **single satoshi token** that has control of the root VerusID. It is options 32 + 2048 combined.
- `"options":35` — this launches a **basket currency**, and **only** the rootID can create subIDs. It is options 1 + 2 + 32 combined.

And so on. More information on basket currencies, simple token currencies and subIDs in the next chapter.

Two Types of Currencies

There are two types of currencies that can be launched with the Verus Protocol. Basket currencies and simple token currencies. Both can be issued decentralized, or centralized (has minting and burning capabilities.)

Basket Currencies

A simple example of a basket currency containing two reserve currencies.

Basket currencies are magical currencies that have reserves. Have a look at the simplified image. If anyone has currency X or Y, they can convert to the basket currency, or convert from reserve to reserve. If anyone has the basket currency, they can go to currency X or Y. A basket currency can have 1 and up to 10 reserve currencies.

The basket currency supply is dynamic, depending on how much is converted to the basket currency (supply minted), or back to its reserve(s) (supply burned).

A basket currency can be 100% backed by its reserves, 5%, or anything in between. This is called the reserve ratio, or the weight. The lower the reserve ratio, the more volatile the currency is when people are converting into or out of the basket currency. The value of the basket currency is directly linked to what is in the reserves and what the reserve ratio is.

When a centralized version of this currency is created, the owner of the rootID can mint currencies into existence, while automatically lowering the reserve ratio. Or they can burn currencies and automatically raise the reserve ratio. Anyone can also just burn the currency at will without raising the reserve ratio.

The conversion fees are incredibly low, 0.025% when converting to and from a basket currency, and 0.05% when converting from reserve to reserve currency. These fees go directly to the worldwide miners and stakers of the protocol, and/or they are accrued into the reserves making the basket currency worth more.

Because all currency conversions are solved simultaneously inside a block, giving all participants the same price, the protocol is MEV-free. The protocol doesn't have any of the problems EVM-like account-based systems have. Verus DeFi is fair, cheap and has no rent-seekers.

Every(!) currency on the Verus network (also mapped ERC-20s!), can be used as reserves for basket currencies. As you might start to understand now – basket currencies are unique currencies that can not be found anywhere else and offer an enormous amount of opportunities for value creation. And anyone can launch them without needing any programming knowledge!

Simple Token Currencies

Simple token currencies are just currencies without any reserves. They are not as exciting as the basket currencies, yet still offer much value. With all the parameters that can be added, subIDs created and decentralized crowdfund mechanism, these currencies can support a lot of use cases that are difficult to do with alternative protocols. [A small example of a centralized currency here.](#)

The supply of this type of currency is static when it's a decentralized version. When it's a centralized version, the owner of the rootID can mint tokens into existence, and anyone can burn them.

This option is also used to create currencies that are mapped to Ethereum ERC-20s. Which means you can send those ERC-20s over to Verus, or from Verus to the ERC-20. This is made possible with the non-custodial Verus-Ethereum Bridge. [You can read more about it here.](#)

And of course, a simple token currency can be one of the reserves in a basket currency.

What are SubIDs

SubIDs are powerful objects on the blockchain. They can be registered in an unlimited amount under any created currency (basket and simple token currency). A few examples are `product.MyBrand@` or `user.MyBrand@`. The name `MyBrand@` is a VerusID (a namespace) that is converted into a currency. The subIDs are 'product' and 'user'.

SubIDs can be bound digitally to many things. They can be bound to people and products. They can be bound to an unlimited amount of content, data, and provable information, both public and private. Including provable contracts and rights that can be bound to ownership of the subID itself.

Another powerful feature is that it has read, write and delete functions in what is called the "content multimap". All kinds of information can be read and processed in the mempool of the blockchain.

It doesn't stop there, it's also a friendly name blockchain address that can receive, send, hold and stake funds, and can be used to login to services, password-free. [Read more about VerusID here.](#)

Parameters to Launch Currencies



This is all anyone needs to do to launch powerful currencies. Zero coding!

Next up are the parameters. Choose the parameters wisely to launch a currency that suits any need. Not all parameters are needed or combinable. There are many to go through, so let's start.

“proofprotocol”

This parameter defines, among others, if the currency is centralized or decentralized. You can choose 1,2 or 3.

1 is default, which launches a decentralized currency (no need to include this parameter when defining such a currency). When subIDs are created with this option, the registration fees are burned.

2 is for a centralized currency. If it's a basket currency the rootID can mint and burn supply while automatically lowering and raising the reserve ratio (also anyone can burn supply without lowering the reserve ratio). Or when it's a simple token currency they can just mint and burn supply. The subID registration fees go to the rootID.

3 is for Ethereum ERC-20 mapped tokens. [Read more here](#)

Example currency:

```
definecurrency'{
  "name": "MyBrand",
  "options": 32,
  "proofprotocol": 2,
  "preallocations": [{"Klaus@": 100}]
}'
```

 A simple token currency called *MyBrand*, centralized (the controller of the rootID can mint and burn), and has a preallocation of 100 tokens to *Klaus@*.

“currencies”

Here you put the names of the currencies (or just one — it must have VRSC when launched on Verus) that will be in the reserves when it's a basket currency (`options:33`).

Or when it's a simple token currency (`options:32`), what people convert during the preconversion timeframe will go to the rootID of the currency, as a funding mechanism. In the case of a simple token currency, combine it with `"conversions"` to determine the preconversion price.

Example currency:

```
definecurrency'{
  "name": "CommunityX",
  "options": 33,
  "currencies": ["vrsctest", "MyBrand"],
  "minpreconversion": [10, 50],
  "initialsupply": 100
}'
```

 A basket currency called *CommunityX*. It needs to get a minimum of 10 *VRSCTEST* and 50 *MyBrand* into its reserves within the preconversion time frame to be launched. The initial supply of 100 *CommunityX* will be distributed among the preconverters.

“conversions”

Use this parameter when launching a simple token currency. Together with `"currencies"`, it can be used as a funding mechanism for the rootID. It's for the preconversion price. So when doing `"conversions": [0.1]`, it means that for every VRSC the preconverter receives 10 CURRENCY after launch. The converted VRSC goes into the rootID.

People can preconvert to this currency within the preconversion time frame. Define a `"startblock"`, or let the default and minimum time frame play out, which is 20 blocks.

Example currency:

```
definecurrency'{
  "name":"CoolBrand",
  "options":32,
  "currencies":["vrsctest"],
  "conversions":[0.1],
  "minpreconversion":1000
}'
```

 This simple token currency is called CoolBrand. During the preconversion time frame people need to convert 1000 VRSCTEST to the rootID. In exchange for that they receive 10.000 CoolBrand. If this minimum amount is not met, the currency will not launch, and everyone who did a preconvert will get their funds back.

“minpreconversion”

Use this parameter to set a minimum amount of preconversions. The minimum amount of preconversions needs to be met or the currency will not launch and everyone gets their conversions returned, minus the transaction and conversion fees. It works both with basket currencies and simple token currencies.

There is a 0.025% fee taken when preconverting. Take this into consideration when trying to meet the minimum amount of preconversions.

Example currency:

```
definecurrency'{
  "name":"CompanyX",
  "options":32,
  "currencies":["vrsctest"],
  "conversions":[2],
  "minpreconversion":500
}'
```

 This simple token currency is called CompanyX. During the preconversion time frame people need to convert 500 VRSCTEST to the rootID. In exchange for that they receive 250 CompanyX. If this minimum amount is not met, the currency will not launch, and everyone who did a preconvert will get their funds back.

“maxpreconversions”

Use this parameter to set a maximum amount of preconversions. During the preconversion time frame the amount set can not be exceeded. Everything above this amount will be automatically refunded after the currency is launched.

Example currency:

```
definecurrency'{
  "name":"CoinCommunity",
  "options":33,
  "currencies":["vrsctest"],
  "maxpreconversion":100,
  "initialsupply":100
}'
```

 This is a basket currency called CoinCommunity. During the preconversion time frame people can convert VRSCTEST into its reserves for 100 CoinCommunity in return. During the preconversion time frame there can not be more than 100 VRSCTEST converted into its reserves. Whatever is preconverted more will be returned.

“initialcontributions”

The rootID can contribute some or all of the minimum preconversions directly as part of the currency definition. Use this parameter to make an initial contribution to either the reserves when it's a basket currency (`options:33`), or to the rootID when it's a simple token currency (`options:32`).

The funds to initially contribute need to be in the rootID when defining the currency. After the preconversion time frame is over and the currency launched, the rootID has received an amount of the launched currency.

Example currency:

```
definecurrency'{
  "name":"CommunityBasket",
  "options":33,
  "currencies":["vrsctest","CoinCommunity"],
  "initialcontributions":[10,200],
  "initialsupply":100,
  "preallocations":[{"Jane@":100},{"John@":50}]
}'
```

 This is a basket currency called *CommunityBasket*. The launcher of the currency wanted to make initial contributions to its reserves. At the moment of broadcasting the currency to the network, there needed to be 210 *VRSCTEST* and 200 *CoinCommunity* in the rootID. The initial supply of 100 went to the rootID (if there weren't any more preconverters). At the same time of the launch, 100 *CoinCommunity* was minted into *Jane@* and 50 into *John@*, this lowered the reserve ratio of the currency.

“initialsupply”

A required parameter for basket currencies (`options:33`). Does not work with simple token currencies. This is the initial supply during the preconversion time frame, before the currency is launched. People preconverting into the reserves receive from this initial supply.

Immediately after the currency is launched, the supply can be larger due to `“preallocations”`.

Example currency:

```
definecurrency'{
  "name":"SocialBrand",
  "options":33, "currencies":["vrsctest"],
  "initialsupply":500,
  "preallocations":[{"Max@":1000}]
}'
```

 This is a basket currency called *SocialBrand*. People can preconvert *VRSCTEST* into its reserves and in return they get 500 *SocialBrand* distributed among them. Immediately after launch *Max@* receives 1000 *SocialBrand*, lowering the reserve ratio of the currency.

“preallocations”

Use this parameter to receive a chosen amount of funds after the preconversion time frame has passed and the currency is launched. Funds can be directed to VerusIDs. Works with simple token currencies and basket currencies.

When using this parameter with basket currencies, after the preconversion time frame has passed and the currency is launched, the reserve ratio is lowered. This is because new currency has been minted after the initial supply (`“initialsupply”`) is distributed.

“prelaunchcarveout”

Only works with basket currencies (`options:33`). Use this to redirect a percentage of preconverted reserves to the rootID.

After the preconversion time frame has passed and the currency is launched, a percentage of the reserves is taken and redirected to the rootID. This lowers the reserve ratio, making the currency more volatile.

Example currency:

```
definecurrency'{
  "name":"BusinessBrand",
  "options":33,
  "currencies":["vrsctest"],
  "initialsupply":100,
  "prelaunchcarveout":0.1
}'
```

🔍 This is a basket currency called BusinessBrand. People can preconvert VRSCTEST into its reserves in return for 100 BusinessBrand distributed among them. When the currency is launched, 10% VRSCTEST is taken out of the reserves, into the rootID. This lowers the reserve ratio by 10%.

“prelaunchdiscount”

Only works with basket currencies (`options:33`). Use this to give people a discount during the preconversion time frame. After the preconversion time frame and the currency is launched, the conversion price will be higher, depending on what percentage the discount was.

When using this parameter, after the currency is launched, the reserve ratio will be lowered by the discounted percentage.

Example currency:

```
definecurrency'{
  "name":"DiscountBrand",
  "options":33,
  "currencies":["vrsctest"],
  "initialsupply":100,
  "prelaunchdiscount":0.5
}'
```

🔍 This is a basket currency called DiscountBrand. People can preconvert VRSCTEST into its reserves in return for 100 DiscountBrand distributed among them. Immediately after the launch of the currency, when people want to convert, the price is 50% higher. Also, the reserve ratio is 50% lower because of the prelaunchdiscount.

“weights”

Only works with basket currencies (`options:33`). Use this to change the respective weights of the reserves in a basket currency. The total of all weights must equal 1. With a minimum of 0.1, since there can't be more than 10 reserve currencies in a basket currency.

Example currency:

```
definecurrency'{
  "name":"MyBusiness",
  "options":33,
  "currencies":["vrsctest","BusinessBrand","DiscountBrand"],
  "initialsupply":100,
  "weights":[0.5,0.25,0.25]
}'
```

🔍 This is a basket currency called MyBusiness. During the preconversion time frame there are various currencies that can be converted into its reserves. They have different weights to them. 0.5 for VRSCTEST, 0.25 for both BusinessBrand and DiscountBrand.

“startblock”

Use this parameter to define the block height when the currency is launched. There is a preconversion time frame before the currency is launched. When omitting this parameter it uses a 20 block preconversion time frame before the currency is launched.

The preconversion time frame is always 20 blocks, this can not be less.

“endblock”

Endblock can not be defined on basket currencies. It does nothing. It could be set as a signal to software that might use the basket currency.

It can be set on centralized (`proofprotocol:2`) simple token currencies. When the endblock is reached, it turns the centralized currency into a decentralized one.

“idregistrationfees”

Use this parameter to change the costs of registering subIDs under the rootID. The default registration fee is 100.

When it's a decentralized currency the fees are burned, when it's centralized the fees go to the rootID.

To enable discounts when using referrals, add `“options”:8` to the currency definition.

“idreferrallevels”

Use this parameter to change the levels of referrals used when registering subIDs. The default is 3 levels.

“nativecurrencyid”

Use this parameter for mapped ERC-20 tokens. The parameter includes the Ethereum contract address for the ERC-20. [Read here for more information.](#)

... . . .

These were all the parameters to use when defining currencies. There are many, and combining them in the right ways make powerful currencies and tokens. The community is always ready to help when defining and launching currencies for your use cases! [Please go to Discord and ask away.](#)

The next chapter explains how to exactly launch these currencies.

How to Launch the Currencies

Launching these currencies is quite easy. Launch with the command-line interface, or Verus Desktop for Windows, Linux or macOS. [Download here.](#)

In the chapter before many examples of currency definitions are given. Let's use one and see how it is launched.

First of all let's see the difference between launching with Verus Desktop and the command-line interface. Verus testnet is used for testing purposes.

⚠ It is highly recommended to launch currencies on testnet first before going to mainnet! ⚠

Verus Desktop: go to “Settings” (cogwheel top-right) -> “Coin Settings”. There you see an input field.

Here you can run commands in Verus Desktop

To use the *input field* and launch a currency on Verus Desktop (see *run* before the command):

```
run definecurrency '{
  "name": "MyBrand",
  "options": 32,
  "proofprotocol": 2,
  "preallocations": [{"Klaus@": 100}]
}'
```

Command-line interface: (this is for testnet, when on mainnet omit `-chain=VRSCTEST` or for any other PBaaS-chain, just edit `VRSCTEST`)

```
./verus -chain=VRSCTEST definecurrency '{
  "name": "MyBrand",
  "options": 32,
  "proofprotocol": 2,
  "preallocations": [{"Klaus@": 100}]
}'
```

Now, after doing these commands in either Verus Desktop or the command-line interface you get a HEX returned. It's a very large array of letters and numbers. We need to use this HEX to broadcast the currency to the blockchain. Only then will the currency really start and the funds taken (distributed to the miners and stakers) for the launch.

Using the HEX

To start the currency use these commands:

Verus Desktop:

```
run sendrawtransaction "HEX"
```

Command-line interface: (this is for testnet, when on mainnet omit `-chain=VRSCTEST` or for any other PBaaS-chain, just edit `VRSCTEST`)

```
./verus -chain=VRSCTEST sendrawtransaction "HEX"
```

After doing these commands the currency has started, the funds from the rootID are taken, and it takes a minimum of 20 blocks to actually launch it (and if all preconversion goals are met).

Currency information

During the preconversion time frame and after the launch you can lookup all kinds of information on the currency.

Verus Desktop:

```
run getcurrency "MyBrand"
```

Command-line interface: (this is for testnet, when on mainnet omit `-chain=VRSCTEST` or for any other PBaaS-chain, just edit `VRSCTEST`)

```
./verus -chain=VRSCTEST getcurrency "MyBrand"
```

Now you know everything to launch currencies!  [The worldwide community is happy to help on Discord!](#)

Start to Build

For all the builders out there, you can start to use the Verus Protocol right now for your Dapps or businesses. Building with Verus is better, faster, cheaper and more secure than anything else out there.

Focus on building the application layer instead of spending giant amounts of time and money on Solidity developers and smart contract audits.

Launch powerful currencies for your brand, business, community and organization **now**.

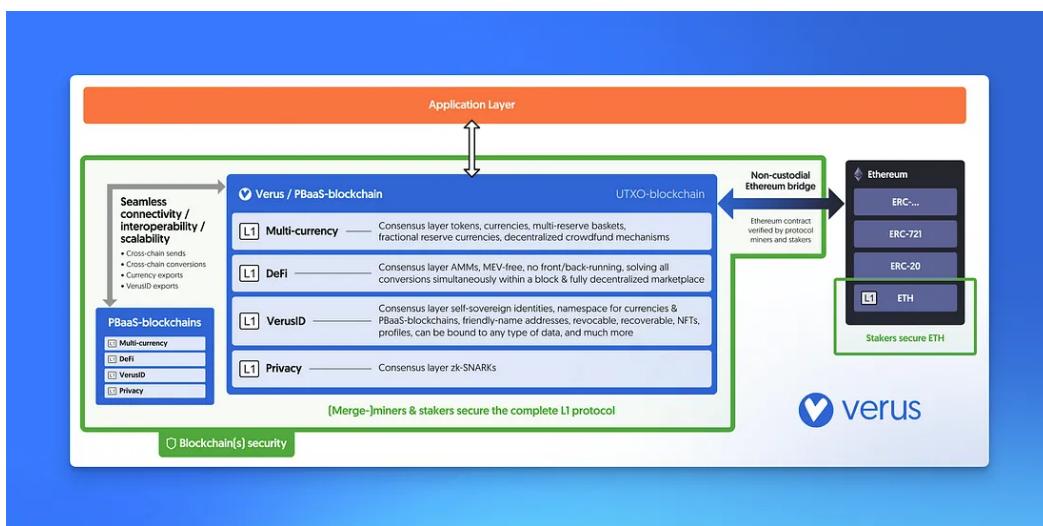


Image showing the L1 primitives of Verus, and how you only need to focus on the application layer when building your Dapp or business!

Join the community. Learn about the protocol. Use Verus. Get ahead of the game.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Special thanks to community member ejuliano for putting in a lot of work that got me started on this! 

Blockchain Technology Blockchain Development Cryptocurrency Investment

Cryptocurrency Blockchain

532



532



Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow



More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024



196



+



304



+

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021



196



+



304



+

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019



346



+



259



+

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024



196



+



259



+

See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024



196



+



538



+

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024



196



+



538



+

Lists

Modern Marketing

103 stories · 523 saves

Generative AI Recommended Reading

52 stories · 894 saves

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4

Q

Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781

Q 4

+

Ahmad Rizwan

12 Lucrative Free Mining Apps To Make Some Serious Money in 2024

Discover the Top Cryptocurrency Mining Apps That Can Generate Profits Without...

5 min read · Mar 19, 2024

180

Q

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2K

Q 29

+

See more recommendations

Introducing Public Blockchains as a Service (PBaaS) — The Revolutionary Layer 0/1 Protocol Launches its Most Anticipated Mainnet Upgrade

After years of hard work and dedication, on the fifth anniversary, the worldwide Verus community is excited to announce the launch of the highly anticipated mainnet upgrade on May 23. The Public Blockchains as a Service release marks a major milestone for Verus and the entire cryptocurrency industry.



Max Theyse · Follow

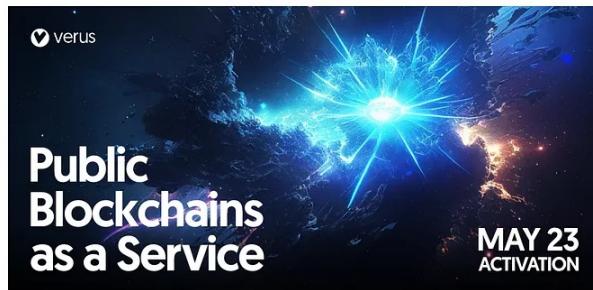
Published in Verus Coin · 8 min read · Apr 25, 2023



304



1



Medium

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

Turning the Crypto Industry on its Head

The wide cryptocurrency industry will see how much better Verus PBaaS is as a solution for true decentralization in cross-chain, DeFi AMMs, on-chain P2P exchange, self-sovereign provable recoverable IDs, NFT capabilities, anti-phishing, anti-MEV, scale, or just about any of the challenges people have on crypto platforms today. In fact, as we prepare for activation of this protocol that enables so many new use cases for crypto, not to mention easy onboarding when entrepreneurs discover how to really leverage VerusIDs, PBaaS is so far beyond what people are experiencing on any crypto platform today that it has been easy to dismiss our community as describing the impossible.

Once PBaaS is live on mainnet and people can actually use all of this themselves, along with every project, chain and currency on the PBaaS network and EVERY ERC20 or ERC721 on Ethereum or even bridged to Ethereum, the truth of Verus will be self-evident, and those who build on Verus will more easily build faster, better, more secure apps and services

with the possibility of provable identity + privacy, crowdfunded projects, businesses, economies, and public infrastructure efforts all seamlessly integrated into UIs that do not need business deals, permission, or any centralized infrastructure to connect services and users, enabling everyone to communicate in provable, private, or public ways that always include bidirectional, secure commerce of all kinds.

Core Capabilities

The core capabilities of this release include (lots to learn to understand it all, but here's a partial list):

Verus Internet Protocol (VIP) — Layer 0 Multichain and Inter-chain Protocol

To our knowledge, the VIP protocol is the only fully decentralized, provable cross-chain technology available on any network that is based on cryptographic proofs of each chain with optional witnesses to confirm chain state. From what we've been able to learn about LayerZero, Cosmos, Polkadot, Thorchain or others, VIP is different from (though closest in some ways to LayerZero), superior to, and more decentralized than all other cross-chain protocols we have seen. We can of course get into any level of discussion on the topic, but expect more descriptions and educational material to follow, now that the protocol was originally conceived of 5 years ago, and has been developed and tested/hardened for years.

VerusID Content Multimaps — Unlimited Provable Data for Every VerusID

Of course this upgrade retains all of the groundbreaking capabilities already available with VerusID, including multisig options, separate revocation and recovery authorities for maximum security to prevent theft of identity or funds, and the ability to timelock your ID, enabling you to even thwart attempts to access your funds after someone compromised your secret keys.

With the PBaaS upgrade, VerusIDs also gain a powerful new capability that amounts to an unlimited, rent-free indexed database for every VerusID on every blockchain of the PBaaS network. This data can be encrypted or published in the clear and used for any purpose or application, whether social networking, voting, oracles, publishing any information, various forms of provable attestations, ratings, and much, much more. With the scalability of PBaaS and VerusID content multimaps together, the Verus PBaaS network becomes the most scalable, permissionless, provable source of human data from which AIs can learn about humanity in the world. We don't believe anything else even comes close, and with the recent rise of AI, it feels great to have this come together as a core original goal at a very appropriate and important time in history.

Permissionless Chain, Token, and Liquidity Basket Currency Launches

Whether you are looking to crowdfund an effort in a manner much like Kickstarter or IndieGoGo, sell identities to which you expect to add value or that are sub-IDs of a very cool root ID, or launch an entirely new blockchain economy and blockchain that starts with all of the Verus technology and a bridge to Verus and Ethereum from day 1, Verus is probably the best platform for you. In fact, as we were planning for the next testnet and wanted to have our own version of DAI to use in the bridge, since we want only decentralized currencies in the live mainnet bridge, we used a command to launch a token on the old testnet, exported it and all its supply to Ethereum's Goerli on the existing bridge, and that will be the DAI proxy/simulant that we will use on the new testnet. It saved us all some time, proving that Verus PBaaS will also be the easiest, most efficient way to launch even an Ethereum ERC20, whether that ERC20 represents a token, DeFi basket, or even another PBaaS blockchain

Verus enables any user with an ID to create their own token currency or even full fledged, multi-currency, ID-issuing 50% POW/50% POS, 51% hash attack resistant blockchain that can send and receive from the Verus chain which launched it.

All PBaaS chains run from the same daemon, and projects may choose to

join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, ID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

Verus also enables any ID owner to define Verus DeFi fractional basket currencies with one or more asset currencies backing the liquidity pool at a fractional percentage ranging from 5% to 100%. The Verus DeFi protocol ensures that all currency conversions that use a particular liquidity pool and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners, stakers, and liquidity providers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi liquidity pools directly into the consensus and cross-chain bridge protocols.

Decentralized Ethereum Bridge

Shortly after PBaaS activates on mainnet, the Ethereum bridge will launch as a decentralized gateway and 1:1 provably mapped currency called “vETH” on the Verus network, also available to all PBaaS chains. The Ethereum bridge will also include a 100% backed basket of 33% Verus, 33% Ethereum, and 33% DAI, which will have the following functions:

- Auto conversion of fees from Verus <-> Ethereum when sending cross chain, based on the on-chain conversion price in the liquidity basket
- Permissionless ability to register *.vETH IDs, which in addition to the normal sub-ID capability of creating a single token that has control over the ID and can be exported to Ethereum as an automatic ERC721 NFT, can alternately be used to create a “mapped currency”, which can be provably mapped 1:1 to any ERC20 currency on the Ethereum network by exporting it to the vETH bridge. Only Verus root IDs, and IDs of a gateway can create mapped currencies to a gateway.
- Decentralized, fair bridge launch. The vETH bridge will launch as other currencies, chains, and gateways can launch, including the bridge converter. When the bridge launches, the contracts will have a surplus of Verus that comes from the fees paid to launch the gateway. In the protocol, that surplus is first used to solve the chicken and egg problem of none of one currency on the other chain. The contracts will cover the Verus fees for all sends from Ethereum before the Bridge converter launches, the liquidity basket becomes active, and the bridge recognizes that launch. Once that happens, the remainder of $\frac{1}{2}$ the fees (5000 VRSC) that are left over from the launch and did not go to miners and stakers or pay Verus fees for people sending from Ethereum.

The Verus Fee Pool and Rewards

Another technology and solution unique to Verus, the Verus Fee Pool technology goes live with PBaaS. What this means is that as people use the cross-chain, DeFi, or purchase/register IDs on the network, the miner or staker will put all fees into the on-chain “fee pool”, and then take 1/100th of the pool in addition to the block reward. This gives everyone incentive to still prioritize transactions based on fees, while preventing validators from gaming the fee system by washing fees-in vs. fees-out or working to reorg/rewind to capture a particularly juicy block reward. Instead, this technology aligns all validator incentives with the health/proper operation of the network and creates a circular on-chain economy that can last well beyond block rewards.

To know more about Verus PBaaS capabilities [read this article](#) 

A Historic Move Forward

For those who actually understand the challenges actual users have in crypto

today, this release represents a historic move forward in public cryptographic networks and credibly and actually neutral infrastructure for society-wide human and AI collaboration and learning. As a community, this version really does realize the vision laid out in the original [Verus Vision Paper](#), and at the same time provides much more than was described there. We are a community of individuals, and it is only because that is what we are that we have all arrived here together. The rest of the world is stuck on their Munchausen's protocols, trying to figure out how to share the front-running and back-running spoils taken from users, but Verus activation gives them a better path forward.

As a community, we do not promise, we describe a shared vision and welcome contributors from everywhere. Now that we, as a community, have delivered on the original phases described in the first vision paper together, it is time for each of us to also consider what we might personally want to build or be part of building over this incredible, geoscale, ID-enabled, fully decentralized platform. Our next efforts across the community should be to realize the promise of this network in the use cases we create, use cases that we will have an advantage creating on Verus over any other platform because it really is that much better.

Read the release notes on GitHub for [CLI](#) and [GUI](#)

Thanks to the incredible combined efforts of so many people in the community ranging from development, companies and projects joining and their open contributions ([Valu/Arkeytyp](#), [CHIPS](#), [vDEX](#), [VaultAlert](#), [craigslist](#), [Electricity Supply Board of Ireland](#) and more) to community members supporting users to helping educate others who can contribute as well, we have run these protocols now for years as we've improved them, and all of these capabilities will go live on the Verus mainnet May 21, 2023, with the activation expected targeting block 2,546,600.

Join the community. Learn about the protocol. Get ahead of the game.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Technology Cryptocurrency News Cryptocurrency
Blockchain Development Blockchain

 304  1

Written by **Max Theyse**

149 Followers · Editor for Verus Coin

Follow 

More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196  

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304  

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346  

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259  

[See all from Max Theyse](#)

[See all from Verus Coin](#)

Recommended from Medium

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

143  

Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Deskt...

5 min read · Oct 12, 2023

146  

Lists

Modern Marketing

103 stories · 523 saves

My Kind Of Medium (All-Time Faves)

71 stories · 261 saves

Generative AI Recommended Reading

52 stories · 894 saves

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4 0

Yash Agarwal

State of Solana DePIN 2024

Dive deep into Decentralised Physical Infrastructure (DePIN) and why Solana is the...

24 min read · Feb 16, 2024

150 0

+

Ilya Ermilov

HOT (Near Protocol) Mining and some hints

Hello friends!

8 min read · Feb 19, 2024

41 0

Ordify

The Tier System

The Ordify Launchpad has only 6 Tiers and operates on a pool weight-based method....

4 min read · Dec 30, 2023

362 3

+

See more recommendations

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)

Introducing Pure — The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC). Hold Pure to be exposed to both Verus and Bitcoin while accruing fees from conversions. Or use it to convert from tBTC to VRSC and vice versa.



Max Theyse · Follow

Published in Verus Coin · 8 min read · Mar 16, 2024

196 Q Share Upvote Comment


⌚ Pure launch block: 2,975,703 (Sunday 24 March, 2024).

✓ Start preconverting to Pure now for an 8 day period. Read further to learn how.

❓ tBTC is BTC on the Ethereum network. [Threshold.network supplies a](#)

Medium

Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

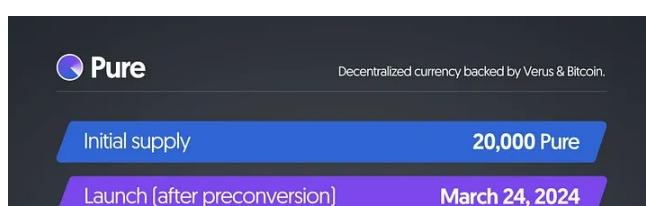
- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)

block producers. No single entity or company controls them. They are not a business — they are rent-free protocols.

- Bitcoin and Verus had fair launches. No ICO, no premine and no developer fees or taxes.
- Bitcoin and Verus have a limited supply.
- Bitcoin and Verus are censorship-resistant.

Hold Pure to be exposed to two decentralized cryptocurrencies upholding true cypherpunk values.





Another benefit of holding Pure is that the currency accrues conversion fees. When users convert from Pure to VRSC or tBTC (or vice versa) they pay a conversion fee of 0.025%. When users convert between VRSC and tBTC they pay a conversion fee of 0.05%.

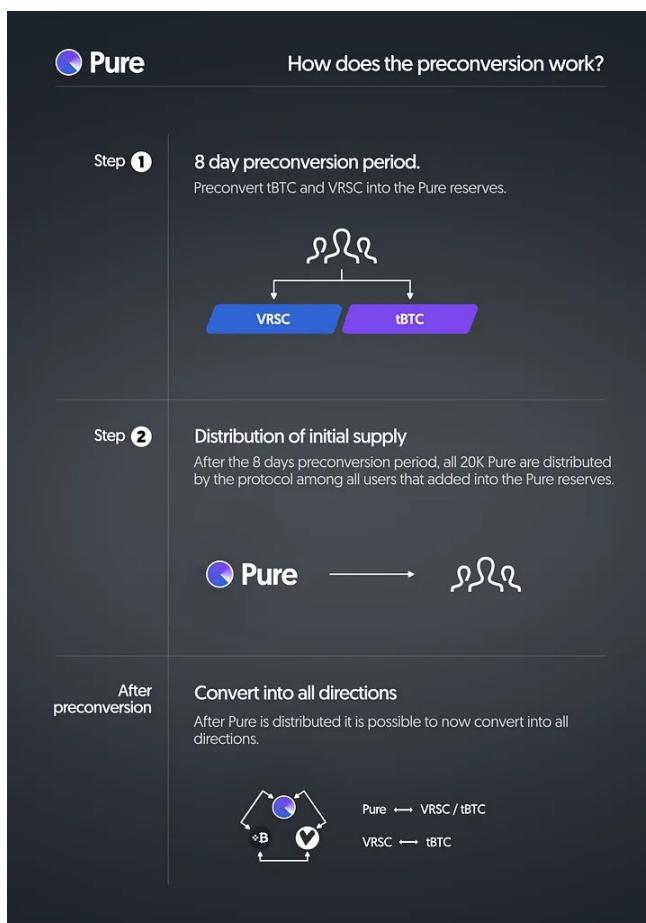
Half of those conversion fees (50%) stay in the Pure reserves, increasing the value of Pure since no new Pure is minted. The conversions are MEV-resistant and verified by the miners and stakers of the protocol. [Learn more about Verus DeFi](#).

Conversions between VRSC and tBTC are only possible after the preconversion period has ended on Sunday 24 March, 2024. From now on you can only add VRSC and tBTC into the reserves of Pure.

Users can also create subIDs which cost 0.00021 tBTC.vETH (in Pure). These costs are burned reducing Pure in circulation. [Learn more about VerusID and what you can do with it here](#). If users create an identity with your subID as referral, they pay 2/3 less – you get 1/3 and 1/3 is discounted.

What is the preconversion period?

The preconversion period is an 8 day period in which users can fund the reserves of Pure with VRSC and tBTC. Depending on how much they added they will get their fair share of the 20,000 Pure after the 8 day period.



The initial supply of Pure is 20,000. After the initial supply has been distributed by the protocol, the supply is dynamic. When users convert from VRSC or tBTC into Pure, Pure is minted. When people convert from Pure back to VRSC or tBTC, the Pure is burned.

What is tBTC, how do I get it & bridge it to Verus?

tBTC is BTC on the Ethereum network. The decentralized bridge between Bitcoin and Ethereum is supplied by the [Threshold Network](#).

Existing solutions that bridge Bitcoin to Ethereum require users to send their Bitcoin to an intermediary, in exchange for an ERC-20 token that represents the original asset. This centralized model requires you to trust a third party and is susceptible to censorship, threatening the premise of Bitcoin as sovereign, secure, permissionless digital asset.

The second generation of tBTC is a truly decentralized (and scalable) bridge between Bitcoin and Ethereum. It provides Bitcoin holders secure and open access to the broader DeFi ecosystem. tBTC v2 allows you to unlock your Bitcoin's value to borrow and lend, mint stablecoins, provide liquidity, and much more.

Get tBTC on Kraken, Uniswap, Curve & others. [Or mint it yourself](#)

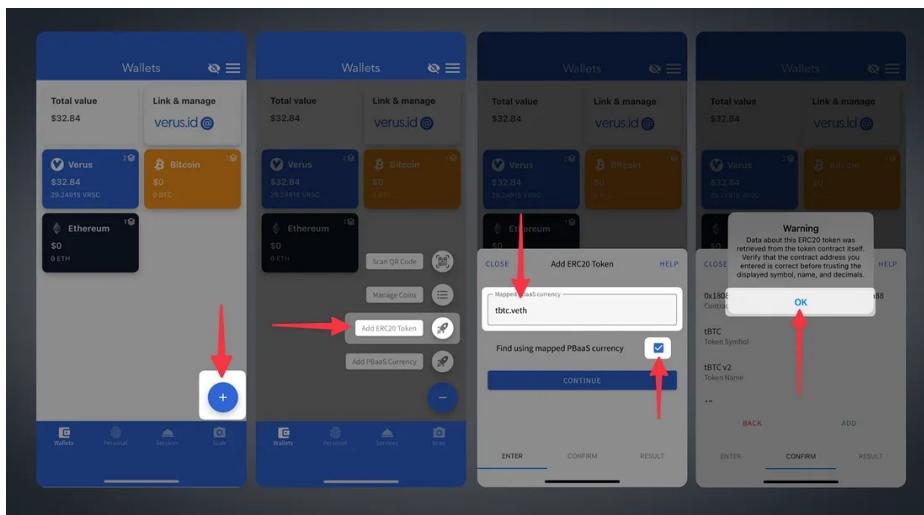
tBTC token address:

<https://etherscan.io/token/0x18084fba666a33d37592fa2633fd49a74dd93a88>

You can send tBTC to your MetaMask wallet or Verus Mobile first, and later you have to bridge it to the Verus blockchain. Bridging is done through the [non-custodial Verus-Ethereum Bridge](#)

Adding tBTC ERC-20 to Verus Mobile

Here is a guide to add the tBTC ERC-20 to Verus Mobile:



How to add tBTC ERC-20 to Verus Mobile

- Press the plus-button bottom-right corner
- Press “Add ERC-20 Token”
- Select checkmark to “Find using mapped PBaaS currency”, and fill in “tbtc.veth”, then press Continue
- Press OK when a warning screen pops up.

You have added the tBTC ERC-20 in Verus Mobile and are now ready to receive it.

Bridging tBTC to the Verus blockchain

You have to bridge the ERC-20 tBTC over to the Verus blockchain with the bridge website or with Verus Mobile to participate in the preconversion period.

⚠️ It costs a considerable amount to bridge over to Verus. The Ethereum blockchain is quite congested and transactions are expensive because of it. Bridging costs estimation: when ETH is @ \$3900 and the gas is at 60 gwei, you pay around \$150 to bridge tBTC over to Verus. It can take up to 60 minutes before the tBTC arrives on the Verus blockchain.

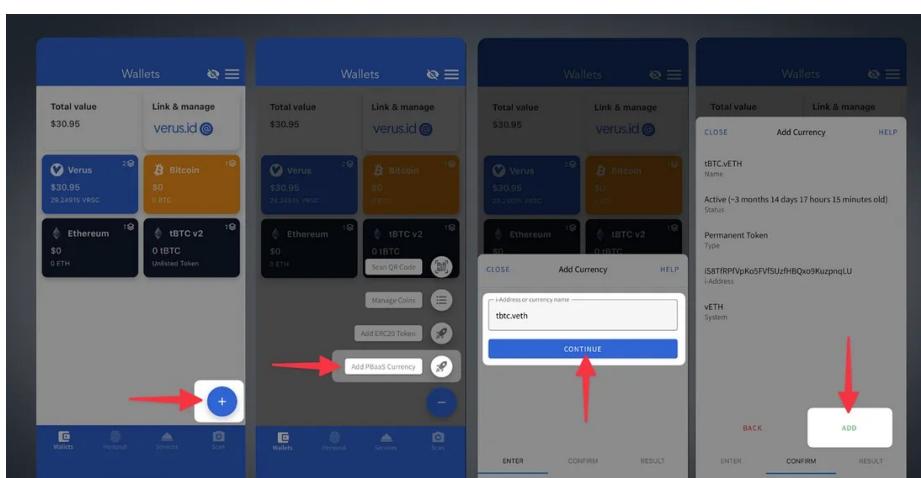
Let's start how to bridge with MetaMask and the [Verus-Ethereum Bridge website](https://eth.verusbridge.io/):

How to bridge tBTC ERC-20 over to Verus with MetaMask and eth.verusbridge.io

- Connect MetaMask with this website: <https://eth.verusbridge.io/>
- Address: where you want the tBTC to end up on the Verus blockchain
- Token: select “[tBTC v2] as tBTC.vETH”
- Destination: select “[tBTC v2] as tBTC.vETH on VRSC”
- Choose the amount you want to bridge over
- If you use Verus Desktop don't forget to add tBTC.vETH in the “Multiverse”-tab!

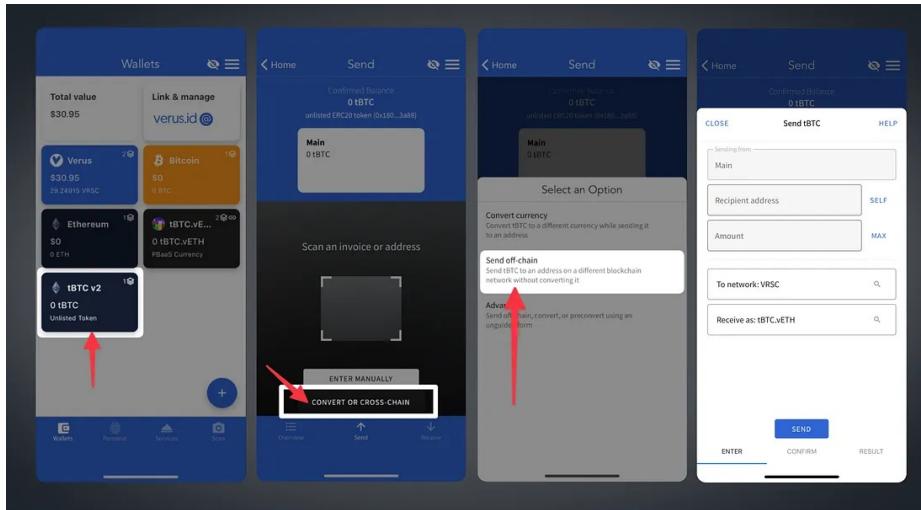
You can also bridge with Verus Mobile.

First let's add tBTC.vETH to your Verus Mobile wallet:



- Press the plus-button bottom-right corner
- Press “Add PBaaS currency”
- Enter “tbtc.veth” and press continue
- Press “ADD”

Now let's bridge the tBTC ERC-20 over to the Verus blockchain.



How to bridge tBTC ERC-20 over to the Verus blockchain as tBTC.vETH

- Press tBTC v2
- Press “Send” and then “Convert or cross-chain”
- Press “Send off-chain”
- As the recipient address fill in a Verus address you own (R- or i-address, or VerusID)
- Fill in the amount you want to send
- Then as “Select network to send to” choose Verus
- Then as “Select currency to receive as” choose tBTC.vETH

Now you can safely bridge the tBTC over.

So now that you've bridged tBTC over to Verus either on Verus Mobile or Verus Desktop, it's time to preconvert.

How do I preconvert to Pure?

When you preconvert you put tBTC.vETH or VRSC in the reserves of Pure. After the 8 day preconversion period you receive an amount of the 20,000 Pure, distributed by the protocol.

You can preconvert with Verus Desktop and Verus Mobile.

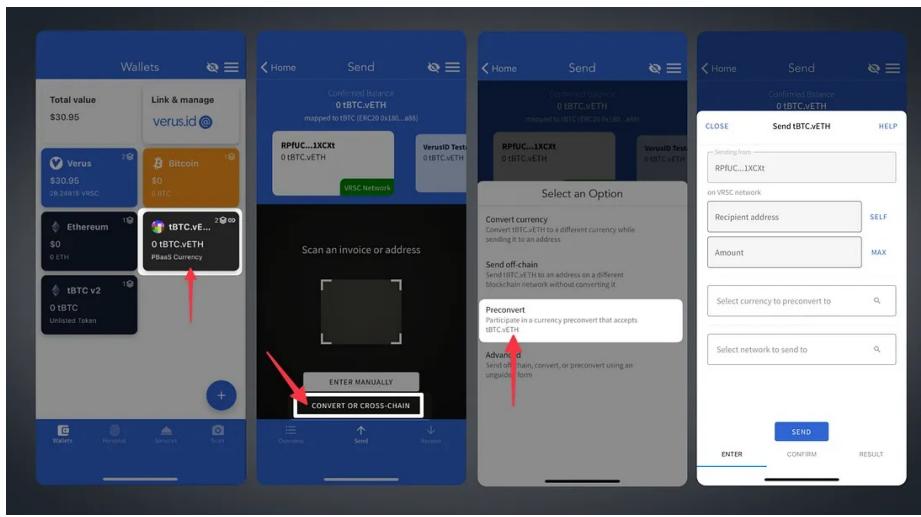
Preconvert with Verus Desktop

On Verus Desktop click the “Convert Currencies”-button and then go to the “Advanced”-tab.

How to preconvert tBTC.vETH or VRSC to Pure with Verus Desktop

- Choose amount to send (tBTC.vETH or VRSC)
- “From Currency”: fill in tBTC.vETH or VRSC
- “To Currency”: fill in Pure
- “Destination Address”: fill in a R-/i-address or VerusID where you want to receive Pure
- Check the box: “Send as pre-convert”
- “Refund Address”: Sometimes a currency has a minimum or maximum amount to preconvert before it launches. For example, when the minimum preconversion amount is not met, and thus the currency not launched, you will receive your funds back. In this case there isn't a minimum or maximum but you still need to fill it in. Can be the same as your Destination Address.
- Now click “Convert Currencies”, and that's it! You have now added a currency into the reserves of Pure and have to wait until Pure is out of the preconversion period.

Preconvert with Verus Mobile



How to preconvert tBTC.vETH to Pure with Verus Mobile

- Press tBTC.vETH or VRSC
- Press “Send” and then “Convert or cross-chain”
- Press “Preconvert”
- As the recipient address fill in a Verus address you own (R- or i-address, or VerusID)
- Fill in the amount you want to preconvert
- Then as “Select currency to preconvert to” choose Pure
- The network to send to is Verus
- Now press “Send” and confirm, and that's it! You have now added a currency into the reserves of Pure and have to wait until Pure is out of the preconversion period.

Now that you have preconverted tBTC or VRSC into the reserves of Pure, all you have to do is wait for the preconversion period to be over. When that

time has come you will receive Pure in your wallet.

Once you have received Pure into your wallet you can always convert it back to VRSC or tBTC.

Join the crypto revolution with Verus & Pure

Pure is a unique currency backed fully by VRSC and tBTC, launched on the Verus Protocol. [Join the Verus Discord](#) and explore what Verus has to offer!

- [Verus solves scalability by neither sacrificing decentralization or security](#)
- [The Verus-Ethereum Bridge is non-custodial & trustless](#)
- [Launch currencies \(like Pure!\) without any coding needed](#)
- [Launch fully interoperable, independent and customizable PBaaS-blockchains](#)
- [VerusID: self-sovereign identities](#)
- [Verus DeFi: simple, low-cost, MEV-resistant and without any middleman](#)
- [Verus is the protocol for builders: join the community at Consensus 2024](#)

And so much more.

Join the community. Learn about the protocol. Use Verus. Get ahead of the game.

[Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Cryptocurrency Cryptocurrency Investment Blockchain Technology Blockchain
Cryptocurrency News

196   

Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow

+

More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

304   

346   

● Vhater in Verus Coin

Mining VerusCoin On Smartphone

the reborn of satoshi vision mining for all

4 min read · Oct 9, 2020

553 3

● Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259

+

[See all from Max Theyse](#)

[See all from Verus Coin](#)

Recommended from Medium

● Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781 4

● Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538 13

+

Lists

Modern Marketing

103 stories · 523 saves

Generative AI Recommended Reading

52 stories · 894 saves

My Kind Of Medium (All-Time Faves)

71 stories · 261 saves

● Tony Aubé

Top 10 Ways You Will Lose All Your Money in Crypto

Crypto is popping these days. Everyone is looking for the next bitcoin to make million...

12 min read · Mar 8, 2024

492 7

● Ahmad Rizwan

12 Lucrative Free Mining Apps To Make Some Serious Money in 2024

Discover the Top Cryptocurrency Mining Apps That Can Generate Profits Without...

5 min read · Mar 19, 2024

180

+

 N. R. Crowningshield  in Kaspa Currency

Kaspa: Accelerating Beyond the Blockchain

From a Concept to a Cutting-Edge Deployment

6 min read · Mar 18, 2024

 107 

 Qsilver

Qubic mining, consensus and computers

Qubic mining seems to be misunderstood by a lot of people, understandably so as it...

5 min read · Jan 14, 2024

 36 

 36 

[See more recommendations](#)

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)



Question: I'm mining since XYZ with XYZ, why I haven't found a block yet?

\$\$ Average Time To Find One Block = (\frac{NetworkHashrate}{LocalHashrate}) * BlockTime \$\$

One block = 6 coins (as now)

NetworkHashrate = retrieved by `getmininginfo` command from </>CLI

LocalHashrate = retrieved by `getmininginfo` command

BlockTime = 60 seconds (average)

note: The above formula assumes you are already mining and your hashrate is already included in the `NetworkHashrate`. For very large LocalHashrate calculations (what-if-I-had scenario) add it to the NetworkHashrate yourself.

Real example with - 31 threads AMD Ryzen 5950x @ 4.4Ghz -

\$\$ Average Time To Find One Block = (\frac{851125882237}{46159950}) * 60 \$\$

\$\$ Average Time To Find One Block = 1,106,317 seconds (307 hours or little under 13 days) \$\$

Bear in mind that these are average times to find a block. In real life you may hit a block much sooner or later after finding the last. In the long run it averages out to the values predicted.

(submitted by @TexWiller, edited by Oink.vrsc@)

note: last revision date 2023-06-21



Is mining profitable?

Necessary files:

Link 1: [Verush Hashrates](#)

Answer:

[↑]

Mining VRSC with Verushash is at this time one of the most profitable coins to mine with your CPU. If you want to know how profitable it is, you need to know a few important details about your own conditions.

1. What hardware will I use to mine with?

1. CPU - If that is a fairly modern CPU with AES and AVX instructions built in.

This will be true if your processor is produced in 2013 or later. For processors between 2008 and 2012 you need to check the specifications.

Older processors can still mine, but they will not perform well.

2. GPU - It is possible to mine with fairly modern Nvidia GPU's, but since Verushash uses specific functions from the

mentioned AES and AVX instruction sets, they will perform better than old CPUs, but worse than modern CPUs.

3. FPGA - Many of these semi-specialized machines can be reprogrammed to mine VRSC. They don't have the same power per processor as a CPU, but they often have multiple processors running parallel. The stronger ones can outperform a CPU easily, but need a lot of power to do so.

1. What is my energy price?

2. Do I use existing hardware or do I buy?

If you know the answer to the hardware, you can look up a comparable one in the Spreadsheet from Link 1, to give you an idea of the performance.

If you know your energy price also, you can [calculate](#) an estimation of how much your hardware can earn you at this moment.

If you have to get new hardware just to mine, think about it for a moment: would you have bought new hardware anyway? If so, you can use this information to get an idea what kind of hardware you want to buy.

Created by Oink.vrsc@



Question: What's the value of VRSC?

Use with: <https://veruspay.io/api/> for simple USD VRSC price, or choose options now added!

Options (values are case insensitive):

currency - BTC or Fiat code like USD or CAD

ticker - ARRR or VRSC

data - volume or price - volume only relevant if exchange is defined

exch - name of supported exchange, e.g. digitalprice - If no exchange, price is average of all supported for that coin.

If no options are set, the default is average price in USD fiat of VRSC.

Examples:

<https://veruspay.io/api/> - This gets the current price of VRSC in USD, weighted against 24hr volume across all exchanges. This is the default return.

<https://veruspay.io/api/?exch=digitalprice¤cy=cad> - This will get the current price on digital price for VRSC and display in CAD fiat

<https://veruspay.io/api/?currency=btc> - This will get the average price of VRSC in BTC, weighted by 24 hr volume across both exchanges

<https://veruspay.io/api/?currency=cad> - This gets the current average price of VRSC in CAD, weighted by 24 hr volume across both exchanges

<https://veruspay.io/api/?exch=cryptobridge&data=volume> - This will get the 24 volume of VRSC on CryptoBridge in the default currency of USD

<https://veruspay.io/api/?exch=cryptobridge&data=volume¤cy=btc> - This does the same but with BTC as the currency result

<https://veruspay.io/api/?currency=cad&ticker=arrr> - Gets the average price of ARRR.

(submitted by @Godballz, API created by @J Oliver Westbrook)



Question 1: I have x amount of mature coins in my public address. How many blocks can I expect to get on average from PoS?

Question 2: When is next halving, what is the block reward, and how long will it last?

Question 3: How many timelocked coins have been unlocked and are mature?

Answer:

<https://github.com/Oink70/Staking-calculator/releases>

Download this spreadsheet and keep it handy to input current blockheight and amount of coins you are staking when you want to know.

You can also enter a percentage for guessing how many coins are actually staking, to see when it matches your own actual average.

See your own percentage of total coins possible to stake, and pay attention to the number day to day, to see how your position holds up.

The spreadsheet will also show how many days until the next halvings.

The calculation of timelocked coins being unlocked is based on an average for the entire period, but should be fairly accurate, and is included in the total of staking coins.

note: Verus Desktop uses the live info from the blockchain with regards to locked coins, coins in z-addresses, current coinsupply, minimum stake age of UTXOs and staking difficulty to display real-time predictions of your profitability on your staking balance.

(submitted by Cragorn.vrsc@, additions by Oink.vrsc@)

note: last revision date 2020-04-30.



Question: What reward do I get for staking (PoS) or Mining (PoW) a block?

The reward received depends of the blocknummer:

Era 1:

1st week: Block 0 - 10080 ==> 0 to 384 VRSC reward ==> 16,588,800 VRSC total this period (reward rising linearly and changing each block)

Era 2:

1st month: Block 10080 - 53279 ==> 384 VRSC reward ==> 8,294,400 VRSC total this period

2nd month: Block 53280 - 96479 ==> 192 VRSC reward ==> 4,147,200 VRSC total this period

3rd month: Block 96480 - 139679 ==> 96 VRSC reward ==> 4,147,200 VRSC total this period

4th month: Block 139680 - 182879 ==> 48 VRSC reward ==> 2,073,600 VRSC total this period

5th month: Block 182880 - 226079 ==> 24 VRSC reward ==> 1,036,800 VRSC total this period

Era 3:

Years 1+2: Block 226080 - 1277279 ==> 24 VRSC reward ==> 25,228,800 VRSC total this period

Years 3+4: Block 1277280 - 2328479 ==> 12 VRSC reward ==> 12,614,400 VRSC total this period

Years 5+6: Block 2328480 - 3379679 ==> 6 VRSC reward ==> 6,307,200 VRSC total this period

. . . halving indefinitely every 1051200 blocks (approximately 2 years)

All rewards equal or over 192 VRSC are time locked to mature at a random block between 129,600 and 1,181,520

(submitted by @keda666, edited by Oink.vrsc@)

note: last review date 2020-02-25.

Question: How to consolidate multiple `wallet.dat` files in one?

Attention: Read it completely before using.

Verus `Wallet.dat`, Chaindata & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

Procedure

[↑]

1. With one of the wallets loaded, issue the following command: `z_exportwallet FILENAME` (ie `z_exportwallet export_instance01`, the filename cannot have a `.` in it.)
2. Copy the generated file to the machine that hosts your main wallet. This file will either be in the same directory as the config file and `wallet.dat` file, or in a location specified by `exportdir` in `VRSC.conf`.
3. Issue the following command (on the "main" verus-cli): `z_importwallet /LOCAL_PATH/EXPORTFILENAME` (ie `/home/user/export_instance01`)

note: Older versions of verusd required `expordir` to be set in `VRSC.conf` before exporting a wallet. If you get an error about your export directory not being set, please upgrade immediately.

These commands can be given in:

- CLI wallet: `./verus z_exportwallet FILENAME & ./verus z_importwallet /LOCAL_PATH/FILENAME`
- Verus Desktop in `settings`, `coin settings`: `run z_exportwallet FILENAME & run z_importwallet /LOCAL_PATH/FILENAME`
- Verus Agama in `settings`, `<CLI>`: `z_exportwallet FILENAME & z_importwallet /LOCAL_PATH/FILENAME`

(submitted by @TexWiller, revised by @englal)

note: last revision date 2020-09-30.



Question: What are the mining pools that I can join?

luckpool.net
pool.verus.io
zergpool.com
wattpool.net
vrsc.ciscotech.dk
www.lepool.com.cn
www.zhuaao.com
aod-tech.com
verus.alphatechit.co.uk
[MadCatMining \(currently inactive\)](#)

The statistics of the known public pools can be checked here:

[Mining Pool Stats](#)

Payouts from mining pools do not need to be shielded first. The mining pool has already taken care of that.

Note: last revision date 2020-11-08.



Question: what are the staking pools that I can join?

- [Dudezmobi](#)
- [Ginasis](#)

A staking pool has advantages and disadvantages over solo staking:

Disadvantages:

1. It causes centralization on the network
2. The pool owner is in control of **all** funds in the pool, including yours.
3. You need to trust the pool operator to share the rewards fairly.
4. You need to trust the pool operator to release your funds back to you on request.
5. You need to trust the pool operator to securely run the pool 24/7.

Advantages:

1. You don't need to run your wallet 24/7 in native mode.

Warning: Do your own research before you decide anything!

Note: last revision date 2020-10-14.



Verus `Wallet.dat`, Chaindata & `VRSC.conf` standard locations

- Linux: `~/.komodo/VRSC`
- Mac OS: `~/Library/Application Support/Komodo/VRSC`
- Windows 10: `%AppData%\Roaming\Komodo\VRSC\`
- OS independent through Verus Desktop: Click `help`, `Show Verus data folder (default)`

Note: last revision date 2020-09-30.

Set up Verus Vault in Verus Desktop (easy method)

Guides

Overview

Get a Verus address

Setup verus-cli

Set up Verus Vault in Verus Desktop (flags)

[Set up Verus Vault in Verus Desktop \(easy method\)](#)

Vault with TimeLock

Vault with DelayLock

Divert staking rewards to different wallet

Claim refunds on the Verus-Ethereum Bridge



Lock your funds with Verus Vault

Easy method

Verus Vault is not yet accessible with clickable interfaces. You can still set up Vault in Verus Desktop. Here's how.

What do you need:

- Latest version Verus Desktop [download here](#)
- VerusID (on the Verus mainchain, or when PBaaS is live on any other chain)

With Verus Vault you can lock funds in your VerusID. When your funds are locked in the Vault you can not spend them anymore, they cannot leave the VerusID. You can still always continue to stake and receive coins.

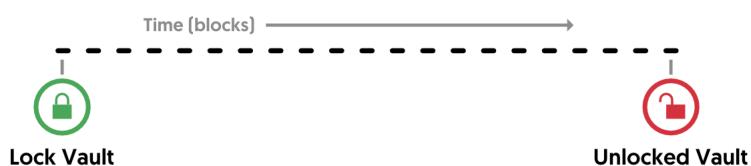
You can lock a VerusID in two different ways that cannot be circumvented by anyone, except the `revocation and recovery authorities` together.

Lock type	How it works
TimeLock	Locks the funds and unlocks until a predetermined number of blocks have passed.
DelayLock	Locks the funds and unlocks with a delay. Funds can not be spent until an unlock has been requested + a predetermined number of blocks have passed.

Get Started

We need to access the commandline interface in Verus Desktop. Go to `settings` (the cogwheel top right corner), then select `Coin Settings`. Here we can fill in the commands to set up your Vault.

Vault with TimeLock



Now let's put a TimeLock on a VerusID. For a TimeLock you need to know the blockheight of the blockchain. Let's say the blockchain's blockheight is at 1,000,000 blocks. You want to lock your VerusID for 1 year. 1 year is 508994 blocks.

Long-Term Locking

For long-term locking it's best to take an average block time of 62 seconds. Yet there are some variables that make it difficult to predict an exact time, leap years for example. Over long-term there are an average of 1394.5484 blocks per day.

- Under `unlockatblock` you put 1508994
- Change `myid@` with your own VerusID

So in our example your VerusID is locked for approximately for 1 year. After that period of time the funds can be spent again.

```
run setidentitytimelock "myid@"
'{
  "unlockatblock":1508994
}'
```

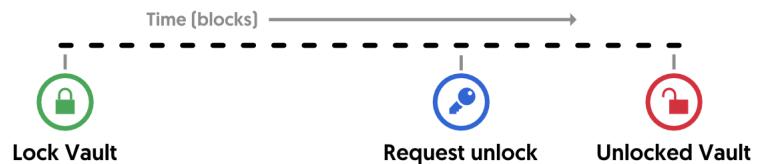
json

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs.

Revoke & Recover

Remember: you can always revoke and recover a locked VerusID.

Vault with DelayLock



Now let's put a DelayLock on a VerusID. This means that you lock the identity, and when you request the identity to be unlocked, a predetermined number of blocks have to pass before you can actually spend the funds again.

Set the DelayLock

Let's say you want to put a DelayLock of 1 week. 1 week is 10,080 blocks (1440x7). This will lock the identity, and when you request an unlock, it takes 1 week (or 10,080 blocks) before the funds can be spent again.

- Under `setunlockdelay` you put 10080
- Change `myid@` with your own VerusID

```
run setidentitytimelock "myid@"
'{
  "setunlockdelay":10080
}'
```

json

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs.

Revoke & Recover

Remember: you can always revoke and recover a locked VerusID.

Request an unlock

Above you locked a VerusID with a DelayLock. Now let's request an unlock. When an unlock has been requested you have to wait the predetermined number of blocks before you can spend the funds again.

To request an unlock you need to know the blockheight of the blockchain. Under `unlockatblock` you can fill in whatever the current blockheight is minus 1. So if the blockheight is at 1,000,000 you can fill in `999999` or just `0` (recommended) to immediately request an unlock.

```
run setidentitytimelock "myid@"
'{
  "unlockatblock":0
}'
```

json

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs.

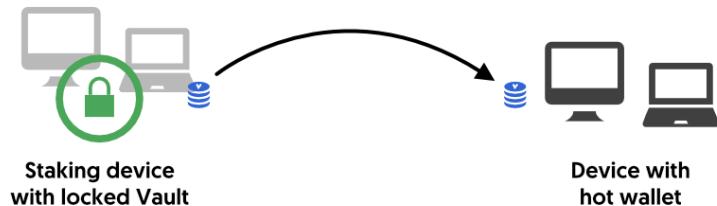
← [Set up Verus Vault in Verus Desktop \(flags\)](#)

[Divert staking rewards to different wallet](#) →

Divert staking rewards to different wallet

Guides

- Overview
- Get a Verus address
- Setup verus-cli
- Set up Verus Vault in Verus Desktop (flags)
- Set up Verus Vault in Verus Desktop (easy method)
- Divert staking rewards to different wallet**
- Claim refunds on the Verus-Ethereum Bridge



If you are staking with funds on a VerusID, and that VerusID is locked with Vault, yet you want to spend your won stakes, this might be for you. Let's explain how you can divert your won stakes to a different wallet with Verus Desktop.

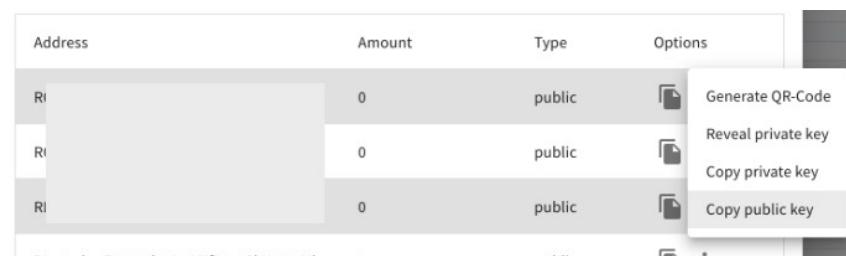
We have two devices with two different wallets:

- Your staking device
- Your hot wallet device

Hot wallet

The hot wallet is the device you want your stakes to arrive on. On the hot wallet we need to lookup the `pubkey` for the address you want your won stakes to arrive.

Go to the `Wallet-tab`, click `receive` under Transparent Balance. Then click the three-dots next to the address you want to use. Click `Copy public key` as seen in the image below. Paste and save this somewhere, we will need it later.



Staking device

This device is staking happily, and your funds are secured with the Vault ([read here how to set up](#)). Let's make sure your won stakes are sent to your hot wallet device.

Go to `settings` (the cogwheel top right corner), then select `Profile Settings` (default). Here you see the option `Custom native mode launch options`. Select `Verus`.

First copy and paste the code from below into the textfield and click `Add launch option`.

```
minetolocalwallet=0
```

Next, copy and paste the code below. Change `xxx` to the `public key` we have from the hot wallet. Click `Add launch option`.

```
pubkey=xxx
```

At last, click the `Save changes` button. **Don't forget this step!** Now close the wallet and open it again. Congratulations, your stakes will now be diverted to your hot wallet.

[← Set up Verus Vault in Verus Desktop \(easy method\)](#)

[Claim refunds on the Verus-Ethereum Bridge →](#)

Claim refunds on the Verus-Ethereum Bridge

Guides

[Overview](#)[Get a Verus address](#)[Setup verus-cli](#)[Set up Verus Vault in Verus Desktop \(flags\)](#)[Set up Verus Vault in Verus Desktop \(easy method\)](#)[Divert staking rewards to different wallet](#)

[Claim refunds on the Verus-Ethereum Bridge](#)

[Ethereum to Ethereum](#)[Verus to Ethereum](#)

Ethereum to Ethereum

When using the Bridge website and making a conversion that goes from Ethereum to Verus and back to Ethereum, you have signed a message that created a refund address based on your Ethereum wallet's private key.

These are the steps to claim refunds when you did an Ethereum to Ethereum transaction:

Verus Mobile

1. Export your Ethereum private key from the wallet you did the conversion with (probably MetaMask)
2. Import that private key into a Verus wallet. The easiest way to do that is with Verus Mobile.
 - Download Verus Mobile on the Play Store or App Store
 - Add a new profile (choose username and password)
 - Choose "import wallet"
 - Choose "Enter key/seed"
 - Enter the Ethereum private key here
 - Add the currency you want to get back (click the plus-button corner-right, "Add PBaaS currency" and choose the one for you)

Verus Desktop

If you are running native mode (full node) Verus Desktop then you can import the Ethereum's private key in this way:

Go to `Settings` (gear-icon top right), then `Coin Settings`, then select the Verus blockchain and type the following:

```
run importprivkey "ETH_PRIVATE_KEY" "" true
```

json

Important!

Verus Desktop will not show any progress on the import. It will take some time for the process to finish in the background.

Verus to Ethereum

When bridging with Verus Desktop from the Verus blockchain to the Ethereum blockchain you had to fill in a refund address.

On the Bridge website you can use that refund address to claim the funds back.

[← Divert staking rewards to different wallet](#)

Guides

Overview

- Get a Verus address
- Setup verus-cli
- Set up Verus Vault in Verus Desktop (flags)
- Set up Verus Vault in Verus Desktop (easy method)
- Divert staking rewards to different wallet
- Claim refunds on the Verus-Ethereum Bridge

Overview

Guides

- [Set up verus-cli for beginners](#)
- [Set up Verus Vault \(flags\)](#)
- [Set up Verus Vault \(easy\)](#)
- [Divert staking rewards to different wallet](#)
- [Claim refunds from Verus-Ethereum Bridge](#)
- [Get Verus address](#)

[Get a Verus address →](#)

Coin Overview

Overview

[Introduction to Verus](#)

Coin Overview

[Verus Proof of Power](#)

Launch Date	21 May 2018
Coin Ticker	VRSC
Average Block Time	1 minute
Transaction Fee	0.0001 VRSC
Max Supply	83,540,184 VRSC
Consensus Algorithm	Verus Proof of Power
Hash Algorithm	VerusHash 2.2
Privacy	Zcash Sapling

← [Introduction to Verus](#)

[Verus Proof of Power](#) →

Verus Proof of Power

VerusPoP is a 50% proof-of-work, 50% proof-of-stake consensus algorithm. [More information on the Verus miner and staker ecosystem.](#)

Overview

- Introduction to Verus
- Coin Overview
- Verus Proof of Power**
 - Hybrid Consensus
 - Attack Resistant
 - VerusHash 2.2
 - PoS Problems Solved

Hybrid Consensus

Verus Proof of Power, or VerusPoP, is a hybrid consensus algorithm which uses a statistical function that combines Proof of Work (PoW) and Proof of Stake (PoS) to validate each block by either PoW or PoS, while averaging to a target percentage of blocks being validated by each form of proof.

In short, it a unique consensus mechanism with 50% of all blocks validated by miners, and the other 50% by stakers.

Attack Resistant

To successfully attack the Verus blockchain, more than 50% of the validation power is needed, called `Chain Power`. A 51% attack would require a combined value of over 50% of both the chain's hashpower and its coin supply. [For technical information on VerusPoP read the whitepaper](#).

VerusPoP provides a decentralizing effect on the network, incentivizing holders to keep nodes online to support the network. Even if a change in network hashrate happens, the PoW/PoS ratio stays the same: 50/50%.

VerusHash 2.2

[From the VerusPoP whitepaper](#):

"VerusHash is specifically developed to deliver a competitive advantage for CPUs with GPUs. It is an exceedingly CPU-friendly long input hash function that uses the quantum-secure, short input Haraka512 V2 as its core compression algorithm. The result is the fastest known cryptocurrency hash algorithm available to modern CPUs and the only hash algorithm which enables today's CPUs and GPUs to compete on an economically comparable level.

Haraka512 V2 is designed as a short input hash to exclusively consume one chunk of 512 bits and produce 256 bits of a hash result. Utilizing Haraka512 V2 VerusHash takes any length of input and produces a 256 bit hash result, unique to VerusHash, that also provides the same security guarantees as Haraka512 V2. This makes VerusHash 256 bit secure for classical computing attacks and 128 bit secure against quantum computers for pre-image and second pre-image attacks.

To understand the VerusHash algorithm it helps to first separate the digest from the core. We then consider the Haraka512 V2 core as an abstract digest function that takes 512 bits (64 bytes) of input and produces 256 bits (32 bytes) of output. Given such a digest function, referred to as `haraka512`, the most concise implementation of VerusHash, in any language to-date, is the following Python code for the VerusHash hash digest as follows:"

```
# verus_hash
def verus_hash(msg):
    buf = [0] * 64
    length = len(msg)
    for i in range(0, length, 32):
```

py

```
    clen = min(32, length - i)
    buf[32:64] = [b for b in msg[i:i + clen]] + [0] * (32 - clen)
    buf[0:32] = haraka512256(buf)
return bytes(buf[0:32])
```

PoS Problems Solved

Verus' staking algorithm solves the two major theoretical issues undermining other PoS systems, [Nothing at Stake](#) and [Weak Subjectivity](#) by leveraging its smart transaction capabilities to remove any incentive to attempt cheating, making it a losing proposition. [Read: How Verus Solved Proof of Stake's Two Biggest Problems: Nothing at Stake and Weak Subjectivity](#) ↗

← [Coin Overview](#)

Introduction to Verus

Overview

Introduction to Verus

Open-source, rent-free, scalable public infrastructure

Low-cost, rapid & easy deployment

Scalability

Security

Interoperability

Decentralized & community-built

Fair launch

Coin Overview

Verus Proof of Power

Welcome to the Verus documentation. Here you find all mainnet and testnet protocol features and implementations. The documentation will be updated continuously.

Open-source, rent-free, scalable public infrastructure

Verus is an open-source, [fairly launched](#), decentralized blockchain protocol with proof-of-work and proof-of-stake as its consensus mechanism. It offers rent-free blockchain tools for creators and organizations to build products, services and systems.

Verus is a multichain protocol with strong focus on scalability, security and decentralization. It can scale to world demand, is proven 51% hash attack resistant and is community built - all coins in circulation are fairly mined and staked.

With Public Blockchains as a Service (PBaaS) anyone can launch scalable, fully interoperable, customizable and independent blockchains for public or private use. Launch tokens & basket currencies (e.g. liquidity pools) on top of blockchains for any use case. Get funding and create markets with protocol level built-in DeFi and smart launch options.

Protocol-level self-sovereign digital identities and namespaces ([VerusID](#)) are at the center of the Verus multichain protocol. Giving builders, communities, businesses and organizations tools never before seen.

Everything happening on the Verus multichain protocol has aligned incentives with the [miners and stakers](#) of the worldwide network. This makes it one of the most secure protocols with opportunities to earn.

Low-cost, rapid & easy deployment

Launch blockchains, tokens, liquidity pools, identities and much more without any coding needed, just simple API commands. Save money on expensive blockchain (Solidity) developers & infrastructure with the rent-free Verus Protocol.

- No programming needed for [blockchain, token and currency launches](#)
- Low protocol fees

Scalability

Verus achieves practically unlimited scalability through its [Public Blockchains as a Service](#). Verus PBaaS enables the provisioning of an unlimited number of

independent, interoperable and interconnected L1 blockchains, which inherit all Verus L1 features.

Verus' scale out TPS (transactions-per-second) is better than what other blockchains have today. A single PBaaS chain can reach between 75 and 800 TPS depending on blocktime; a network of several thousand chains would have a total bandwidth of 1 million+ TPS. But there is no maximum TPS metric because such a thing does not exist on an open fractal network like Verus.

- **Create use case specific blockchains and currencies**
- **Move activity between chains to avoid congestion**

Security

Verus and its PBaaS-blockchains are 51% hash attack resistant. All functionality can be found in the consensus layer (L1), making the protocol secure for developers and users. No smart contract risks — Verus uses smart transactions, not smart contracts.

- **51% hash attack resistant through [Verus Proof of Power](#)**
- Read: [Verus Smart Transactions vs. Smart Contracts](#) ↗

Interoperability

Verus enables a world where all blockchains communicate with each other. All PBaaS-chains are interoperable and interconnected. Verus and all PBaaS-chains are now also connected to Ethereum, in a trustless and non-custodial way.

- Read: [Verus Internet Protocol \(VIP\) — Provable, Decentralized Cross-chain Communication](#) ↗
- [Verus-Ethereum Bridge](#)

Decentralized & community-built

Verus is a decentralized public blockchain, a community driven project in the true spirit of Bitcoin. Anyone can participate and contribute, no matter who you are or where you come from. Verus is:

- **Open**
- **Borderless**
- **Public**
- **Neutral**
- **Censorship resistant**

Anyone can start mining and staking the Verus Protocol. Mobile phones and ARMs (e.g. Orange Pi 5) are the most energy efficient devices to mine Verus (and 22 other PBaaS-chains) with. To stake there are no minimum requirements or locking of funds.

There is no company behind Verus, it is community-built. All protocol fees go to the block producers of the network (the miners and stakers).

These characteristics that anyone can participate in the Verus Protocol makes it

one of the most decentralized networks worldwide.

Fair launch

Verus had a fair launch, meaning that everyone had, and still has equal opportunity to collect its currency through mining and staking. For Verus this means:

- **No ICO has been held**
- **No founder or developer fees/tax**
- **No premine**
- **No commercial interests**
- **No rent-seeking behavior**

The launch of the Verus blockchain was announced on the Bitcointalk.org forum [See post](#) 15 minutes before the first block could be mined.

Rich List

[Take a look at the richlist to see coin distribution.](#)

[Coin Overview →](#)

API commands

All functionality is easily accessible by doing API commands. Here are a few examples:

Verus DeFi

Introduction

API commands

Converting (DeFi)

Sending

Converting (DeFi)

Estimate conversion price

The `estimateconversion` API estimates what you might receive for a certain conversion.

Example:

```
./verus -chain=VRSCTEST estimateconversion '{  
    "currency": "vrsctest",  
    "convertto": "veth",  
    "via": "bridge.veth",  
    "amount": 1000  
}'
```

json

Example:

```
./verus -chain=VRSCTEST estimateconversion '{  
    "currency": "vrsctest",  
    "convertto": "bridge.veth",  
    "amount": 500  
}'
```

json

Get currency converters

The `getcurrencyconverters` API retrieves all currencies that have at least 1000 VRSC in reserve, are greater than 10% VRSC reserve ratio, and have all listed currencies as reserves.

Example `btc` and `eth`:

```
./verus -chain=VRSCTEST getcurrencyconverters btc eth
```

json

Converting VRSCTEST to basket currency

Converting VRSCTEST to a basket currency, VRSC-BTC, using IDs as a funding source:

```
./verus -chain=VRSCTEST sendcurrency "*i" '[{  
    "address": "bob@",  
    "amount": 10,  
}]'
```

json

```
        "convertto": "VRSC-BTC"
    }]'
```

Converting VRSCTEST to BTC via basket currency

Converting VRSCTEST to another reserve, BTC through a basket currency, VRSC-BTC:

```
./verus -chain=VRSCTEST sendcurrency "*" '[{
    "address": "bob@",
    "amount": 10,
    "convertto": "BTC",
    "via": "VRSC-BTC"
}]'
```

Preconverting

Preconverting to new currency, NEWCOIN, before it is active:

```
./verus -chain=VRSCTEST sendcurrency "*" '[{
    "address": "alice@",
    "amount": 10,
    "convertto": "NEWCOIN",
    "preconvert": true,
    "refundto": "alice@"
}]'
```

Converting VRSCTEST cross-chain to PBaaS-chain

```
./verus -chain=VRSCTEST sendcurrency "*" '[{
    "address": "RXLYm4J6qi7yam9zXtkEkRwbvCrnWKGZuv",
    "amount": 10,
    "convertto": "PBaaSChain",
    "exportto": "Bridge.PBaaSChain",
    "via": "Bridge.PBaaSChain"
}]'
```

Converting PBaaS-chain to VRSCTEST

```
./verus -chain=PBaaSChain sendcurrency "*" '[{
    "address": "RXLYm4J6qi7yam9zXtkEkRwbvCrnWKGZuv",
    "amount": 10,
    "convertto": "VRSCTEST",
    "exportto": "VRSCTEST",
    "via": "Bridge.PBaaSChain"
}]'
```

Sending

Sending VRSCTEST from a single address (bob@) to a single recipient (alice@):

```
./verus -chain=VRSCTEST sendcurrency "bob@" '[{  
    "currency":"vrsctest",  
    "address":"alice@",  
    "amount":10  
}]'
```

json

Sending VRSCTEST from all private wallet funds to two recipients with friendly-name z-addresses (alice@:private and bob@:private):

```
./verus -chain=VRSCTEST sendcurrency "*Z" '[{  
    "currency":"vrsctest",  
    "address":"alice@:private",  
    "amount":10  
},  
{  
    "currency": "VRSCTEST",  
    "address": "bob@:private",  
    "amount":10  
}]'
```

json

Sending VRSCTEST cross-chain to PBaaSChain:

```
./verus -chain=VRSCTEST sendcurrency "*" '[{  
    "address": "RXLYm4J6qi7yam9zXtkEkRwbvCnnKGZuv",  
    "amount":10,  
    "exportto": "Bridge.PBaaSChain"  
}]'
```

json

[← Introduction](#)

Introduction

Verus DeFi is incredibly simple, low-cost, MEV-resistant and without any middleman. You can convert into a currency that has reserves (you are now "providing liquidity"), and you can convert out of the currency again, back into its reserves. Furthermore you can convert from reserve to reserve. [Learn more about basket currencies \(DeFi AMMs\)](#).

Verus DeFi

Introduction

L1 DeFi

MEV-resistance

API commands

Verus DeFi	Details
<input checked="" type="checkbox"/> MEV-resistant	Because of protocol design there is no front/back running. Every participant gets the same, fair conversion rate in one or more blocks.
<input checked="" type="checkbox"/> Protocol level security	All DeFi operations take place on the consensus layer of the protocol, and are verified by miners and stakers. There is no smart contract risk.
<input checked="" type="checkbox"/> Low fees	Protocol conversion fees are as low as 0.025%, or as high as 0.05%.

Two conversion types:

Conversion type	Fee	Fee goes to
Basket currency ↔ reserve	0.025%	0.0125% added to reserves of the basket currency, 0.0125% to the block reward for miners and stakers
Reserve ↔ reserve	0.05%	0.025% added to reserves of the basket currency, 0.025% to the block reward for miners and stakers

L1 DeFi

Verus is a `UTXO-based` blockchain with `smart transactions`. All smart capabilities are implemented on the protocol level. This has many advantages over blockchain projects that use layer two solutions. [Read "Smart Transactions vs. Smart Contracts"](#)

Advantages of DeFi at the protocol level:

- Increased security at the application level - Verus DeFi is not implemented by having many smart contract authors creating smart contracts on top of the protocol, so there can be no exploits by searching for unintended "cracks" in the seams between contracts.
- Increased security at the protocol level - Verus DeFi is implemented in the protocol as part of the consensus, following the fundamental systems design principle which says that the most important security layers should be located in the system/protocol itself.

MEV-resistance

The Verus protocol solves all transactions `simultaneously` within a block (as opposed to serially, in order, as is done on Ethereum and all other systems which use the VM-model). This has important implications for security, fairness, and efficiency:

- Elimination of front-running, back-running and sandwich attacks.
- Enhancing system-wide liquidity, thus reducing slippage, as conversions going to and from any given currency within the same block are offset against each other.
- Providing all users converting to and from a currency within the same block the same fair price with no spread.

[API commands →](#)

Verus Data eXchange Format (VDXF)

VDXF

Verus Data eXchange Format (VDXF)

- Overview
- Definition of VDXF types
- Namespace for Type Definitions
 - VerusID
- Implementation

The Verus Data Exchange Format provides a fully interoperable system for defining data types that may consist of structured or unstructured data and associated content or keys that may be used to retrieve such data from centralized or decentralized storage for use in and across centralized or decentralized applications.

The Verus Data eXchange Format (VDXF) object is a structured data representation of the Verus Data eXchange Format, designed to facilitate the exchange of information across different systems and programming languages. It encapsulates data in a serialized byte format, making it interoperable and easy to transmit or store. This document outlines the structure and functionality of a VDXF object, focusing on its serialized form rather than the methods used to manipulate it in any specific programming language.

Overview

The Verus Data Exchange Format enables application developers to define globally unique data types and publish references to the same, which may refer to structured or unstructured data that can be located unambiguously via an URL, which implicitly provides both location and decoding information, enabling applications to use such data, in whole or in part, if they know how, or even ignore parts of the data, while remaining compatible with those parts they understand. VDXF typee keys are globally unique identifiers, which are defined as human readable names along with a specification of how to define and convert unlimited length, human readable type names into collision-free 20 byte IDs, which can be used as type keys associated with content or location values in various forms of data records. These data records, which may have application specific structures or no structure at all, besides length form the basis of an interoperable data exchange format across decentralized applications.

Definition of VDXF types

VDXF is not a strongly opinionated or highly specified type description specification, and, instead, focuses on a model for recognizing an unlimited number of user defined data types, using a standard human readable format for definition and encoding of the type specifier, which is hashed, using the VDXF specification and standard methodology, to produce collision-free, 20 byte keys, which can be associated with retrieveable content hashes and location qualifiers that enable applications to locate, recognize types of, parse, and decode any form of application or system specific data. VDXF specifies some basic type formats, as necessary to enable initial applications, but leaves further specifications of applicaiton specific data formats, of which there may be an unlimited number, as an open-ended option for those needing new data type

definitions for efficient application development. It is recommended that new fundamental data types not be defined unless necessary, but adherence to such recommendation is not enforced at the consensus protocol layer.

Namespace for Type Definitions - VerusID

Namespaces for type definitions are equivalent to VerusIDs, a protocol first implemented on the Verus Blockchain, and also one that can support IDs registered on any blockchain or uniquely named system that becomes recognized via a consensus-based bridge on the Verus network. Currently, to be recognized as a unique namespace, the easiest way is to base it on a VerusID, registered on the Verus blockchain network.

Generally, one may think of two types of VerusIDs, those defined on the Verus network or on independent PBaaS (Public Blockchains as a Service) blockchains spawned originally from and registered on the Verus blockchain network, or VerusIDs, which may also exist on fully external systems that may have been created without any registration on the Verus network initially. In order for an externally created VerusID to be recognizable on the Verus blockchain network or by applications using the VDXF that are compatible with the Verus blockchain network that external system must provide a recognized bridge to the Verus blockchain.

First, it is important to understand the requirements of registered VerusID identity names, which will also inform how externally generated VerusIDs are recognized as well. For the purposes of the VDXF, we do not require compatibility of the internal structure of IDs across different systems, and only define compatibility requirements of the naming systems and how those names translate into recognisable IDs on the Verus network.

Implementation

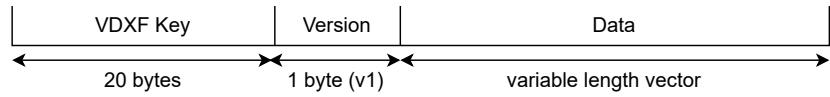
A VDXF object is fundamentally a serialized byte array that can be written to or interpreted by any system, given the appropriate libraries for handling its format. The serialization process transforms structured data into a byte stream, while deserialization reverses this process, reconstructing the original structured data from the byte stream.

Key Components

The VDXF object comprises several key components encoded into bytes:

- **VDXF Key:** A unique, 20 byte identifier for the VDXF object, that can be represented as a human readable string. This key is essential for identifying the type of data the VDXF object represents.
- **Version:** Indicates the version of the VDXF object.
- **Data:** The actual content stored within the VDXF object. This can be any form of data, structured or unstructured, that needs to be serialized.

Serialization Process



The serialization process involves converting the structured data within the VDXF object into a byte stream. This includes encoding the VDXF key, version, and actual data into bytes.

Components Encoding

- **VDXF Key:** 20 byte fixed length slice encoded using Base58Check
- **Version:** Serialized using variable integer encoding to optimize space. The version ensures that the serialized data can be correctly interpreted by systems aware of different versioning.
- **Data:** The actual data is serialized into a byte buffer. The format and encoding of this data can vary widely depending on the type of data being serialized and the intended use case.

Deserialization Process

Deserialization is the reverse of serialization, where the byte stream is converted back into structured data according to the VDXF format specification. This process involves reading the byte stream, extracting and decoding the VDXF key, version, and data components, and reconstructing the original structured data.

VDXF in Action

Learn how the CHIPS project (decentralized poker) uses [VerusID](#) and [VDXF](#)

Register VerusID / subID

VerusID

Introduction

[Register VerusID / subID](#)

VerusID SSID Login

Register a VerusID or subID. First a name needs to be committed, this costs a transaction fee (0.0001), after a block has passed the VerusID or subID can be registered.

VerusID name commitment

Commit a VerusID name by using the following command below. If no referral is available leave it empty.

```
./verus -chain=VRSCTEST registernamecommitment "YOUR_ID_NAME" "YOUR_R_ADDR" --
```

Using the command above gives an output. Take that output and add the highlighted lines from below:

VerusID registration

```
./verus -chain=VRSCTEST registeridentity '{  
  "txid": "2a614ae147a8abcb870eb45d5ddbf1e1d283b942a5e77340d0d268c7fd4726",  
  "namereservation": {  
    "version": 1,  
    "name": "test-id",  
    "parent": "iJhCezBExJHvtyH3fGhNnt2NhU4Ztkf2yq",  
    "salt": "bf5b76bb38cefd2bec266bdcc2f2f37cb321c9aab103e4aa802fdef90224a",  
    "referral": "",  
    "nameid": "iMGMwQhtnaVwdrzev9uMspuNyQbYhCJEmU"  
  },  
  "identity": {  
    "name": "YOUR_ID_NAME",  
    "primaryaddresses": ["R_ADDRESS_CHOSEN_WITH_NAME_COMMITMENT"],  
    "minimumsignatures": 1,  
    "revocationauthority": ["CHOOSE_I_ADDRESS"],  
    "recoveryauthority": ["CHOOSE_I_ADDRESS"]  
  }  
}'
```

SubID name commitment

Commit a subID name by using the following command below (it is almost the same as doing a VerusID name commitment, except currency name is added.)

```
./verus -chain=VRSCTEST registernamecommitment "YOUR_ID_NAME" "YOUR_R_ADDR" --
```

SubID registration

The same as with VerusID registration.

← [Introduction](#)

[VerusID SSID Login](#) →

VerusID SSID Login

VerusID

Introduction

Register VerusID / subID

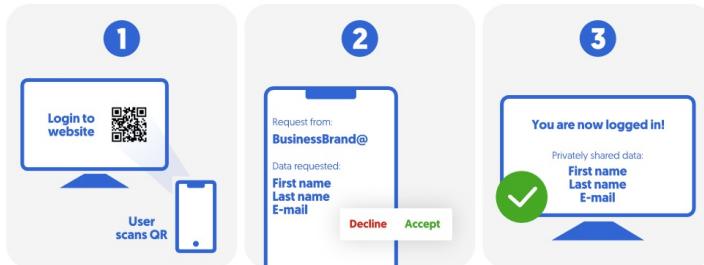
VerusID SSID Login

VerusID Login Demo

Test the VerusID login with Verus Mobile [here](#)

Login to websites and services

Login with VerusID on supported websites and services. Users control how, when, and with whom their personal data is shared. Users control their personal data fully.



These are the steps:

1. **User Scans QR Code:** The user scans a QR code if they want to proceed to login. If the user has a valid attestation the website respects, it can be used on login.
2. **Mobile App Verification and Data Request:** Once Verus Mobile ([Google Play](#), [App Store](#)) scans the QR code the user can verify the request from the site. In this case it will ask the user to reveal some identity data. The user can confirm the source of the request of data by verifying, and then trusting.
3. **Login and Data Sharing:** At the website the user will be logged in using a reply the phone posts to the websites server in the background. This causes an immediate login and sharing of the data privately.

[← Register VerusID / subID](#)

Introduction

VerusID

Introduction

Feature List

Real Estate

Friendly Name

Costs

Structure

Revoke & Recover

Verus Vault

Marketplace

Privacy

Signatures

Multisig

Messages

Register VerusID / subID

VerusID SSID Login

VerusID is the first decentralized and self-sovereign identity of its kind, the permanent namespace for the Verus Protocol, and the building block to create Web3 Dapps.

Feature List

Feature	Details
Namespace	VerusID is the permanent namespace for the Verus Protocol and can be registered by anyone.
Self-sovereign identity	VerusID can function as a self-sovereign identity for anyone in the world, empowering individuals with complete autonomy both online and offline.
Blockchain launches	With the VerusID namespace anyone can launch fully interconnected, customizable, independent and secure blockchains, without any coding needed (just simple commands).
Currency & token launches	With the VerusID namespace anyone can launch powerful currencies and tokens, including basket currencies (DeFi AMMs).
Publish & store data	Use VerusID and VDXF as a controlled public storage system. Publish and store data with multiple levels of nesting.
Revoking & recovering	Each VerusID has revocation and recovery authorities (which are also VerusIDs). Autonomously revoke access to a VerusID, and recover all assets and data on a VerusID.
Friendly name address	A VerusID is a friendly name address that can send, receive and hold assets.
Verus Vault	Enable theft-proof Verus Vault. Set locks or timelocks to secure assets on a VerusID.
Marketplace	Peer-to-peer decentralized marketplace for VerusIDs and currencies and tokens.
Privacy	Point a private address to a VerusID. Send and receive native assets with full anonymity.
Signatures	Create unforgeable, verifiable signatures with VerusID. Sign files, hashes and messages.
Multisig	Multiple organizations or people can manage a VerusID.
Messages	Send and receive completely private messages through VerusID private addresses.
SSID login	Login to supported VerusID services without ever needing a password.

Costs	
Structure	

Feature	Details
SubID	Under each launched currency and token subIDs can be registered. SubIDs have the exact same features as VerusIDs, although they can not launch blockchains, currencies or tokens.

VerusIDs can be anything you want. They can be bound digitally to many things. They can be bound to you, or other VerusIDs they have authority or ownership of. They can be bound to an unlimited amount of content, data, and provable information, both public and private. Including provable contracts and rights that can be bound to ownership of the VerusID itself.

VerusIDs can hold funds. They can be personal profiles, corporate websites, or government portals. VerusIDs are much more than identities or today's NFTs. They are owned assets of all kinds on the Verus blockchain.

A VerusID can be revoked and recovered, set (time)locks and can be controlled by any number of identities. Sign and verify data, files and messages. VerusID has privacy through added z-addresses and can send and receive messages. Identities can communicate in standardized ways through the novel [Verus Data Exchange Format \(VDXF\)](#).

Real Estate

A VerusID is premium real estate on the Verus blockchain. To create PBaaS-chains, tokens and currencies on the Verus blockchain, a VerusID is necessary. The name you assign to the VerusID is also the name of the PBaaS-chain, token or currency.

Each PBaaS-chain has standard VerusID support and creators of these chains can assign the costs of identities themselves. The costs will then be paid in the native coin of the PBaaS-chain.

Friendly Name

Each VerusID can have an easy to remember name, chosen by the user. It has never been easier to send and receive funds.

Supporting Worldwide Adoption

All characters from all character sets (except / : * ? " < > | @ .) are available to create a VerusID.

Costs

On the Verus blockchain a VerusID costs 100 VRSC. These costs can be discounted with referrals. All costs paid are going to the [miners and stakers of the ecosystem](#) and to the referrals if used.

	VerusID Cost
Base cost	100 VRSC
Cost with referral used	80 VRSC

Referrals

A referral system is implemented to reward users with identities. For each VerusID a user creates, the referral identity receives 20 VRSC. The referral system goes three levels down.

If one owns three identities that are stringed together through its referrals, one can receive 60 VRSC with each VerusID creation. This way an identity only has to cost 20 VRSC.

Support the Verus Vision

Use [Verus Coin Foundation@](#) when creating a VerusID on the Verus blockchain. All proceeds will go to the further development of the Verus vision.

Structure

An example of a VerusID:

Structure	Information
VerusID Name	The name is a unique namespace and human readable cryptocurrency address.
Primary Address	The primary address is the owner of the VerusID, as it contains the private key. It can contain more than one VerusID.
Identity Address	The identity address is, next to the name, the identifier of the VerusID.
Private Address	An optional attached private z-address
Revocation Authority	The identity address that can revoke the VerusID
Recovery Authority	The identity address that can recover the VerusID
Contentmap	VDXF key/value data

Input:

```
./verus getidentity "Verus Coin Foundation@"
```

Output:

```
{
```

json

```
        "fullyqualifiedname": "Verus Coin Foundation.VRSC@",
        "identity": {
            "version": 3,
            "flags": 0,
            "primaryaddresses": ["REpxm9bCLMiHRNVPA9unPBWixie7uHFA5C"],
            "minimumsignatures": 1,
            "name": "Verus Coin Foundation",
            "identityaddress": "i5v3h9FWdRFbNHU7DfcGykQjRaHtMqu7",
            "parent": "i5w5MuNik5NtLcYmNzcvaoixooEebB6MGV",
            "systemid": "i5w5MuNik5NtLcYmNzcvaoixooEebB6MGV",
            "contentmap": {

            },
            "contentmultimap": {
                "i5Zkx5Z7tEfh42xtKfwbJ5LgEWE9rEgpFY": [
                    "i5Zkx5Z7tEfh42xtKfwbJ5LgEWE9rEgpFY": {
                        "version": 1,
                        "action": 2,
                        "entrykey": "73960afaad96f923c616b26f9646c059021d4ffa",
                        "valuehash": "10230fb3df7c507f062593c55d94d1442f937b68b71e045c442e1e496470"
                    }
                ],
                {
                    "i5Zkx5Z7tEfh42xtKfwbJ5LgEWE9rEgpFY": {
                        "version": 1,
                        "action": 2,
                        "entrykey": "73960afaad96f923c616b26f9646c059021d4ffa",
                        "valuehash": "9ed2b3516d4cccd2d419bfb12f325902e1a3f566d222445c97005e4e8fee5"
                    }
                ],
                "iSJ38vYX7qoCtotc9wBHb1vZdR3oTgoHCX": [
                    "0186ff9300d99a27d51944ef1563b8c3b"
                ],
                "revocationauthority": "i5v3h9FWdRFbNHU7DfcGykQjRaHtMqu7",
                "recoveryauthority": "i5v3h9FWdRFbNHU7DfcGykQjRaHtMqu7",
                "privateaddress": "zs1dycegwse0x67qvy2fksukcng3ekkgvly2qwjckj8fxraam33xu2y",
                "timelock": 0
            },
            "status": "active",
            "canspendfor": false,
            "cansignfor": false,
            "blockheight": 2588672,
            "txid": "882e3e5e928038bdabae648f0690d919bce85759b3ecc845db458cc1dba0fe83",
            "vout": 0
        }
    }

```

Revoke & Recover

Revoking and recovering identities are essentials in a decentralized system. Users need to have full self-sovereignty to move around in an ecosystem without central control. VerusID is the first decentralized identity system where users have full control over their identities.

When creating a VerusID users can assign a `RevokeID` and a `RecoveryID` to their VerusID. They can be the same identities, different ones or appointed `self`. These assigned identities are also VerusIDs and thus must be purchased.

Action	Outcome
Revoking	Funds can not be spent anymore
Recovering	Recover all assets to a new address. Funds and UTXOs can be spent again

Be Careful

Don't assign a `RecoveryID` to `self` when the `RevokeID` is assigned to another identity. This way when you revoke an identity, you can not recover it anymore.

Verus Vault

The Verus Vault for identities is a unique feature to create extra security. Set locks or timelocks to safeguard funds on a VerusID. Locked identities can not spend funds.

How it Works

There are three stages the Vault can be set to. When the vault is locked it can not spend funds.

Action	Outcome
Locked until x blocks	Funds can not be spent, until a predetermined number of blocks have passed
Locked with delay	Funds can not be spent, until an unlock has been requested + predetermined number of blocks have passed
Unlocked	Funds can be spent

When a VerusID is locked or timelocked, it can still receive, hold and stake funds. It can also still be used for signing.

[Learn here how to set up Verus Vault in Verus Desktop](#)

Revoking Locked VerusID

Even when a VerusID is locked, it can still be **revoked and recovered**.

Vault Use Case Examples

Safe Staking

Put funds to stake with on a VerusID. Use Vault to lock the identity with a delay of 10,080 blocks (~1 week). Now whenever someone gains access to the private keys of the locked VerusID, they have to unlock the identity in order to spend the funds. When someone makes an unlock request that isn't you, you are warned. The intruder has to wait 10,080 blocks

before he can spend the funds. If you have set up revocation and recovery authorities, you now have one week to safeguard your funds away from prying hands, back into your control.

Trusts

Setting up a trust fund for your children. Give your child a time locked VerusID with funds on them. When he or she turns 18, the VerusID unlocks and the funds can be spent. Keep revocation and recovery authorities with yourself or a trustee.

Vesting Periods

When doing a currency or chain launch, development funds can be diverted to a locked VerusID. After two years the VerusID unlocks and the funds can only be spent with signatures of other developers. You can also spread the funds across multiple identities with different unlock periods.

Marketplace

With the VerusID Marketplace protocol, anyone is able to buy and sell VerusIDs. You can look for offers on any VerusID (buy or sell offers). If you like the best offer on your VerusID, or if someone likes an offer you made on theirs, the deal is made 100% peer-to-peer, decentralized on the blockchain, without any middleman or contract controller.

For total payment privacy, you can even pay or receive payment using private addresses and zero knowledge transactions based on the Zcash Sapling protocol.

Exchanging in Private

In addition to advertising worldwide on the blockchain to buy or sell VerusIDs, you can also make an exchange with the Marketplace without ever posting the offer on the blockchain until it is agreed and signed by all parties. Combine that with zero knowledge transactions, and it's a great way to transact worldwide, escrow-free in private.

Use Case Example

A business sells subscriptions for exclusive content. They make VerusIDs with contracts that give access to the exclusive content. They now create a transaction that would pay for the VerusID. They give the transaction to the buyer who then executes it. The buyer now owns the VerusID that gives access to the exclusive content.

Now imagine how you can do this for a VerusID that can be any kind of asset. A new way for everyone to engage in peer-to-peer, escrow-free commerce has arrived.

RPC APIs

API	What it does
makeoffer	define what you offer and for what. What you offer can be funds, VerusIDs, or when PBaaS goes live even other currencies. What you want in return can also be funds, VerusIDs or currencies. In exchange for what you offer, you also define what VerusID or how much you want for it and in what currency
takeoffer	take a specific offer in exchange for its request
getoffers	specify which VerusID or currency you want to see offers for or on offer, and it returns all offers (buy and sell) in all currencies, sorted by highest to lowest price
closeoffers	close expired or unexpired offers which you opened with makeoffer
listopenoffers	list all offers that you have opened with makeoffer

Privacy

A VerusID can contain a pointer to a `z-address`. These are private addresses that can not be checked on the public blockchain. Attach any z-address to a VerusID.

Signatures

Create unforgeable, verifiable signatures with VerusID. Sign files, hashes and messages. Use the protocol to verify those signatures for free.

Multisig

Multiple VerusIDs can have spending or signing ability of one VerusID. This means that multiple organizations or people can manage a VerusID.

Messages

Send private messages to VerusIDs.

[Register VerusID / subID →](#)

[Sign in](#)

monkins1010 / **VerusPay** Public

forked from [OliverCodez/VerusPay](#)

[Code](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)[...](#)

Home

[Jump to bottom](#)

Chris edited this page on Jun 24, 2020 · 7 revisions

VerusPay Installation guide for WooCommerce

⚠ Disclaimer no SSL security and IP locking of the firewall is mentioned in this guide, Please use at your own risk. Add SSL and firewalls as necessary before deploying on a live Shop. ⚠

This guide will show you how to setup the VerusPay plugin so that your WooCommerce store can accept cryptocurrency payments from VRSC, KMD, ARRR, ZCASH & more.

What is covered

- [**Prerequisites**](#) What you'll need
- [**How to install WordPress**](#), which gives you a website for people to visit. With this website you can create blog pages, news pages, idea pages, anything your creative mind desires, but most importantly your WordPress site allows you to run an online shop using a Plug-in called WooCommerce.
- [**How to Install the command line wallet**](#), you'll need an interface to the Verus block-chain to send and receive money to anyone in the whole world. With this unassuming text based program running on your server computer, it allows you to receive payments from customers and send them to your own personal wallet or wherever you want to process them. It also enables you mine and stake your cryptocurrency if you so choose.
- [**How to install Verus ChainTools**](#), having a shop and a wallet enables you to process customers orders and handle cryptocurrency, but we need to connect them together so that the shop website can request payment information from your wallet. This is where ChainTools comes in, it keeps your cryptocurrency wallet separate from your website traffic for safety. It also allows you to connect to multiple Command line wallets to your shop so you can accept more than one type of cryptocurrency.

- [How to install the VerusPay Plugin](#), armed with a running WordPress Website, next step is to setup the VerusPay Plugin that gives the customer the option to choose VRSC, KMD, ARRR, ZCASH & more cryptocurrencies as payment options.
- [How to test everything is setup correctly](#), to confirm everything went well and that your shop is up and running we'll guide you through all the checks and tests you need to do.

▼ Pages 7

Find a page...

▼ Home

 VerusPay Installation guide for WooCommerce

 ▶ [How to install the Verus Command Line Wallet](#)

 ▶ [How to install the VerusPay Plugin in WooCommerce](#)

 ▶ [How to install Verus ChainTools](#)

 ▶ [How to test that you have successfully set everything up](#)

 ▶ [Prerequisites](#)

 ▶ [WordPress Install](#)

Clone this wiki locally

<https://github.com/monkins1010/VerusPay.wiki.git>



[Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)



© 2024 GitHub, Inc.

How to create your first Verus ID with CLI -

This guide will explain how to get your first Verus ID using the Command Line Interface. I recommend reading it all down first, and then go through the process of creating an ID.

If you don't have the wallet installed yet, check our website at <https://veruscoin.io/wallet.html>. Scroll down to **Wallets** and choose your operating system.

Wallets

System requirements: 64bit CPU/OS. Windows, Linux (Ubuntu tested), or Mac High Sierra and upwards. 8GB Memory.

[v0.6.1 Release Notes](#)

These are the official codebases and wallets supported by the community, which have the most advanced Verus features.

Jumpstart your native wallet using this [bootstrap](#) and save hours on synchronizing.

[Click here for the bootstrap procedure](#)

Operating System	Verus-Enhanced Agama v0.6.1	Command Line v0.6.1
Windows	SHA256:cfcdd637b40590b1e17bd91ca0336e1...	SHA256:1b5fb64cad0bb29ab76b0215b7051e...
Linux	SHA256:9bfbe98a585d6cac97d14b9e0a2e4ed...	SHA256:b514dbc8ae804ef16ea5080751e2...
MacOS	SHA256:6506e7fe0c250d403a6d7f7cb2594a3...	SHA256:4984e04a3d5932b99e3bfcb8a74d3...

If you didn't set up your wallet before, please also refer to the guides on that page, especially see the "FAQ and How-to" section at <https://veruscoin.io/service.html>, and check the **Bootstrap procedure** (marked in above image) to speed up syncing of your wallet at https://veruscoin.io/downloads/how-to/how-to_bootstrap.html

If you have troubles, please visit us on Discord <https://discord.gg/VRKMP2S> and ask in the **#community-support** channel, so somebody can support you.

Ok, back to topic. This guide will explain how you can get your first ID with a clean and fresh installed CLI wallet. Some of the steps will no longer apply when you already have a working version of that wallet, but you will notice.

Getting a [Verus ID](#) requires a balance of 100.0002 VRSC in your wallet (or 80.0002 if you have a referral ID). The ID will cost 100 (or 80) VRSC, and there is a transaction fee of 2 x 0.0001 VRSC (at time of writing this guide). So, if you don't have that much, you might want to get that before. If you don't know how, please also see <https://veruscoin.io/getVRSC.html> for that.

When you want to create an ID:

- start the CLI daemon (The example below will run as a daemon and also stake your balance and mine on 1 thread)

```
./verusd -mint -gen -genproclimit=1 -daemon 1>/dev/null 2>&1
```

- Once the daemon is started, check your blockheight and compare it to the current blockheight on <https://explorer.veruscoin.io>

```
./verus checkblockcount
```

which should result in a blocknumber:

838996

- Now it is time to check if you have sufficient funds to acquire an ID. Check your balance with:

```
./verus z_gettotalbalance
```

- which should show an output similar to this:

```
{
  "transparent": "640.0310769",
  "interest": "0.00",
  "private": "0.00",
  "total": "640.0310769"
}
```

- If it's less than 100, you need to get some more before you can create an ID. If you have recently sent funds to your address and that isn't included in this balance, you can wait for the transfer to show up.

- To create an ID, you have to check first if that ID already exists with this command (I'll be using **Demo ID** in this guide):

```
./verus getidentity "Demo ID@"
```

If the identity is not yet registered you get the following output:

```
error code: -5
error message:
Identity not found
```

If the identity is already registered, you will be provided with details about that ID

- You can use any character except **\ / : . * ? " < > | @** for your ID
 - And you can not use an empty space as a first character.
 - But don't worry, the system won't let you commit such names, even if you try.
 - What you can use is e.g. an empty space in between, which would allow you to commit a name like "Lucky Joe" (without the quotes of course), but also "Lucky_Joe", "LuckyJoe", and even "Lückÿ Jöë" or "幸运 八八八" would work. There is no rule for creating a useful ID, and you can create several IDs if you like. That means, you can create a matching ID for each purpose.

- The command to commit a name for your ID:

```
registernamecommitment "Demo ID" "RFoinkqd2QinMi25mUu5h7k3f4ETaBQwpo" "Verus Coin Foundation@"
```

- The text between the **first pair** of quotes is your ID name
- The text between the **second pair** of quotes is the address in your wallet that pays the fees to create this ID
- The text between the **third pair** of quotes is the optional referral ID, which lowers your purchase price.

The command above will return you a json formatted message:

```
{
  "txid": "862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c",
  "namereservation": {
    "name": "Demo ID",
    "salt": "739e248f525c8fbbb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261",
    "referral": "i5v3h9FwvdRFbNHU7DfcpgykQjRaHtMqu7",
    "parent": "",
    "nameid": "iRLC6sRksACDVRYEoz32iY6kE9ax5xyTyE"
  }
}
```

- The message that the command above returned, contains the information which is needed for the registration of the ID, but the above transaction first has to be mined into the blockchain, so you will have to wait at least 1 block before you can issue the next command. To make sure the transaction is mined into the chain you can check the TXID from that message on the explorer:
<https://explorer.veruscoin.io>

- The command to register your ID:

```
./verus registeridentity '{"txid":  
"862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c", "namereservation": { "name":  
"Demo ID", "salt": "739e248f525c8fb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261", "referral":  
"i5v3h9FwVdRFbNHU7DfcGykJQjRaHtMqu7", "parent": "", "nameid":  
"iRLC6sRksACDVRYEoz32iY6kE9ax5xyTyE"}, "identity": {"name": "Demo ID",  
"primaryaddresses": ["RFoinkqd2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures": 1,  
"privateaddress":  
"zs16xdne62g2w162vq8kmw103k1v4qs5zf3jxmax9vj7f443ymj59ds4qtvj8tp67zvs58fvvpc22n"} }'
```

- The data shown in red is copied from the response message from the commit command.
- The data shown in green is your public address that your ID is bound to.
- The data shown in purple, is your private address, which can be used for private transfers, messaging, voting, polling etcetera.
- After running this command the CLI will respond with a single line containing the TXID:
de2909a8832a94182037f3ea29f7382b5a2841f505ab20ceb8334e3712127078

- Once this TXID is mined into a block on the blockchain (again: you can use the explorer to check), your ID is registered. You can check the registration using:

```
./verus listidentities
```

The output will show you all identities in your wallet:

```
[
  {
    "identity": {
      "version": 1,
      "flags": 0,
      "primaryaddresses": [
        "RFoinkqd2QinMi25mUu5h7k3f4ETaBQWpo"
      ],
      "minimumsignatures": 1,
      "identityaddress": "iNcmBb1z1pdw2qw4UDhNp1MXg7Rq9Yk8ge",
      "parent": "RQVsJRF98iq8YmRQdehzRcbLGHEx6YfjdH",
      "name": "!",
      "contentmap": {
      },
      "revocationauthority": "iNcmBb1z1pdw2qw4UDhNp1MXg7Rq9Yk8ge",
      "recoveryauthority": "iNcmBb1z1pdw2qw4UDhNp1MXg7Rq9Yk8ge",
      "privateaddress": "zs1sx3yklu4ezq8ht5xnghrjhm4jxs8mcgxaqlky50e548mqngv8u8vnptrluq5w9jft2ackpxsh9e"
    },
    "blockheight": 3701,
    "txid": "48ba2a4e1a5e5ee0cada495b268b85679fc43bf573a3ad86aa1e117e37f1b848",
    "status": "active",
    "canspendfor": true,
    "cansignfor": true
  }
]
```

The procedure above creates IDs that have all authorities set to themselves.

- If you want to update your ID after creating, you can use the command **updateidentity**. In our example it would look like this, if I wanted to update the revocation authority and recovery authority to other existing identities:

```
./verus updateidentity '{"version": 1, "flags": 0, "primaryaddresses": ["RFoinkqd2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures": 1, "identityaddress": "iNcmBb1z1pdw2qw4UDhNp1MXg7Rq9Yk8ge", "parent": "RQVsJRF98iq8YmRQdehzRcbLGHex6YfjdH", "name": "Demo ID", "contenthashes": [], "revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55", "recoveryauthority": "iLbcafWjFgvrZ9yGNKofWmBw1DgJyAave2", "privateaddress": "zs1sx3yk1u4ezq8ht5xnghrjh4jxs8mcgxaqlky50e548mqngv8u8vnptrluq5w9jft2ackpxsh9e"}'
```

- The ID addresses in red reflect the changes I made, to assign the authorities to other IDs.
- The **"revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55"** and **"recoveryauthority": "iLbcafWjFgvrZ9yGNKofWmBw1DgJyAave2"** can also be added to the registration process, to assign these authorities right away. The resulting command would look like this, in our example:

```
./verus registeridentity '{"txid": "862485a6af4b6989531cf530e5a13b531243c805702de40b0d8086f3527c174c", "namereservation": { "name": "Demo ID", "salt": "739e248f525c8fb01c4c480a6c325c10e28ab1a00dc0f81b6583e1b9cab261", "referral": "i5v3h9FWvdRFbNHU7DfcPgykQjRaHtMqu7", "parent": "", "nameid": "iRLC6sRksACDVRYEoz32iY6kE9ax5xyTyE"}, "identity": {"name": "Demo ID", "primaryaddresses": ["RFoinkqd2QinMi25mUu5h7k3f4ETaBQWpo"], "minimumsignatures": 1, "privateaddress": "zs16xdne62g2w162vq8kmw103k1v4qs5zf3jxmax9vj7f443ymj59ds4qtvj8tp67zvs58fvvpc22n", "revocationauthority": "i5vt1Cx7oDMjfFSNSFpdMFQLoyD9gqDN55", "recoveryauthority": "iLbcafWjFgvrZ9yGNKofWmBw1DgJyAave2"}}'
```

- The data shown in **red** is copied from the response message from the commit command.
- The data shown in **green** is your public address that your ID is bound to.
- The data shown in **purple**, is your private address, which can be used for private transfers, messaging, voting, polling etcetera.
- The data shown in **Dark yellow**, are the i-addresses for the revocation and recovery addresses. You can use the ID-name too, instead of the i-addresses.

**) Some notes on IDs

You see that you can enter three IDs, viz. the **main ID**, the **revocation ID**, and the **recovery ID**. Actually they are equal for their own purpose, each of them is an own, normal, independent ID, and can do anything the **main ID** can do on their own. They don't even have to be your IDs. But they just can be assigned to different roles (a.k.a. authority). If you choose to do so, here's what can be controlled by them:

- The **main ID** is the one you would do all transactions with, and other stuff that is still to come. It always has the power to change both of the other IDs.
- The **Revocation ID** can revoke the **main ID**, but also has control over changing anything that is related to the **Revocation ID** itself. This means, if you change the **Revocation ID** to an ID you do not control, you no longer can change the **Revocation ID** again, except from the wallet that also controls the **Revocation ID**. This also applies to **Recovery ID**.
- The **Recovery ID** can of course recover the **main ID** *when it was revoked*. And (only) in this process, it can also change the *public and private addresses* of the **main ID**, and it's **Revocation ID**.
- Both the **Revocation ID** and the **Recovery ID** can be used for more than one **main ID**, so you only need to create them once. And they can be the same, too, so you can use one other ID both for **revocation** and **recovery**.
- **If you want to be able to revoke and recover your main ID, you need at least one more ID for that. The main ID can not revoke itself.** However, if you want to have a totally self-sovereign ID, and control the private and public address it is assigned to in another way, you can use self-assignment.
- **Revocation and Recovery ID(s) should be created before your main ID.** Although there is a way to change them later, it's rather complicated (at time of writing this guide). So if you plan to be able to revoke and recover, it's much easier to create them first.

Now you see what these IDs can control. What they *do not* control, nor revoke or recover, are the underlying *public (R-)* and *private (z-)* addresses that you enter on creation. These will always be in control of the wallet that has the private *keys* (a.k.a. WIFs) for them. They will stay unchanged, can't neither be revoked nor recovered. That is only possible for **IDs**.

You will, however, find a new address after creating an ID, which is an *i-address*. This address is tied to an ID, and only exists together with a valid ID. It does not have a private *key*, and can not be restored other than by the associated ID.

For the experienced user it may even be an option to create an ID (e.g., for revocation or recovery) on a new wallet, with a separate wallet.dat file. The wallet.dat can be saved and secured offline, so it is at hand when needed. But you only should do so if you're really sure what you are doing. If you would like to have some support with this, feel free to ask in our Discord <https://discord.gg/VRKMP2S>.

Now, after you know a bit more about IDs, you can decide how many and which IDs you want to create.

Finally, a few notes on referrals.

When you enter a referral ID, purchasing an ID will cost 80 instead of 100 VRSC (not including transaction fees). At the same time, the referral ID will get 20 VRSC.

Furthermore, when someone is using your ID as a referral, you will get 20, and the referral you were using will again get 20 VRSC (but purchase will not get cheaper than 80 VRSC).

This referral propagation will work up to the 3rd generation, i.e., up to the referrer of the referrer of the referrer. Each of them will have 20 VRSC. But at least 20 VRSC will always go back to the miners and stakers of Verus Coin, and will be added to one of the coming blocks. That's also why mining and staking will become more profitable with each ID that was created.

Thanks for reading, and don't hesitate to ask in our discord if you need support! :)

veruscoin.io

How to create your first Verus ID -

This guide will explain how to get your first Verus ID using the *Verus Desktop* wallet. I recommend reading it all down first, and then go through the process of creating an ID.

If you don't have the wallet installed yet, check our website at <https://veruscoin.io/wallet.html>. Scroll down to [Verus Desktop](#) and choose your operating system.

Jumpstart your native wallet using
this [bootstrap](#) and save hours on
synchronizing. [click here for bootstrap procedure](#)

Verus Desktop

System requirements: 64bit CPU/OS.
Windows, Linux (Ubuntu tested), or
Mac High Sierra and upwards. 8GB
Memory.

[Verus Desktop v0.0.7-alpha Release Notes](#)

Although some will choose to use Verus-Connect in place of Agama, this is an Alpha release.
Currently, Verus-Desktop is still in development.
For this reason, we currently still recommend using the Verus-Enhanced Agama Wallet.

 Windows

 Linux

 MacOS

Verus Desktop v0.0.7-alpha

SHA256:[c4287a95fc0669718261ad8...](#)

Verus Desktop v0.0.7-alpha

SHA256:[f01b5acb58394d0456eecca...](#)

Verus Desktop v0.0.7-alpha

SHA256:[34584c92640e217b4bb441...](#)

If you didn't set up Verus Agama or Verus Desktop before, please also refer to the guides on that page, especially see the "FAQ and How-to" section at <https://veruscoin.io/service.html>, and check the **Bootstrap procedure** (marked in above image) to speed up syncing of your wallet at https://veruscoin.io/downloads/how-to/how-to_bootstrap.html

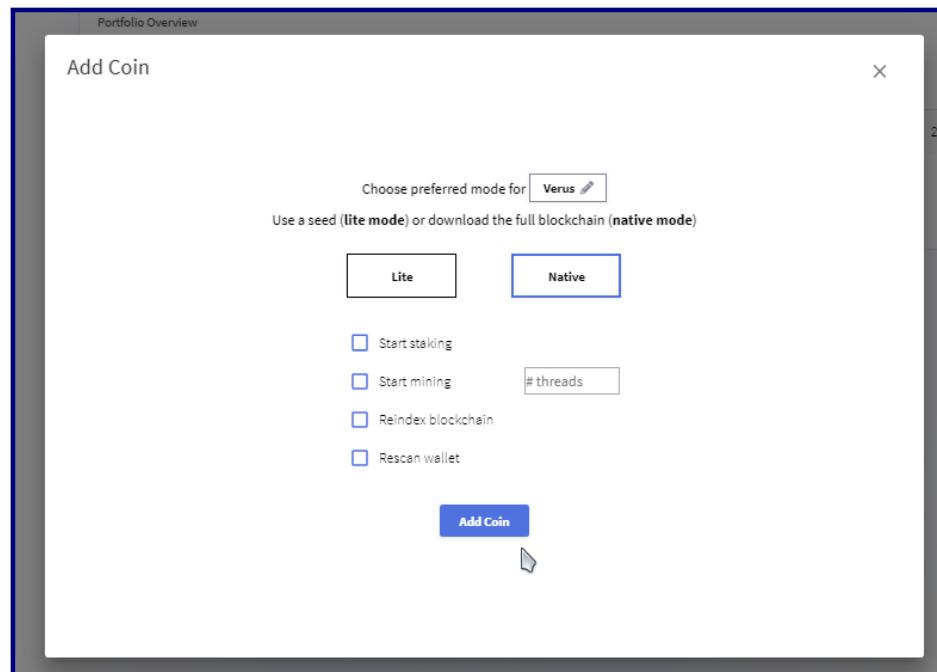
If you have troubles, please visit us on Discord <https://discord.gg/VRKMP2S> and ask in the **#community-support** channel, so somebody can support you.

Ok, back to topic. This guide will explain how you can get your first ID with a clean and fresh installed Verus Desktop wallet. Some of the steps will no longer apply when you already have a working version of that wallet, but you will notice.

Getting a [Verus ID](#) requires a balance of 100.0002 VRSC in your wallet (or 80.0002 if you have a referral ID). The ID will cost 100 (or 80) VRSC, and there is a transaction fee of 2 x 0.0001 VRSC (at time of writing this guide). So, if you don't have that much, you might want to get that before. If you don't know how, please also see <https://veruscoin.io/getVRSC.html> for that.

When you want to create an ID:

- start Verus Desktop
- create a profile name
- select "Wallet" from the "Get started" menu
- click "add a coin"
- from the drop down menu, select Verus and click "continue"
- on the next screen, select "native" (should be preselected), but don't select any of the boxes
- now click "Add Coin"



On the next screen you will see the Dashboard.

Verus Desktop (v0.0.7-alpha)

Edit View Window Debug

Wallet Verus Identities Mining \$ - ⚙️ ⏮

Dashboard + Add Coin

Verus NATIVE EUR VRSC

Total Portfolio Value EUR Profile me Profile Settings

Portfolio Overview

100% Verus VRSC EUR -0.39%

Verus 0.07 EUR/VRSC -0.39%

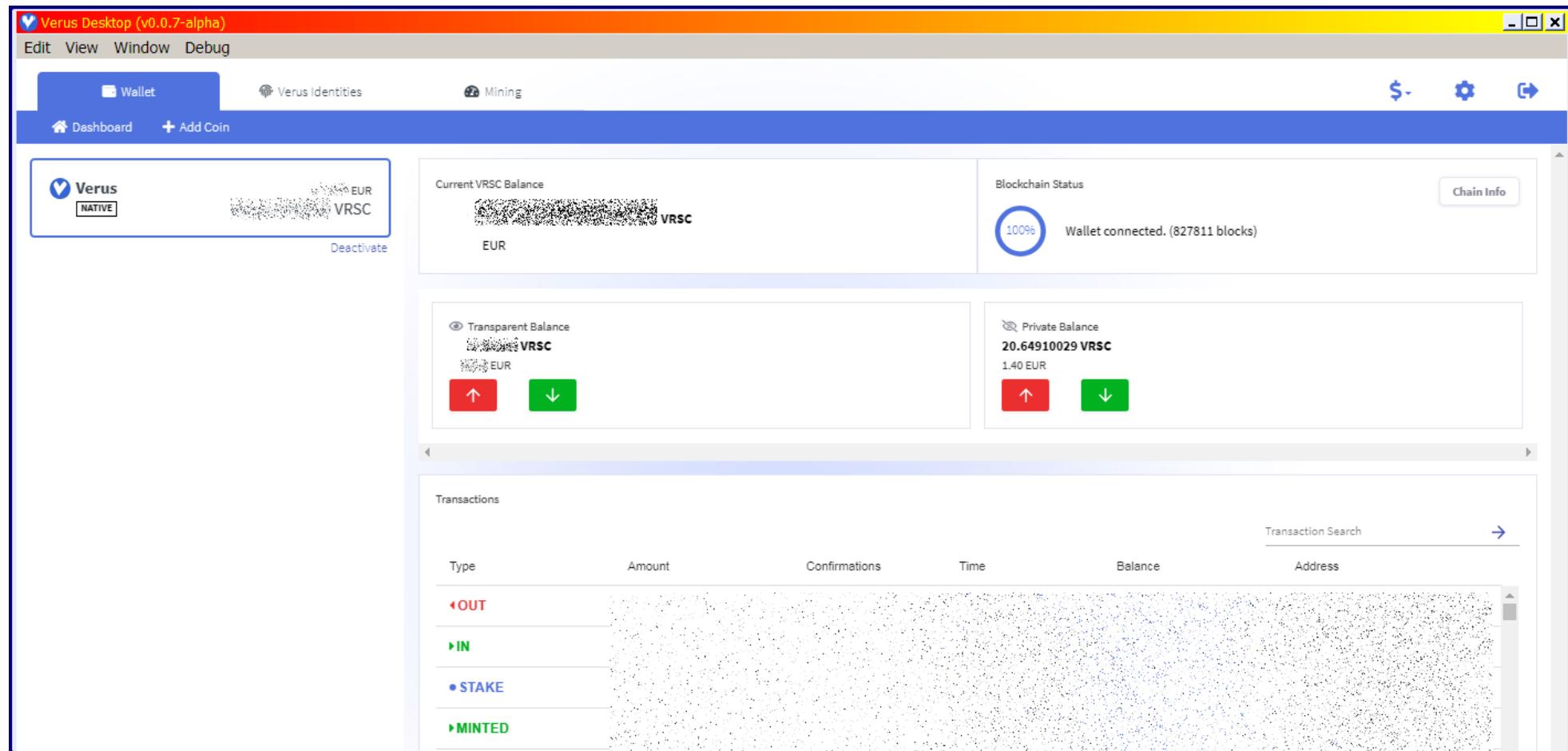
The screenshot shows the Verus Desktop application interface. At the top, there's a navigation bar with 'Edit', 'View', 'Window', and 'Debug' options. Below that is a header with 'Wallet', 'Verus Identities', 'Mining', and a currency dropdown set to '\$ -'. There are also settings and a help icon. The main area has a blue header bar with 'Dashboard' and '+ Add Coin' buttons. On the left, there's a card for 'Verus NATIVE' with a QR code and 'VRSC' below it. The central part shows a large blue circle, likely representing the total portfolio value. Below it is a table for the portfolio overview:

100%	Verus	VRSC	EUR	-0.39%
------	-------	------	-----	--------

At the bottom, there's another table for the coin price:

Verus	0.07 EUR/VRSC	-0.39%
-------	---------------	--------

To look at your Verus, click on the badge on the left.



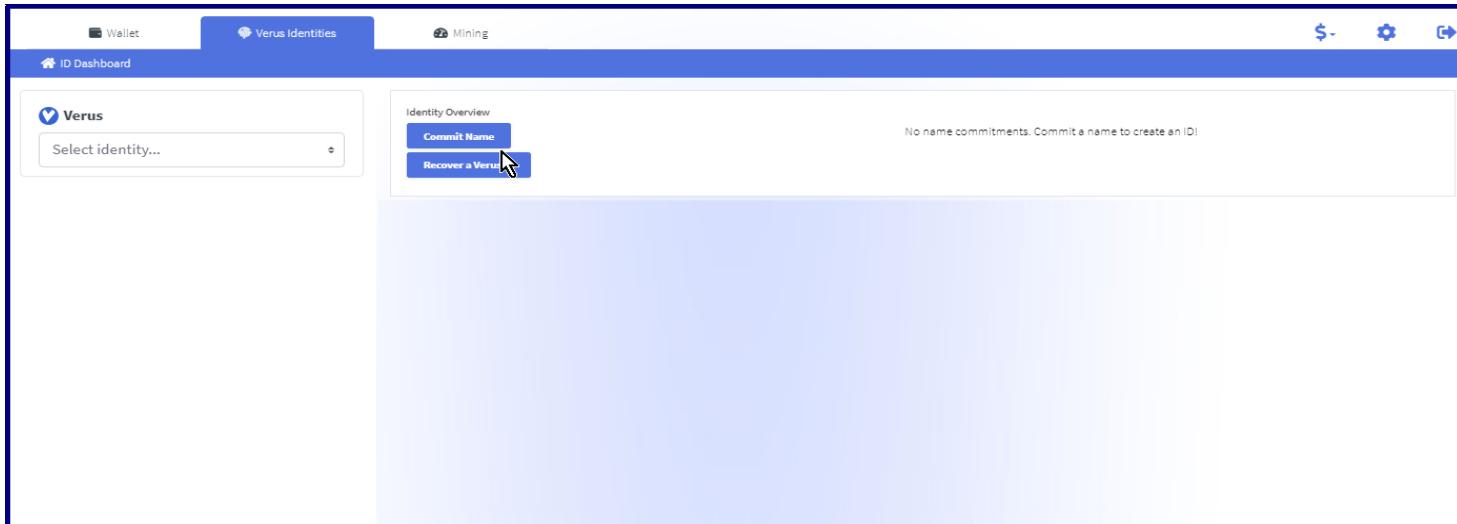
You would have to wait now until the wallet is synced with the blockchain. You can see this when you watch the Blockchain status on the top right. As soon as you have a "100%" showing in the blue circle, you're good to continue.

You should also see your available Verus balance under "Current VRSC balance" now. If it's less than 100, you need to get some more before you can create an ID. If you still have coins in "Pending VRSC balance" and altogether you would have enough, you can continue.

Now, click on the "Verus Identities" tab (on top, above the blue ribbon).



If this is your first identity, there should be not much more to see than a "Select Identity..." under the Verus badge on the left, besides a "Commit Name" above a "Recover a Verus ID" button in the Identity Overview on the right.

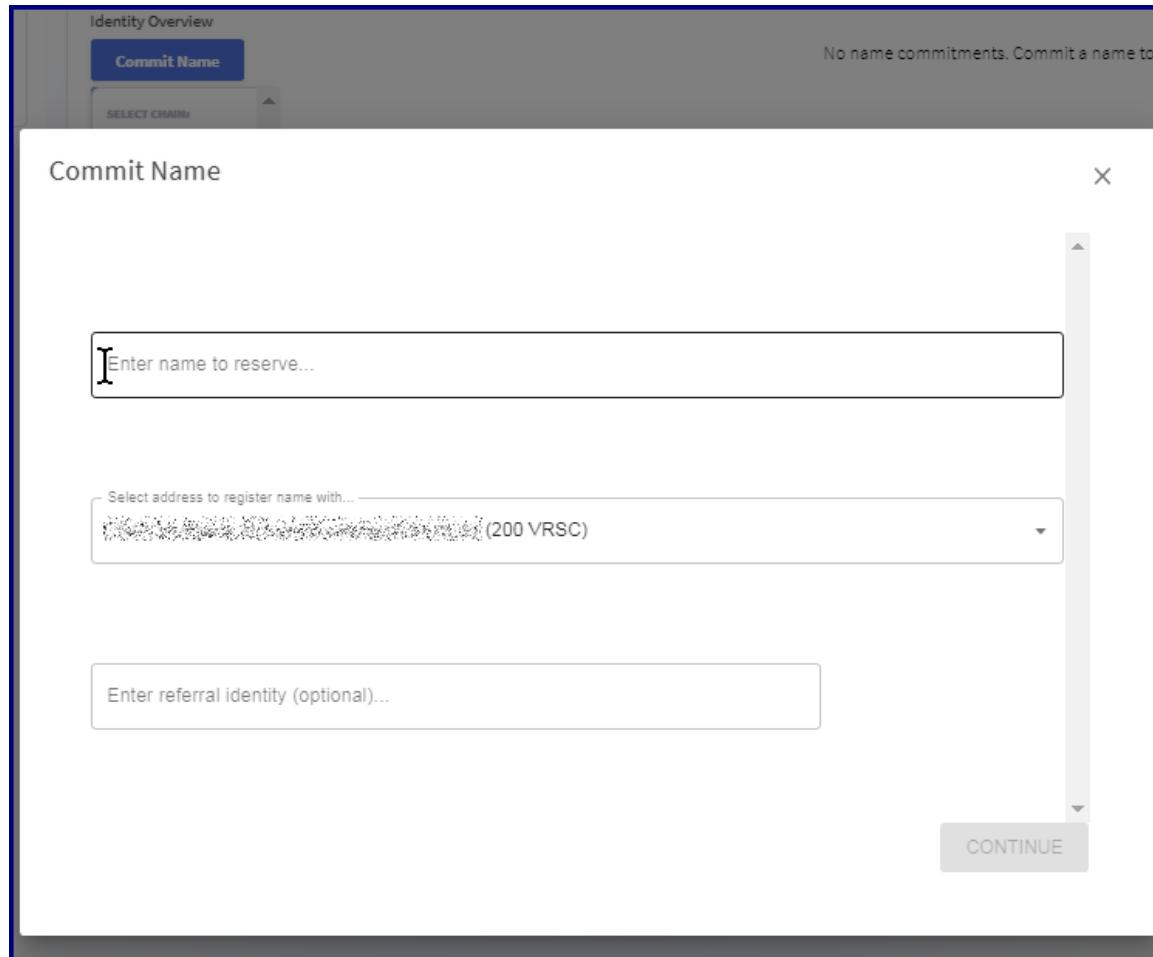


To create an ID, you have to check first if that ID already exists. If not, you can reserve the name at the same time.

To do so, click on "Commit Name".

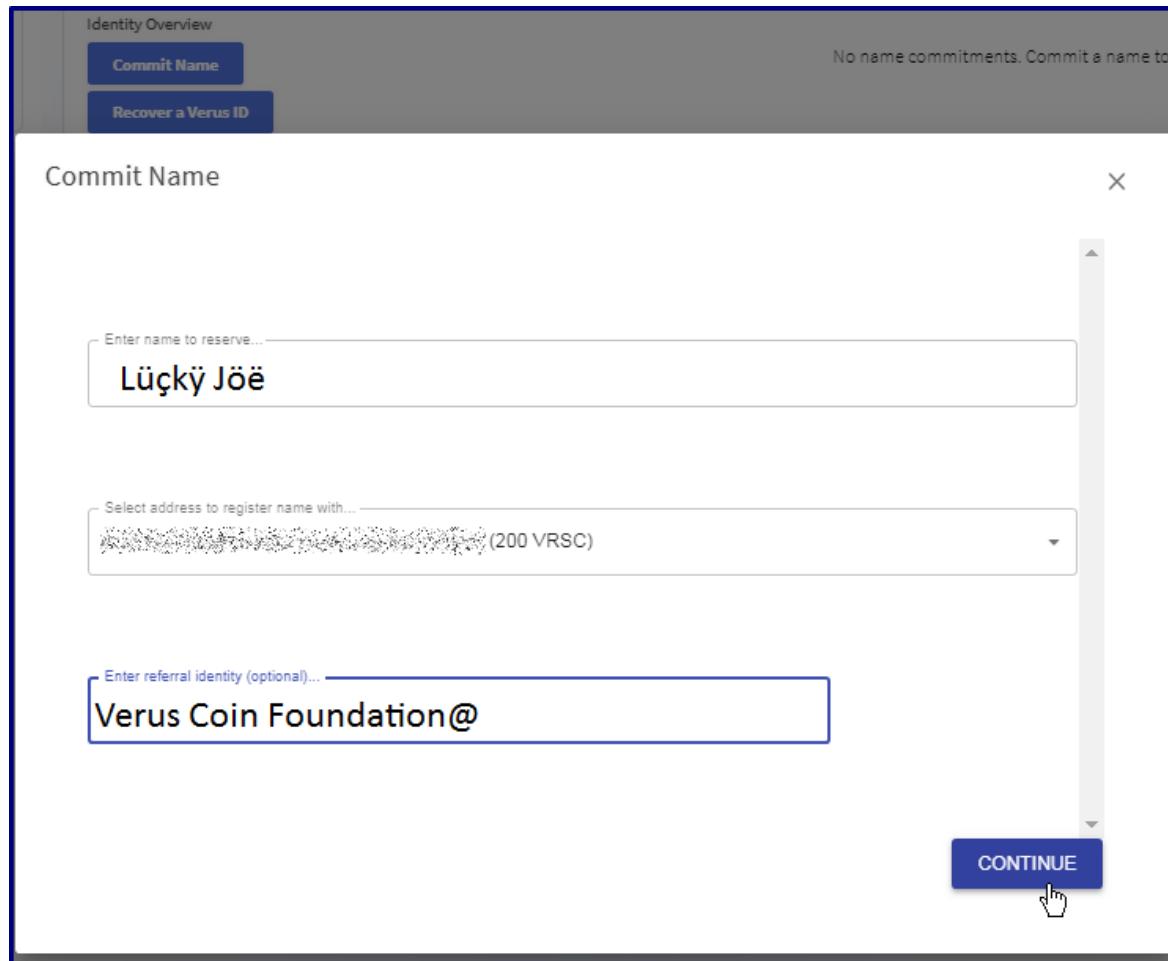
You now will see a drop down menu, which only shows Verus (at time of writing this guide). Reason for this is the coming PBaaS, but that will be explained elsewhere. So, for now, select Verus and click on it.

- In the pop up window, first line, you can now enter your desired ID name. You can enter any character except \/:.*?"<>|@ And you can not use an empty space as a first character.
But don't worry, the system won't let you commit such names, even if you try.
What you can use is e.g. an empty space in between, which would allow you to commit a name like "Lucky Joe" (without the quotes of course), but also "Lucky_Joe", "LuckyJoe", and even "Lücký Jöë" or "幸运 八八八" would work. There is no rule for creating a useful ID, and you can create several IDs if you like. That means, you can create a matching ID for each purpose.



- in the line below, you can select the public address that will be linked to your ID (private address will be linked later). You can create an ID on an empty address, but you need to have the fee of 100 (or 80) VRSC and the transaction fee of 0.0002 (at time of writing this guide) in your wallet.
- in the last line, you can enter an existing ID for referral.
You can leave that referral empty, then your ID would cost 100 VRSC. Or you can enter a valid ID, then your ID would cost 80 VRSC, while 20 VRSC will go to the referral ID at the same time.
If you don't have a referral ID, you can use [Verus Coin Foundation@](#), which will make 20 VRSC go directly to the foundation and help the project development. Or if you want to donate a bit to the author of this guide, you can use [me@](#) as a referral ID (to learn more about referral bounties, please read further at the end of this guide).

After entering all data, the mask should look similar to this:



The screenshot shows a 'Commit Name' dialog box. At the top, there are three buttons: 'Identity Overview', 'Commit Name' (which is highlighted in blue), and 'Recover a Verus ID'. The main area has a heading 'Commit Name' and a close button 'X'. It contains three input fields: 'Enter name to reserve...' with the text 'Lückÿ Jöë', 'Select address to register name with...' with a dropdown menu showing a blurred address and '(200 VRSC)', and 'Enter referral identity (optional)...' with the text 'Verus Coin Foundation@'. A 'CONTINUE' button is at the bottom, with a hand cursor icon pointing to it.

Ok, after you made the entries and the selection, click "continue".

You will now see a pop up with all the information for confirmation.

Commit Name

Control Address:

Name:

Referral ID:

BACK

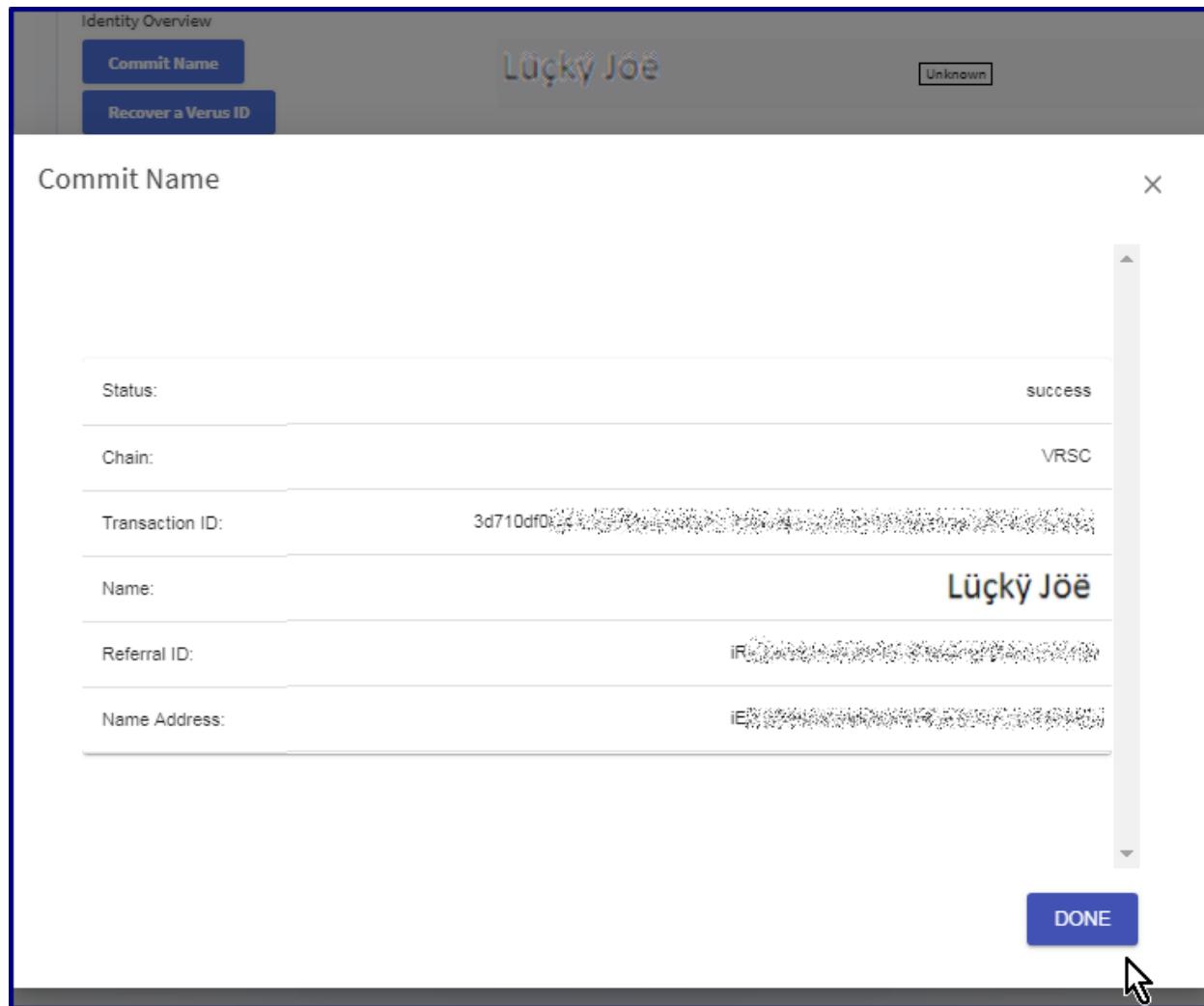
CONTINUE

If you see anything wrong there, you can now still go back and revise your entries. But if everything is ok, click "continue" and go on.

Now, when you entered a valid ID name, you will shortly see a window with the progress of the process, just before you get a confirmation window that shows what was processed.

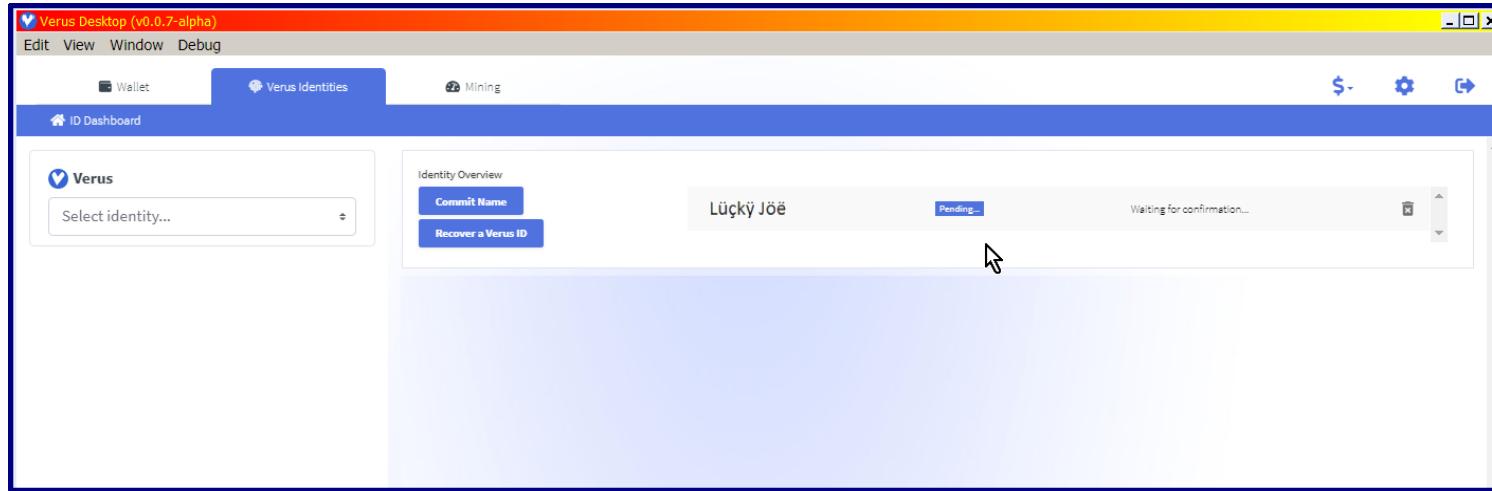
If anything did not work, e.g. if the ID name was not valid or already registered, you will see an error. In this case, go back and change what is necessary.

Shortly after you confirmed all and it worked, you should see a "Commit Name" window pop up. This window will confirm that the process of checking and registering your ID name has successfully started.

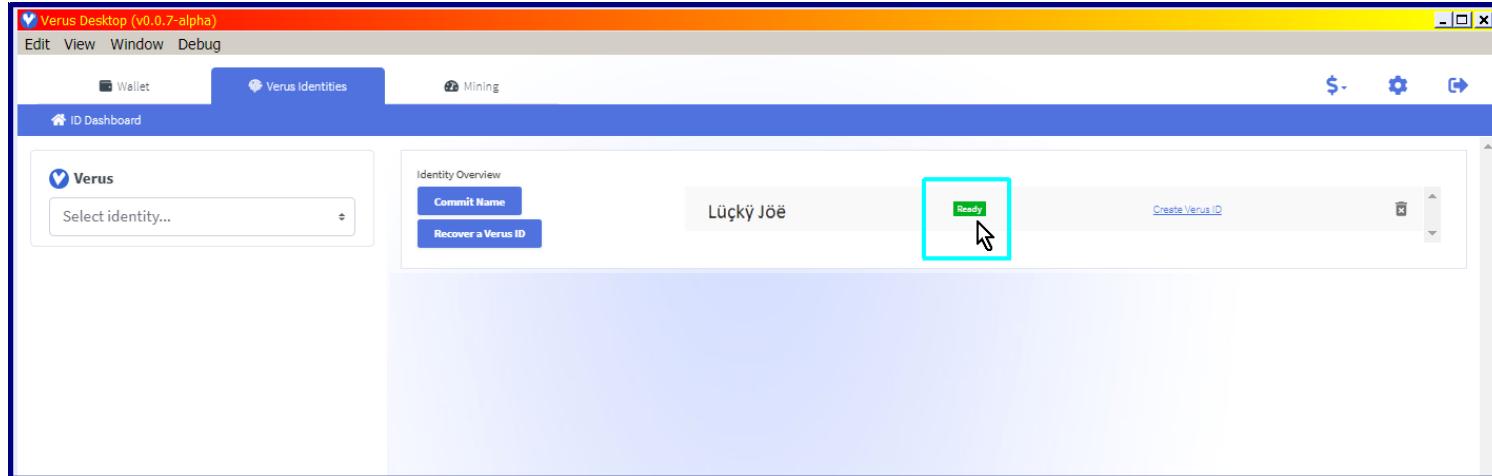


You can close this window by clicking on "Done".

When the "Commit Name" window is closed, you will see the status of the name checking and reservation on the top right of your Identity Overview window.



As soon as the status changes from a blue "Pending" to a green "Ready", we can finally go on to register your ID.



Please note: You haven't purchased your ID yet!

Until now, we only checked and reserved a name, and we set the basics that are necessary for that. The ID is not yet registered!

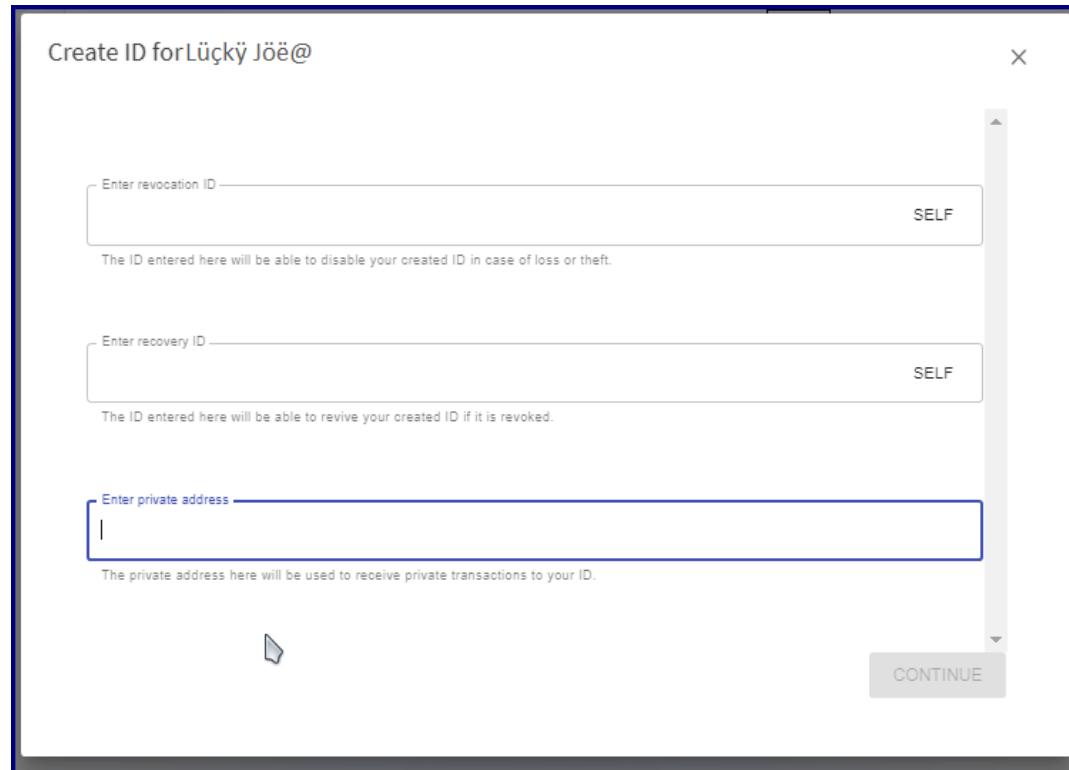
You can also see this in your transactions, where you will see one outgoing and one incoming transaction of 0 VRSC from and to the public address you have chosen above. The process of checking and reservation is for free (except for one transaction fee).

So, until this point, you can still cancel the process by clicking the grey trash can with an X on it, next to that. If you don't want to cancel, let's continue.

Now, we will finally register your ID.

Please click on "[Create Verus ID](#)", which is right to the „Ready“ status note in your Identity Overview.

You will see a window pop up with three fields to fill.



These fields are:

- "Enter revocation ID": If you have another ID, or a friend, or family member with an ID you really trust**, you can enter it here. This ID will be able to revoke your ID. It's a safety in case your ID gets compromised in any way, or if you can't access your wallet any more, or similar. If you don't know of any ID you want to enter here, you can leave it blank. If you want to make the ID independent and fully self-sovereign, you can click the "SELF" button. But you may also leave that line empty..
- "Enter recovery ID": This ID will allow you to recover a revoked ID **, to make it valid again. Same rules as above, if you have somebody with an ID you trust, or you already have another ID, you can use it. For self-sovereign IDs, use the "SELF" button. And you can of course leave it empty, too.**
- "Enter private address": Ok, that's needed to receive private transaction. Usually we don't have that memorised, so we need to go back to the wallet and copy the address.

****) Some notes on IDs**

You see that you can enter three IDs, viz. the **main ID**, the **revocation ID**, and the **recovery ID**. Actually they are equal for their own purpose, each of them is an own, normal, independent ID, and can do anything the **main ID** can do on their own. They don't even have to be your IDs. But they just can be assigned to different roles (a.k.a. authority). If you choose to do so, here's what can be controlled by them:

- ◊ The **main ID** is the one you would do all transactions with, and other stuff that is still to come. It always has the power to change both of the other IDs.
- ◊ The **Revocation ID** can revoke the **main ID**, but also has control over changing anything that is related to the **Revocation ID** itself. This means, if you change the **Revocation ID** to an ID you do not control, you no longer can change the **Revocation ID** again, except from the wallet that also controls the **Revocation ID**. This also applies to **Recovery ID**.
- ◊ The **Recovery ID** can of course recover the **main ID** *when it was revoked*. And (only) in this process, it can also change the *public and private addresses* of the **main ID**, and it's **Revocation ID**.
- ◊ Both the **Revocation ID** and the **Recovery ID** can be used for more than one **main ID**, so you only need to create them once. And they can be the same, too, so you can use one other ID both for **revocation** and **recovery**.
- ◊ **If you want to be able to revoke and recover your main ID, you need at least one more ID for that. The main ID can not revoke itself.** However, if you want to have a totally self-sovereign ID, and control the private and public address it is assigned to in another way, you can use self-assignment.
- ◊ **Revocation and Recovery ID(s) should be created before your main ID.** Although there is a way to change them later, it's rather complicated (at time of writing this guide). So if you plan to be able to revoke and recover, it's much easier to create them first.

Now you see what these IDs can control. What they *do not* control, nor revoke or recover, are the underlying *public (R-) and private (z-)* *addresses* that you enter on creation. These will always be in control of the wallet that has the private *keys* (a.k.a. WIFs) for them. They will stay unchanged, can't neither be revoked nor recovered. That is only possible for **IDs**.

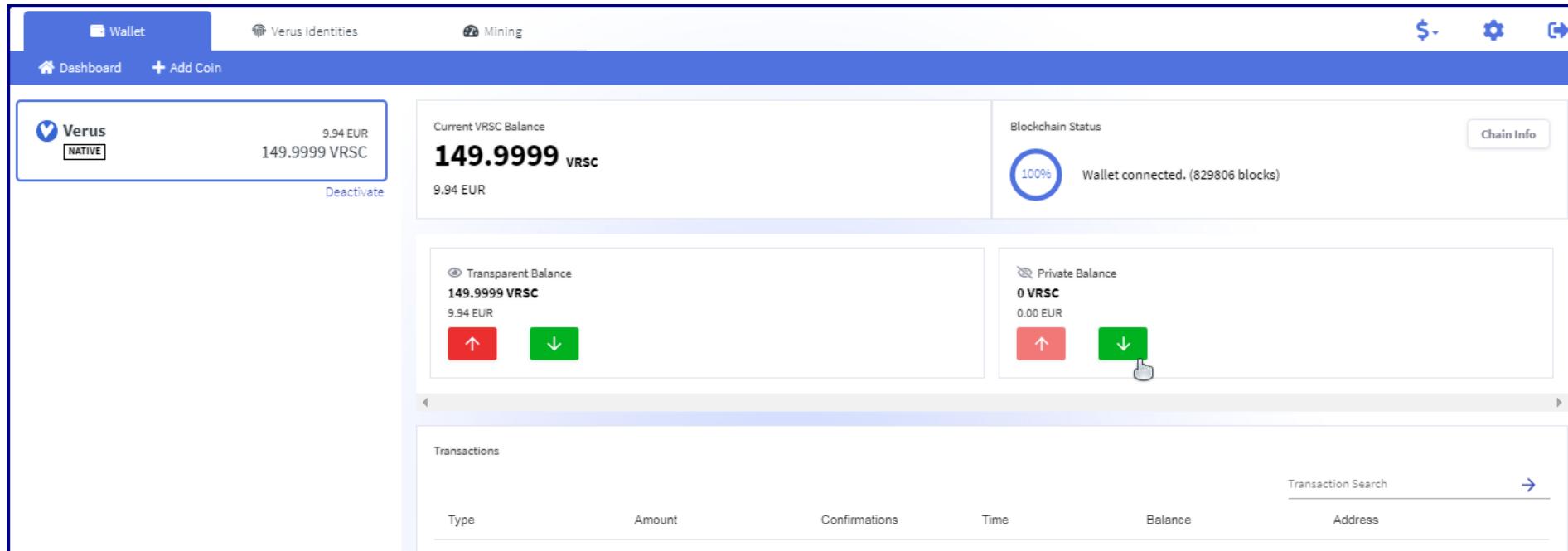
You will, however, find a new address after creating an ID, which is an *i-address*. This address is tied to an ID, and only exists together with a valid ID. It does not have a private *key*, and can not be restored other than by the associated ID.

For the experienced user it may even be an option to create an ID (e.g., for revocation or recovery) on a new wallet, with a separate wallet.dat file. The wallet.dat can be saved and secured offline, so it is at hand when needed. But you only should do so if you're really sure what you are doing. If you would like to have some support with this, feel free to ask in our Discord <https://discord.gg/VRKMP2S>.

Now, after you know a bit more about IDs, you can decide how many and which IDs you want to create.

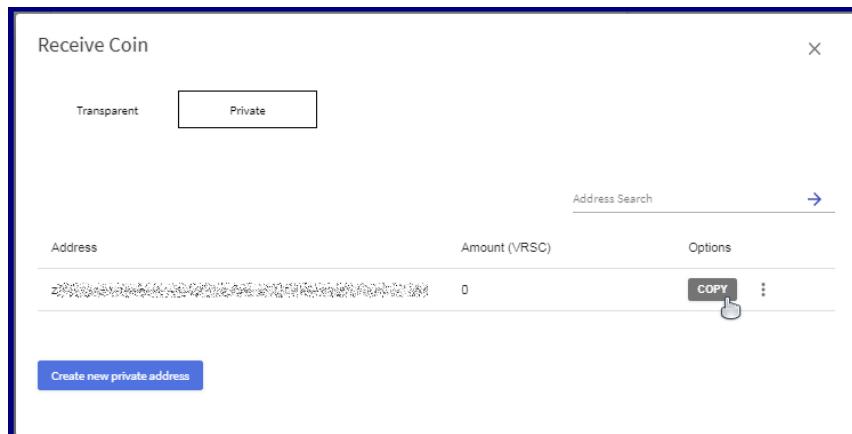
Let's go on in creating your first ID, go back to the Wallet tab (on the left, above the blue ribbon).

There, when you click on the Verus badge, you can see your funds, both the transparent (i.e., public), and the private balance.



The screenshot shows the Verus Wallet interface. At the top, there are tabs for 'Wallet', 'Verus Identities', and 'Mining'. On the left, a 'Dashboard' and '+ Add Coin' button are visible. A 'Verus' badge is highlighted, showing '9.94 EUR' and '149.9999 VRSC'. A 'Deactivate' button is below it. In the center, the 'Current VRSC Balance' is displayed as **149.9999 VRSC**. Below it, the 'Transparent Balance' is shown as **149.9999 VRSC** and **9.94 EUR**, with red and green up/down arrows. To the right, the 'Blockchain Status' shows a 100% progress bar and the message 'Wallet connected. (829806 blocks)'. The bottom section shows a table for 'Transactions' with columns for Type, Amount, Confirmations, Time, Balance, and Address. A 'Transaction Search' bar is at the bottom right.

To copy a private address, you simply click on the green down-arrow in the Private Balance badge.

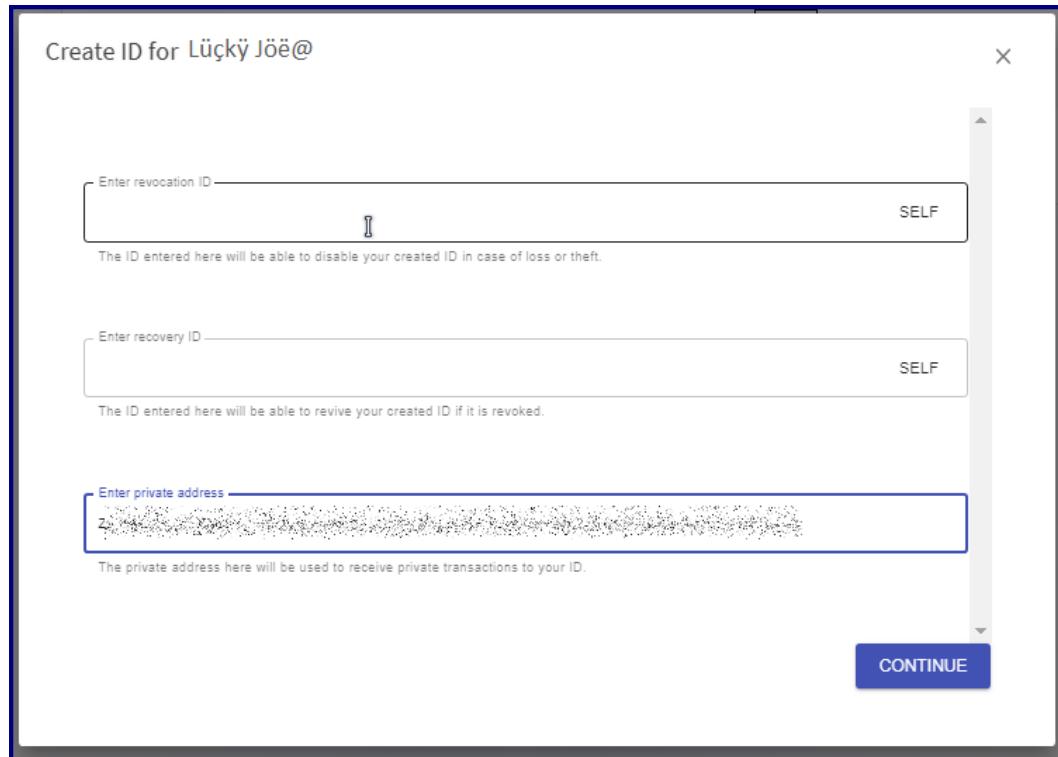


The screenshot shows the 'Receive Coin' dialog. It has tabs for 'Transparent' and 'Private', with 'Private' selected. An 'Address Search' bar is at the top right. Below it, there are fields for 'Address', 'Amount (/VRSC)', and 'Options'. The 'Address' field contains a long string of characters. The 'Amount (/VRSC)' field has a value of '0'. The 'Options' field has a 'COPY' button and a dropdown menu. At the bottom, there is a blue 'Create new private address' button.

If you don't have a private address yet, or you don't want to use one of those you have, now simply create a new private (z-) address by clicking the blue button.

To copy the private address then just click the "Copy" button next to your private address.

Now you can go back to the "Verus Identities" tab and click again on "Create Verus ID" for your reserved name. As you don't have a revocation or recovery ID yet we will leave that empty for now. Just paste your private address in the last field and click "Continue".



Now a confirmation window will pop up that shows all the information you entered. It will also show the fee that will be deducted for purchasing, and if you entered a Referral ID, the i-address of which is also shown.

If you want to cancel the creation of your ID, this will be the last possibility to do that.

However, if you want to create the ID you can see there, click on „Continue“.

Create ID for Lückÿ Jöë@

X

Est. Cost:

80 VRSC

Referral:

[REDACTED]

Name:

Lückÿ Jöë

Chain:

VRSC

Primary Address:

[REDACTED]

Revocation ID:

Recovery ID:

Private Address:

BACK

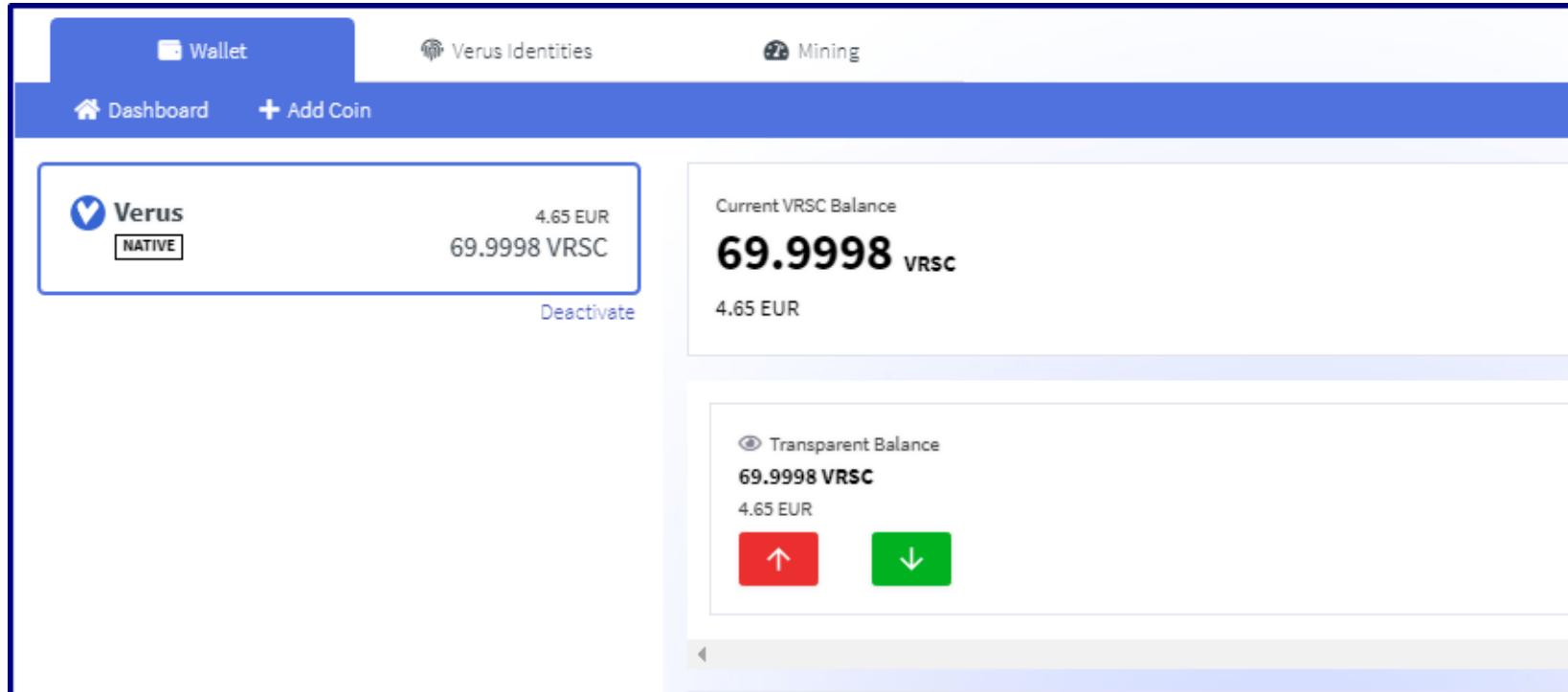
CONTINUE



You may now see a window popping up shortly for the process.

But right after that, the confirmation window will show again, together with a blue ribbon below it to confirm that your 100 (or 80) Verus coins were transferred to purchase your ID.

All you have to do now is wait a little, until the transaction was confirmed by the network. You may notice that the fee was deducted from your funds, also visible in transactions, and your wallet shows the remaining funds.



The screenshot shows the Verus wallet interface. The top navigation bar includes tabs for Wallet, Verus Identities, and Mining. Below the navigation is a blue header bar with Dashboard and Add Coin buttons. The main content area features a box for the Verus coin, which is labeled as NATIVE and shows a balance of 4.65 EUR and 69.9998 VRSC. A Deactivate button is located below this box. To the right, a Mining section displays the Current VRSC Balance as 69.9998 VRSC and 4.65 EUR. Below the mining section is a box for a Transparent Balance, also showing 69.9998 VRSC and 4.65 EUR, with up and down buttons for adjusting the balance.

And then - right after one confirmation of the network - your ID is ready to use. You can see that in the Verus ID tab:



The screenshot shows the Verus ID dashboard. The top navigation bar includes tabs for Wallet, Verus Identities, and Mining. The ID Dashboard tab is active. On the left, a box for the Verus coin shows a dropdown menu with the placeholder "Select identity...". In the center, an Identity Overview section shows a "Commit Name" button, a "Recover a Verus ID" button, and a field for "Lücký Jöë@" with a "Used" status and a link to "tracknamecommitment". Below this is a list of identities, with "Lücký Jöë@" selected. The selected identity card shows a profile picture, the name "Lücký Jöë@", a "Can Spend" status, a balance of 69.9998 VRSC, and two buttons: a red one with a minus sign and a blue one with a plus sign. A cursor arrow is visible at the bottom of the identity card.

Congratulations, you are now proud owner of a Verus ID!

Finally, a few notes on referrals.

When you enter a referral ID, purchasing an ID will cost 80 instead of 100 VRSC (not including transaction fees). At the same time, the referral ID will get 20 VRSC.

Furthermore, when someone is using your ID as a referral, you will get 20, and the referral you were using will again get 20 VRSC (but purchase will not get cheaper than 80 VRSC).

This referral propagation will work up to the 3rd generation, i.e., up to the referrer of the referrer of the referrer. Each of them will have 20 VRSC. But at least 20 VRSC will always go back to the miners and stakers of Verus Coin, and will be added to one of the coming blocks. That's also why mining and staking will become more profitable with each ID that was created.

Thanks for reading, and don't hesitate to ask in our discord if you need support! :)

veruscoin.io

Scalability, Decentralization & Security — What Trilemma?

The blockchain trilemma is a classic idea that truly decentralized blockchains need to choose between security or scalability. Verus offers a different perspective and a new approach. Together we will come to the conclusion that the Verus Protocol solves scalability by neither sacrificing decentralization or security, effectively solving the trilemma.



Max Theyse · Follow

Published in Verus Coin · 10 min read · Feb 12, 2024

285

1



A Closer Look at the Trilemma

Before we dive in, let's examine the current perceived definition of the

The image shows a modal window from Medium's sign-up process. It has two main sections: 'Free' and 'Membership'. The 'Free' section lists three benefits: 'Distraction-free reading. No ads.', 'Organize your knowledge with lists and highlights.', and 'Tell your story. Find your audience.' It features a 'Sign up for free' button. The 'Membership' section lists five benefits: 'Access the best member-only stories.', 'Support independent authors.', 'Listen to audio narrations.', 'Read offline.', and 'Join the Partner Program and earn for your writing.' It features a 'Try for \$5/month' button.

The trilemma is currently looked at from a single-chain standpoint. A false premise that does not scale to the world.

There is truth to the fact that it is very difficult to have a decentralized single blockchain that also has a high tps. The larger the number of validators the harder it is to find consensus among them all, which makes them slower, and makes it more difficult to get them to process large numbers of transactions. To get a high tps blockchain you would need to lower the number of validators so they can find consensus more easily, and raise the hardware requirements for running a node so they can process those transactions quicker too. Yet doing so compromises true decentralization. The lower the number of validators, the less decentralized, the less robust. And the higher the hardware requirements, the less accessible it is for people around the world, the less decentralized.

The trilemma is currently looked at from a single-chain standpoint. A false premise that does not scale to the world. We will show you how Verus offers a new and better approach.

Scalability Does NOT Mean High Tps

Transactions per second are nothing more than just a performance indicator — like GHz for computers.

Cryptocurrency protocols often boast about how fast their blockchains are (transactions per second) as an indicator of their scalability. Contrary to popular belief this is a false premise. Transactions per second are nothing more than just a performance indicator. Like how we use GHz (CPU clock speed) to indicate how fast a computer is, or an even better example — how fast servers are.

To bring the promise of crypto to billions of people around the globe let's make a comparison with the Internet. The Internet does not run on one single server. All websites and applications that we use daily are **not** connected to one single server. The Internet has not been scaled to the world by constantly upgrading the one server to a faster one. It's millions of servers that are servicing billions of people around the world. These millions of servers are perfectly connected, and completely interoperable so that using the Internet today is a seamless experience.

Now, with this perspective in mind, let's get back to the current situation in crypto land. Most, if not all, crypto protocols are trying to service the world with their one single, monolithic blockchain. The monolithic blockchain becomes a central hub where all smart contracts and applications revolve around. When they notice that it doesn't scale (the transaction fees start ramping up) the industry comes up with solutions that heavily compromise on decentralization and security (e.g. L2s, semi-centralized protocols, master nodes). This is not the answer.

Instead of acknowledging that the Internet today is a giant, interconnected multi-server world where all applications and websites are seamlessly connected, most of the crypto industry thinks we live in a single server (single blockchain) world.

Verus acknowledged that to realize the promise of cryptocurrency and blockchain, it needed to scale for billions of people around the globe and thus it needed to embrace a multi-server, multi-chain world. And that is exactly what the Verus Protocol does today. Without sacrificing decentralization and security.

Unlimited in Scale

Before we start explaining how Verus is truly decentralized and secure, let's start with how Verus embraces the multi-chain world and delivers unlimited scalability.

Most, if not all, current blockchain protocols argue that to have low transaction fees they must achieve higher transactions per second. They try to scale up — increasing the performance of their monolithic blockchains. This only gets them so far. At some point, they will bottleneck again, and in the process, they compromise on decentralization and security.

Verus offers a new approach, one in which each organization or business can launch its own highly capable, rent-free, no-coding-required blockchain. It's called Verus PBaaS (Public Blockchains as a Service). These PBaaS-chains are fully independent, can be customized to the organization's need, inherit all the same features as the Verus blockchain, and most importantly are fully interconnected and interoperable with the complete network and other bridged protocols (e.g. Ethereum). A single PBaaS-chain can do between 75 and 800 tps depending on the options chosen.

There is no upper limit to deploying PBaaS-chains. Just like there is no upper limit to servers being deployed. When we reach a bandwidth limit we don't go out upgrading our single server, we put another server next to it to spread out the usage. The same thing is here, reaching unlimited scalability.

Of course, this only works when those PBaaS-chains are fully interoperable and interconnected, which they are — in a provable, trustless and decentralized way. There is no centralized entity sitting in between the PBaaS-chains regulating everything.

Cross-chain transactions and conversions, cross-chain bridging of identities (VerusID), tokens and basket currencies, even Ethereum (& ERC20/ERC721/ERC1155) — it's all built into the consensus mechanism, trustlessly bridged over by the decentralized network of miners and stakers only. This is all on mainnet now, for everyone to use.

We can conclude that Verus is scalable to billions of people around the world with the unlimited deployment of PBaaS-chains. But does it compromise on decentralization and thus security? Let's dive into it.

Verus lead developer Mike Tontonghi explains how unlimited scalability is achieved.

Built to be Decentralized

There are a few things at play here that work in harmony to achieve decentralization and security — Verus Proof of Power (the hybrid 50% proof-of-work, 50% proof-of-stake consensus mechanism), merge-mining and the Fee Pool.

Verus had no ICO, had no premine and had and has no developer fee or tax.

Before we jump into the above let's acknowledge some facts that are pivotal to being a true decentralized protocol, to being credibly neutral. Just like Bitcoin before, Verus had no ICO, had no premine and had and has no developer fee or tax. Verus is not a company or a business. Verus is rent-free and all protocol fees go to the block producers. These facts are super important to incentivize decentralization and to be a credibly neutral protocol in the eyes of the world.

A Hybrid Consensus Mechanism

Now let's unpack Verus Proof of Power, the consensus mechanism of the Verus blockchain, and all PBaaS-chains. It's a hybrid of 50% proof-of-work and 50% proof-of-stake, and it's [a provable solution to 51% hash attacks](#). Out of the 1440 daily blocks (on Verus, and when a PBaaS-chain opts for 1-minute block times), half is solved by mining and half by staking.

VerusHash, the proof-of-work mechanism which disincentivizes ASIC & FPGA development, and favors CPUs, mobile phones and ARM-devices for mining. Mining with mobile phones and for example, Orange Pi 5's are the most cost-efficient way to do it. This proves a significant leap towards a more inclusive mining environment. It's accessible to everyone and thus creates a naturally decentralized protocol.



An Orange Pi 5 mining farm by community member DCAL4.

Additionally, to give even more power to miners, they can merge-mine up to 22 PBaaS-chains simultaneously without sacrificing their original hashing power. This dramatically increases efficiency and network security. So, when considering an unlimited number of PBaaS-chains on the network, each miner can choose up to 22 to mine. They can choose for profitability, or choose the ones they want to support.

This approach to mining is environmentally conscious. The protocol's design allows for low-power devices to mine extremely efficiently. It lowers the entry barrier for participation in blockchain validation, especially in low-income countries, yet can also give life to phones that are old or with broken screens. Thus significantly reducing the environmental impact associated with traditional PoW mining.

Then we have the proof-of-stake part of the consensus mechanism. This one is really simple and powerful. Anyone can run a node and start staking. There is no minimum amount of VRSC needed (pretty unique in the world of PoS). Verus solved the “nothing-at-stake”-problem therefore there is no slashing of funds. Funds staking are never locked and running a node can be done with something as cheap as a Raspberry Pi 4. All-in-all proving yet again that Verus is accessible to everyone, creating a naturally decentralized protocol.

We can conclude that Verus is truly a decentralized protocol, naturally emerging from its extremely low-barrier mining and staking. So what does that mean for security?

Protocol Security at Another Level

Seeing as the Verus Protocol is naturally decentralized and provable 51% hash attack resistant, it's extremely secure. PBaaS-chains take these properties with them as they have the same consensus mechanism, and the Verus miners can choose up to 21 of them to merge-mine. It's fair to say that there is no compromise on security given the scalability of the protocol.

Verus Introduces the Fee Pool

Let's first explain the fees the protocol generates for the miners and stakers:

- PBaaS-chain launches: 10,000 VRSC (5,000 for the block producers of Verus, 5,000 for the block producers of the newly launched chain)
- Currency launches (tokens, basket currencies, liquidity pools, ERC-20 mapped currencies): 200 VRSC
- VerusID registrations: 100, 80, 60, 40 or 20 VRSC
- subID registrations: 0.02 VRSC
- DeFi conversion fees: min. 0.0125%, max. 0.025%
- Transaction fees: 0.0001 VRSC

Builders and users pay these protocol fees for their operations and the fees are directed into the Fee Pool. Then, for each new block, 1% of the Fee Pool is added on top of the regular coinbase reward (currently 6 VRSC).

The Fee Pool is introduced as a security measure to keep the protocol stable. We have seen in other blockchain protocols that when a block has large fees, block producers try to "snipe" the block (putting lots of hash onto the network, then pulling it out of the network, or even trying to reorder the blocks for their own gain). In doing so they destabilize the protocol. Verus mitigates this behavior by spreading the extra fees over many blocks.

Smart Contracts Are Not Secure

Most blockchain protocols use the VM-based application model — smart contracts written with Solidity. It's full of smart contract hacks and bugs and has insecure and phishing-prone wallet approval mechanisms. On top of that MEV (maximum extractable value) is rampant, due to the serial processing of transactions on the VM-model (it's easy for a block producer to reorder transactions inside a block for their own gain). We can honestly say that the VM-model is not secure.

The Verus approach is different as it follows the fundamental systems design principle — the most important security layers should be located in the protocol itself, and not coded on top via L2 (smart contracts). All operations on Verus and PBaaS-chains are directly connected to consensus, secured by the miners and stakers of the protocol.

Now you might ask yourself "But can we build dApps that are as capable as building them with the VM-model?". The answer is yes. Developing dApps with Verus is much more straightforward than building to a VM-based application model and results in inherently more capable, secure and scalable solutions.

Does Verus Solve the Blockchain Trilemma?

A fair question. And hopefully we answered with enough detail. The answer is "yes!".

Let's summarize. Verus does not compromise on decentralization and security to reach unlimited scalability. Unlimited scalability is reached by scaling out, not scaling up. It is the equivalent not of upgrading your server but of putting an extra server next to it to spread the bandwidth around.

Decentralization is achieved by making mining and staking accessible for everyone through extremely low hardware requirements. This naturally creates a large number of miners and nodes (stakers), making the network robust against attacks of all sorts.

High security is not only reached through the decentralized nature of the network but also because of the Fee Pool and the fact that all operations are directly connected to consensus and thus the miners and stakers.

Now is the Time to Build with Verus

Without having to rely on smart contracts, builders can develop dApps that are even more powerful, secure and scalable. Builders can use the no-

coding-required API commands, together with VerusID and its VDXF (Verus Data Exchange Format) as a controlled public storage system with multiple levels of nesting to create dApps of any kind.

The foundation has been laid for a true Internet of Value. Verus urges all builders to [join the community on Discord](#) and to get familiar with the cutting-edge blockchain technology. Go build your project with Verus without having to learn a new programming language. It's here, it's ready, all live on mainnet.

Find Verus at Consensus 2024, May 29–31, Austin, USA. The community is happy to inform you on the dApp building opportunities! Read more here: [Consensus 2024 – Verus Showcases Fully Completed PBaaS Blockchain Technology](#)

Try Yourself!

Look up the [complete command list here](#). Go to [docs.verus.io](#) to get guidance on API commands (e.g. [launching currencies, tokens & liquidity pools](#)).

Join the community. Learn about the protocol. Use Verus & build.

 [Join the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Development Blockchain Cryptocurrency Cryptocurrency News
Blockchain Technology

 285  1

Written by Max Theyse

149 Followers · Editor for Verus Coin

More from Max Theyse and Verus Coin

 Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

 196 

 Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

 304 

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346  1 

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259  

[See all from Max Theyse](#)

[See all from Verus Coin](#)

Recommended from Medium

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538  13 

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196  

Lists

Modern Marketing

103 stories · 523 saves

Generative AI Recommended Reading

52 stories · 894 saves

My Kind Of Medium (All-Time Faves)

71 stories · 261 saves

N. R. Crowningshield in Kaspa Currency

Kaspa: Accelerating Beyond the Blockchain

From a Concept to a Cutting-Edge Deployment

6 min read · Mar 18, 2024

107  

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2k  29 

 Riti Thummalapenta

Is Quantum Computing the End of Blockchain?

An overview of the threats posed by quantum computing, and how we're tackling them.

4 min read · Oct 23, 2023

 14 

 ⁺

 KryptoPunk

Bittensor wallet scanner: TensorScan AI (100x faster!)

Discover the revolutionary Bittensor Wallet Scanner: Tensor Scan AI, the ultimate gem i...

3 min read · Mar 22, 2024

 122 

 ⁺

[See more recommendations](#)

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)

Unleash the Power of Verus — Technological Showcase of Centrally-Issued Token Fully Backed by Gold

In this article, we showcase how to launch a centrally controlled token with minting and burning capabilities, fully backed by gold. We use the innovative Verus blockchain technology with its versatile currency launches, which can be used both for centralized and decentralized projects. And we don't even need to code anything!



Max Theyse · Follow

Published in Verus Coin · 5 min read · Feb 19, 2023

354



Medium

Sign up to discover human stories that deepen your understanding of the world.

or 20, depending on different use cases.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

Sign up for free

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

Try for \$5/month

Defining a Currency

Defining a currency on the Verus network is simple, just two commands. When the currency is successfully launched it can flow around within the protocol, and outside of the protocol ([learn how to export a currency as an ERC-20 to Ethereum](#)). Let's get started.

VerusID Namespace

verus.id @

Before defining the currency, we already registered the VerusID `vgg@`. This cost us 100 VRSCTEST (or 80 when using a referral), which were distributed to the block producers. These costs can drastically vary depending on which PBaaS-blockchain we create a VerusID.

VerusID is the namespace for currencies and PBaaS-blockchains, among many other things. [Learn more about VerusID](#) 

We also said to have a 2 of 2 signature setup. This means the identity is controlled by two primary addresses, with always two signatures needed to do any action (like launching the currency, minting, burning and sending).

Defining the currency

Defining a currency costs 200 VRSCTEST. These costs can change depending on which blockchain you want to define a currency on. When launching PBaaS-chains you can define your own costs in the chains' native currency. **Good to know:** these costs are going to the miners and stakers of the blockchain.

With a few simple commands, we will:

- Define the currency
- Have it signed by a second signature
- Launch the currency

Before starting we must fund `VGG@` with 200+ VRSCTEST to launch the currency. The chain definition is as follows:

```
./verus -chain=VRSCTEST definecurrency '{  
  "name": "VGG",  
  "options": 32,  
  "proofprotocol": 2  
}'
```

It's a simple chain definition, with only `"options": 32` (= the currency is just a token), and `"proofprotocol": 2` (= the controller of the VerusID where the currency is launched from can mint and burn tokens at will).

These are some options and specifications that are possible when launching a currency: as a simple token, as fractional reserve currency or multi-reserve basket, start & end block, min. & max. preconversions, initial contributions, pre-launch discount & carveout, initial supply and much more.

★ To learn more about the many different launch options when defining a currency on Verus, [go here](#)  We encourage anyone to start testing the protocol to get ahead of the game. [The community is happy to help on Discord](#) 

Approving and launching the currency

Now, the second signer needs to come in and sign off on the currency definition, and launch the currency.

The command from before gives out a HEX. The second signer needs to sign this HEX as follows:

```
./verus -chain=VRSCTEST signrawtransaction "hex"
```

This gives out another HEX. We use it to submit the transaction to the chain, which in turn launches the currency.

```
./verus -chain=VRSCTEST sendrawtransaction "hex"
```

After doing the command above, it takes around 15 blocks before the currency is officially launched on the network.

Next step is the minting and burning of tokens, according to how much gold the company has in custody.

Minting and burning

Say a customer put 10,000 ounces of gold into the custody of the company. Now let's make sure the customer gets that amount of VGG into their wallet.

We mint 10,000 VGG with this simple command:

```
./verus -chain=VRSCTEST sendcurrency "VGG@" '[{"address": "Customer1@", "currency": "VGG", "mintnew": 1, "amount": 10000}]'
```

VGG@ is the change-address, Customer1@ is the address where the 10,000 VGG are minted to.

The second signer needs to do `signrawtransaction` and `sendrawtransaction`, as we have done before, to actually mint the tokens into Customer1@.

Now, if someone wants to redeem their VGG for gold again, we need to be able to burn that amount from the supply of VGG. We burn with the following command:

```
./verus -chain=VRSCTEST sendcurrency "VGG@" '[{"address": "VGG@", "currency": "VGG", "amount": 5000, "burn": true}]'
```

To make it happen, the second signer needs to sign off on it again.

If we want to check to see the supply of the currency, we can do the following command:

```
./verus -chain=VRSCTEST getcurrency "vgg"
```

Use VGG in Many Ways

This article showcased a simple example of how a business can function using the Verus protocol. What we can do with the network goes far beyond what is possible using current blockchain protocols. And all of that without any programming needed!

We can also easily export the VGG token to Ethereum as an ERC-20. It can also become one of the reserves in a multi-reserve liquidity basket AMM (automated market maker) as part of Verus DeFi.

Anybody can define a multi-reserve currency which includes VGG as a reserve with weighted percentage (10-90%). The reserves grow as they retain a small fee whenever assets are traded in/out/through.

A multi-reserve currency itself can also be used as one of the reserves in another multi-reserve liquidity basket AMM token Verus.

Stay tuned for more examples, or start reading [docs.verus.io](#), or [join the Discord to learn](#).

Credits for the article go to community member 'ejuliano'.

 [Explore with the community on Discord](#)

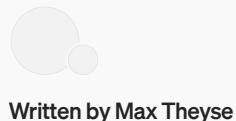
[Follow on Twitter](#)

[Go to verus.io](#)

Blockchain Development Blockchain Technology Cryptocurrency Innovation
Cryptocurrency News

 354 



149 Followers · Editor for Verus Coin

[Follow](#) 

More from Max Theyse and Verus Coin

 Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

 196 

 Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

 304 

 Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

 Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

346 1

259

+

See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196

Q

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538

Q 13

+

Lists

Tech & Tools

16 stories · 193 saves

Generative AI Recommended Reading

52 stories · 894 saves

Modern Marketing

103 stories · 523 saves

Business

41 stories · 88 saves

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

143

Q

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4

Q

+

Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781

Q 4

Yash Agarwal

State of Solana DePIN 2024

Dive deep into Decentralised Physical Infrastructure (DePIN) and why Solana is the...

24 min read · Feb 16, 2024

150

Q

+

([See more recommendations](#))

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)

Get Ahead of the Game with Verus' Groundbreaking 1:1 Decentralized Currency Mapping for ERC-20 Tokens

In this article, we show you how to map and bridge an Ethereum ERC-20 token to the Verus testnet blockchain as a currency one-to-one. We use the groundbreaking non-custodial and provable Verus-Ethereum bridge, and we don't even need to code anything!



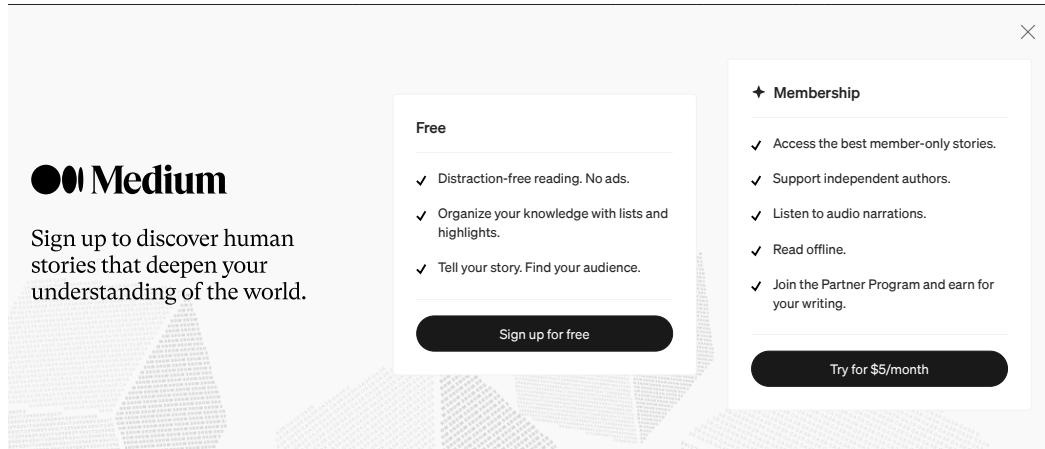
Max Theyse · Follow

Published in Verus Coin · 4 min read · Feb 9, 2023

176



Note that this walkthrough is for the Verus Testnet blockchain, which simulates



The image shows a modal window from Medium. It has two main sections: 'Free' on the left and 'Membership' on the right. The 'Free' section lists three benefits: 'Distraction-free reading. No ads.', 'Organize your knowledge with lists and highlights.', and 'Tell your story. Find your audience.' Below this is a 'Sign up for free' button. The 'Membership' section lists five benefits: 'Access the best member-only stories.', 'Support independent authors.', 'Listen to audio narrations.', 'Read offline.', and 'Join the Partner Program and earn for your writing.' Below this is a 'Try for \$5/month' button. The background of the modal features a faint, abstract geometric pattern.

- convert currencies through the bridge converter currency. Use on Ethereum as well as on Verus. [Learn more on bridge converter currencies](#)

Defining a Currency

Defining a currency on the Verus network is simple, just two commands are all you need. When the currency is successfully launched it can flow around within the protocol, and outside of the protocol (like an ERC-20, in our example). Let's get started.

VerusID Namespace

Before defining the currency, we already registered the VerusID `MaxUSDC@`. This cost us 100 VRSCTEST (or 80 when using a referral), which were distributed to the block producers. These costs can drastically vary depending on which PBaaS-chain you create a VerusID.

VerusID is the **namespace** for currencies and PBaaS-blockchains, among many other things. [Learn more about VerusID](#) ☕

Defining the mapped currency

Defining a currency costs 200 VRSCTEST. These costs can change depending on which blockchain you want to define a currency. When launching PBaaS-chains you can define your own costs in the chains' native currency. **Good to know:** these costs are going to the miners and stakers of the blockchain.

With a few simple commands we will:

- Define a currency mapped to the USDC Ethereum contract with a registered namespace (VerusID) on the Verus Testnet blockchain
- Export the currency to Ethereum through [the non-custodial, provable bridge](#) ☕

Before we can define the currency which is mapped to USDC on Ethereum, we must fund `MaxUSDC@` with enough VRSCTEST to launch the currency. And after that with enough vETH or VRSCTEST to pay for the gas fees that export the currency to Ethereum.

200+ VRSCTEST to launch the currency, and around 0.05 vETH or equivalent worth of VRSCTEST to export the currency to Ethereum.

Fund your VerusID with vETH in two ways:

1. Send ETH to Verus (into vETH) with [the Verus-Ethereum bridge](#) ☕
2. Convert VRSCTEST or any other currency into vETH using Verus' protocol level DeFi [more info on how to convert](#) ☕

The command that defines our currency `MaxUSDC`, which is mapped to the ERC-20 USDC (Goerli Testnet) [contract address on Etherscan](#) ☕, is as follows:

```
./verus -chain=VRSCTEST definecurrency '{
  "name": "MaxUSDC",
  "options": 32,
  "systemid": "veth",
  "parent": "vrsctest",
  "launchsystemid": "vrsctest",
  "nativecurrencyid": {
    "type": 9,
    "address": "0x98339D8C260052B7ad81c28c16C0b98420f2B46a"
  },
  "initialsupply": 0,
  "proofprotocol": 3
}'
```

★ To learn more about the many different launch options when defining a currency on Verus, [go here](#) ☕ We encourage anyone to start testing the protocol to get ahead of the game. [The community is happy to help on Discord](#) ☕

When we do the command above we get returned a HEX. Only after we make a transaction with the HEX, will the currency launch be broadcasted to

the network. See below:

```
./verus -chain=VRSCTEST sendrawtransaction "hex"
```

Now you can see the currency on the Verus network. We just need to make sure it's on Ethereum as well.

Exporting the mapped currency

Now, let's export the currency from Verus to Ethereum as an ERC-20 over the non-custodial bridge. We do this with the following command:

```
./verus chain=VRSCTEST sendcurrency "MaxUSDC@" '[{
  "address": "0xAD84C9EE28FB4c3b9C6ef1B86D1ED00A82DB84E9",
  "currency": "MaxUSDC",
  "amount": 0,
  "exportto": "veth",
  "exportcurrency": true,
  "feecurrency": "veth"
}]'
```

Note: the first `MaxUSDC@` field is the change address for fees being returned to you, and the `address` is the Verus-Ethereum non-custodial smart contract bridge address. You can also omit `feecurrency` if you would rather pay the export fees in VRSCTEST.

After the bridge has been notarized to the blockheight where you have exported the currency, you can choose it from the token dropdown [on the bridge website](#) .

ERC-20s on Verus

After sending from Ethereum to Verus, the token is free to move through the Internet of Value on the Verus network, across PBaaS-blockchains, and also back & forth with Ethereum.

As easily as the token was first defined and launched, it can now become one of the reserves in a multi-reserve liquidity basket AMM as part of Verus DeFi.

Anybody can define a multi-reserve currency which includes `MaxUSDC` as a reserve with weighted percentage (10–90%). The reserves grow as they retain a small fee whenever assets are traded in/out/through.

A multi-reserve currency itself can also be used as one of the reserves in another multi-reserve liquidity basket AMM token on Verus.

 Stay tuned for more on this later, or start reading [docs.verus.io](#), or [join the Discord to learn](#).

...

Credits for the article go to community member 'ejuliano'.

...

 [Explore with the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)



Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow

+

More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304

+

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346

1

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259

+

See all from Max Theyse

See all from Verus Coin

Recommended from Medium

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

143

4

+

Lists

data science and AI
40 stories · 122 saves

Generative AI Recommended Reading
52 stories · 894 saves

Modern Marketing
103 stories · 523 saves

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2k

29

Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Deskt...

5 min read · Oct 12, 2023

+

146

+

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538

13

Yash Agarwal

State of Solana DePIN 2024

Dive deep into Decentralised Physical Infrastructure (DePIN) and why Solana is the...

24 min read · Feb 16, 2024

+

150

+

See more recommendations

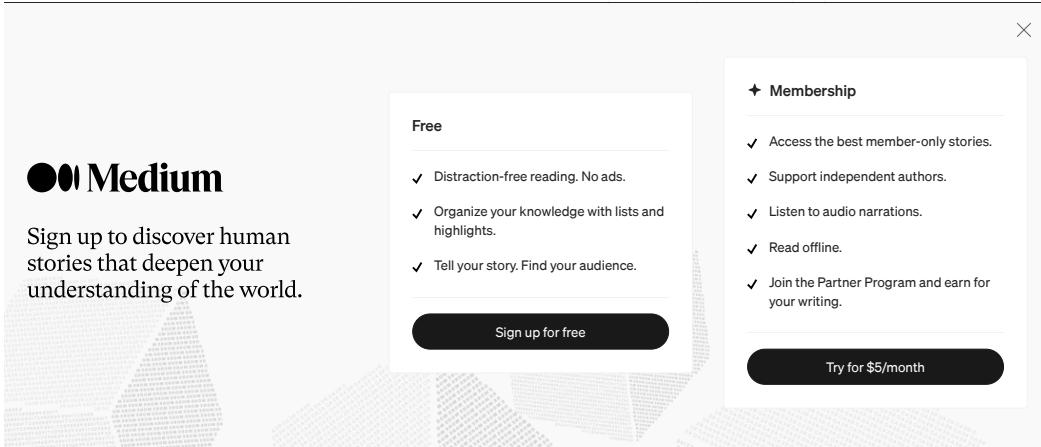
Get Ahead of the Game with Verus' Groundbreaking 1:1 Decentralized Currency Mapping for ERC-20 Tokens

In this article, we show you how to map and bridge an Ethereum ERC-20 token to the Verus testnet blockchain as a currency one-to-one. We use the groundbreaking non-custodial and provable Verus-Ethereum bridge, and we don't even need to code anything!

 Max Theyse · Follow
Published in Verus Coin · 4 min read · Feb 9, 2023

A scenic landscape featuring a bridge over a river, surrounded by lush green trees and misty mountains, symbolizing a non-custodial and provable Ethereum bridge.

 Note that this walkthrough is for the Verus Testnet blockchain, which simulates



- convert currencies through the bridge converter currency. Use on Ethereum as well as on Verus. [Learn more on bridge converter currencies](#) 

Defining a Currency

Defining a currency on the Verus network is simple, just two commands are all you need. When the currency is successfully launched it can flow around within the protocol, and outside of the protocol (like an ERC-20, in our example). Let's get started.

VerusID Namespace

Before defining the currency, we already registered the VerusID `MaxUSDC@`. This cost us 100 VRSCTEST (or 80 when using a referral), which were distributed to the block producers. These costs can drastically vary depending on which PBaaS-chain you create a VerusID.

VerusID is the **namespace** for currencies and PBaaS-blockchains, among many other things. [Learn more about VerusID](#) ☕

Defining the mapped currency

Defining a currency costs 200 VRSCTEST. These costs can change depending on which blockchain you want to define a currency. When launching PBaaS-chains you can define your own costs in the chains' native currency. **Good to know:** these costs are going to the miners and stakers of the blockchain.

With a few simple commands we will:

- Define a currency mapped to the USDC Ethereum contract with a registered namespace (VerusID) on the Verus Testnet blockchain
- Export the currency to Ethereum through [the non-custodial, provable bridge](#) ☕

Before we can define the currency which is mapped to USDC on Ethereum, we must fund `MaxUSDC@` with enough VRSCTEST to launch the currency. And after that with enough vETH or VRSCTEST to pay for the gas fees that export the currency to Ethereum.

200+ VRSCTEST to launch the currency, and around 0.05 vETH or equivalent worth of VRSCTEST to export the currency to Ethereum.

Fund your VerusID with vETH in two ways:

1. Send ETH to Verus (into vETH) with [the Verus-Ethereum bridge](#) ☕
2. Convert VRSCTEST or any other currency into vETH using Verus' protocol level DeFi [more info on how to convert](#) ☕

The command that defines our currency `MaxUSDC`, which is mapped to the ERC-20 USDC (Goerli Testnet) [contract address on Etherscan](#) ☕, is as follows:

```
./verus -chain=VRSCTEST definecurrency '{
  "name": "MaxUSDC",
  "options": 32,
  "systemid": "veth",
  "parent": "vrsctest",
  "launchsystemid": "vrsctest",
  "nativecurrencyid": {
    "type": 9,
    "address": "0x98339D8C260052B7ad81c28c16C0b98420f2B46a"
  },
  "initialsupply": 0,
  "proofprotocol": 3
}'
```

★ To learn more about the many different launch options when defining a currency on Verus, [go here](#) ☕ We encourage anyone to start testing the protocol to get ahead of the game. [The community is happy to help on Discord](#) ☕

When we do the command above we get returned a HEX. Only after we make a transaction with the HEX, will the currency launch be broadcasted to

the network. See below:

```
./verus -chain=VRSCTEST sendrawtransaction "hex"
```

Now you can see the currency on the Verus network. We just need to make sure it's on Ethereum as well.

Exporting the mapped currency

Now, let's export the currency from Verus to Ethereum as an ERC-20 over the non-custodial bridge. We do this with the following command:

```
./verus chain=VRSCTEST sendcurrency "MaxUSDC@" '[{
  "address": "0xAD84C9EE28FB4c3b9C6ef1B86D1ED00A82DB84E9",
  "currency": "MaxUSDC",
  "amount": 0,
  "exportto": "veth",
  "exportcurrency": true,
  "feecurrency": "veth"
}]'
```

Note: the first `MaxUSDC@` field is the change address for fees being returned to you, and the `address` is the Verus-Ethereum non-custodial smart contract bridge address. You can also omit `feecurrency` if you would rather pay the export fees in VRSCTEST.

After the bridge has been notarized to the blockheight where you have exported the currency, you can choose it from the token dropdown [on the bridge website](#) .

ERC-20s on Verus

After sending from Ethereum to Verus, the token is free to move through the Internet of Value on the Verus network, across PBaaS-blockchains, and also back & forth with Ethereum.

As easily as the token was first defined and launched, it can now become one of the reserves in a multi-reserve liquidity basket AMM as part of Verus DeFi.

Anybody can define a multi-reserve currency which includes `MaxUSDC` as a reserve with weighted percentage (10–90%). The reserves grow as they retain a small fee whenever assets are traded in/out/through.

A multi-reserve currency itself can also be used as one of the reserves in another multi-reserve liquidity basket AMM token on Verus.

 Stay tuned for more on this later, or start reading [docs.verus.io](#), or [join the Discord to learn](#).

...

Credits for the article go to community member 'ejuliano'.

...

 [Explore with the community on Discord](#)

[Follow on Twitter](#)

[Go to verus.io](#)



Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow

+

More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196

Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304

+

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346

1

Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259

+

See all from Max Theyse

See all from Verus Coin

Recommended from Medium

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

143

4

+

Lists

data science and AI
40 stories · 122 saves

Generative AI Recommended Reading
52 stories · 894 saves

Modern Marketing
103 stories · 523 saves

Passive Crypto Mining

My Top 15 Passive Crypto Miners of 2024

I've spent 2 weeks compiling the most profitable Crypto Miners for 2024. You can...

11 min read · Jan 12, 2024

2k

29

Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Deskt...

5 min read · Oct 12, 2023

+

146

+

Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538

13

Yash Agarwal

State of Solana DePIN 2024

Dive deep into Decentralised Physical Infrastructure (DePIN) and why Solana is the...

24 min read · Feb 16, 2024

+

150

+

See more recommendations

Jul-Oct 2022 — Verus Monthly Recap

The latest Verus updates on development, community, ecosystem and more. [Subscribe to the newsletter here.](#)



Max Theyse · Follow

Published in Verus Coin · 5 min read · Nov 17, 2022

1



You can receive the exact same content through a newsletter. [Subscribe here.](#) I'll keep things short and to the point.

Verus Desktop v0.9.5 Update

⌚ Update your Verus Desktop wallet before 19 November 11 PM UTC. This

Medium

Sign up to discover human stories that deepen your understanding of the world.

[Download Verus Desktop v0.9.5](#)

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

[Sign up for free](#)

Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

[Try for \\$5/month](#)



Verus PBaaS & DeFi Updates

In anticipation of the Verus PBaaS mainnet upgrade — delivering no-programming needed blockchain and currency launches, MEV-free DeFi, non-custodial Ethereum bridging and so much more — community

developers introduce a protocol improvement designed to make the Verus anti-MEV DeFi protocol even more resistant than what we believe was already the leading algorithm to multi-block attempts at MEV.

The protocol improvement makes the following adjustments:

- Conversions and cross-chain transfers still roll-up into exports that are collected from one or more blocks of transfer transactions, but to determine when a block is the last of the prior export or first in the next requires a random bit pulled from later blocks.
- The last miner or staker to complete a block of an export retroactively receives 10% of all fees from processing that export, including all conversions, once the rollup is processed as an import. This reward used to go to the miner or staker who processes the export, which is no longer the case.
- Import transactions can now include up to $n/2$ arbitrage transactions, where n is the number of reserves in a liquidity basket currency.

These changes together serve to disincentivize any miner or staker from attempting to exclude a large set of transactions from a block and front run all of them, as they will have an even chance of having all of those excluded transactions put back in with their transaction, turning potential gain into potential loss. They would be better off saving their working capital to arbitrage imports by joining transactions, bringing them close to market, and earning money without risk of loss. In addition, if they do end up putting more value of transactions in a block that caps the export rollup, they will reliably earn meaningful rewards without risk of loss in that way as well.

Bug Bounties

While we don't expect it to be easy to find bugs in the protocol, The Verus Coin Foundation is offering a bounty of at least 500 VRSC (possibly more for security related reports) for the first 10 people who are first to report any actual protocol bug before mainnet release, meaning failure of a properly executed command or API on a functioning VRSCTEST or PBaaS chain to function as intended.

If anyone is first to find and describe the exploit for an exploitable security hole in any part of the protocol, the bounty will be a minimum of 1,000 VRSC, and for a serious potential security issue, 10,000 VRSC.

 [See PBaaS-Testnet Release Notes](#)

 [Join Discord and report bounties](#)

...

 verus.id @

Make applications more secure with the Verus Authentication Protocol.



Introducing VerusID to Verus Mobile

This version of Verus Mobile takes a major leap forward, enabling support for VerusID, including sending and receiving funds and use of your VerusID as a password-free, secure and self-sovereign web or mobile app sign-in, made possible with the VerusID authentication protocol.

Verus Mobile v0.3.0-24 supports the version 1.0 of the VerusID authentication protocol, which prepares for the PBaaS upgrade and supports application provisioning of IDs, an important way applications can onboard users, give them a verified self-sovereign ID and funds address, enable sending and receiving payments, and make applications more secure by design with an easy, password-free sign-in.

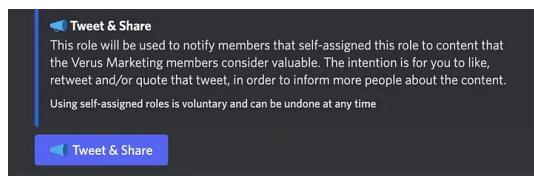
In addition to its new support for VerusID and the Verus Authentication Protocol, this version of Verus Mobile continues to include Wyre as a fiat on-ramp and maintains support for Bitcoin, Ethereum, and a variety of popular coins and tokens.

If you are an app developer who wants to benefit from and help pioneer a better, more user-centric, inherently commerce-enabled, verifiable application model for a true Internet of Value, this is the release that enables you! If you've got mobile or web applications that you are seriously building and would like to integrate VerusID authentication, contact one of the community developers for more details.

 [Download Verus Mobile with VerusID](#)

The Valuverse

[Valu](#), a company using Verus technology, gave a sneak preview of their upcoming Valuverse. A virtual world where you are at the centre of your value. Check out the awesome videos below.



Join Twitter Army

We need your help! The Verus community wants to amplify tweets that are important. We make it easy for you: get notified when there are high-quality tweets that need more likes and retweets.

[Join the discord](#), assign yourself the role of 'Tweet & Share' (in the #welcome channel) and get retweeting!

Block Reward Halving



Verus Network Halving

The Verus network block reward halving is almost around the corner. At block 2,329,920 (~Saturday 17 Dec 2022), the block reward will go from 12 to 6 VRSC.

[!\[\]\(2f9c7457283e9de4726329d0aac22594_img.jpg\) Countdown website](#)

...

New Marketing Videos

...

...

[Subscribe to get the \(bi-\)monthly recap to your mailbox](#)

[Blockchain](#) [Cryptocurrency News](#) [Digital Identity](#) [Blockchain Technology](#)

[Crypto](#)

 1 



Written by Max Theyse

149 Followers · Editor for Verus Coin

Follow



More from Max Theyse and Verus Coin

Max Theyse in Verus Coin

Introducing Pure—The Currency 100% Backed by Verus & Bitcoin

Pure is a decentralized currency fully backed by Verus (VRSC) & Bitcoin (tBTC).

8 min read · Mar 16, 2024

196



Max Theyse in Verus Coin

Verus: Profit Generating Protocol for Miners and Stakers

Future and present Verus miners: take notice. Be ready to maximize profit with your...

4 min read · May 12, 2021

304



1+

Oliver in Verus Coin

How to Start CPU Mining Verus Coin VRSC from Your Computer i...

A Dummies Step by Step Guide to Pool Mining Verus Coin with a CPU!

8 min read · Jan 10, 2019

346



Max Theyse

How to preconvert into Bridge.vARRR

Participate in the first ever Verus PBaaS-chain launch—vARRR. Learn how to...

4 min read · Mar 20, 2024

259



1+

See all from Max Theyse

See all from Verus Coin

Recommended from Medium

Max Theyse in Verus Coin

How-to Participate in the Verus-Ethereum Bridge Launch

Instructions on how to use the Verus-Ethereum Bridge website and Verus Desktop...

5 min read · Oct 12, 2023

146



Albert Peter in Cryptocurrency Scripts

Top 5 Crypto Gems Predicted to Reach 100x-1000x Gains in 2024

Discover the top 5 crypto gems poised for 100x-1000x gains in 2024. Don't miss out on...

6 min read · Mar 14, 2024

538



13

Lists

My Kind Of Medium (All-Time Faves)

71 stories · 261 saves

Modern Marketing

103 stories · 523 saves

Edge Ruler in Coinmonks

Top 3 Cryptos to Hold until 2025: Like Buying BITCOIN at 10\$

In today's crypto landscape, amidst a prolonged bear market, investors find...

4 min read · Mar 16, 2024

781 4

Ordify

The Tier System

The Ordify Launchpad has only 6 Tiers and operates on a pool weight-based method....

4 min read · Dec 30, 2023

362 3

+

PYRIN-ONE in pyrin

PYRIN AMA (Ask me Anything)

On February 1st, 2024, the PYRIN team held its first AMA (Ask Me Anything). The team...

13 min read · Feb 2, 2024

143 4

Perzibal

How to mine TAI using TonAi on telegram ? Full guide.

Start Mining on Telegram bot for Free

2 min read · Feb 11, 2024

4 4

+

See more recommendations

[Help](#) [Status](#) [About](#) [Careers](#) [Blog](#) [Privacy](#) [Terms](#) [Text to speech](#) [Teams](#)

Transfer Destination

The Transfer Destination construct is a universal component used within the Verus blockchain network, designed to define destinations within blockchain operations comprehensively. This construct is crucial for specifying the end points in a variety of blockchain transactions, supporting a wide array of destination types to accommodate diverse blockchain functionalities and cross-chain interactions.

Core Concepts

Transfer Destination encapsulates key information necessary for blockchain transactions, including the destination type, destination-specific bytes, gateway information, and associated fees. It supports a flexible architecture for defining complex transaction paths, enhancing the blockchain's capability to handle sophisticated and multi-layered operations.

Destination Types

The Transfer Destination construct supports various destination types, each serving specific purposes:

- **DEST_INVALID (0):** Represents an invalid or unspecified destination type, used as a default or error state.
- **DEST_PK (1):** Indicates a public key destination, typically used for transactions directly to a public key.
- **DEST_PKH (2):** Stands for a public key hash destination, common in many blockchain platforms for sending transactions to a hashed version of a public key (e.g. an r-address).
- **DEST_SH (3):** Represents a script hash destination, used for transactions that should be processed by a specific script, enabling smart contracts or complex spending conditions.
- **DEST_ID (4):** Identifies a [VerusID](#) destination.
- **DEST_FULLID (5):**
- **DEST_REGISTERCURRENCY (6):**
- **DEST_QUANTUM (7):** Used for quantum-resistant addresses.
- **DEST_NESTEDTRANSFER (8):**
- **DEST_ETH (9):** Specifies an Ethereum account as the destination, facilitating cross-chain transactions with Ethereum.
- **DEST_ETHNFT (10):** Indicates a destination for an Ethereum-compatible Non-Fungible Token (NFT), enabling the mapping of NFTs across different blockchain systems.
- **DEST_RAW (11):** Represents a raw data destination, allowing for arbitrary data to be included as a destination, offering maximum flexibility.

Flags

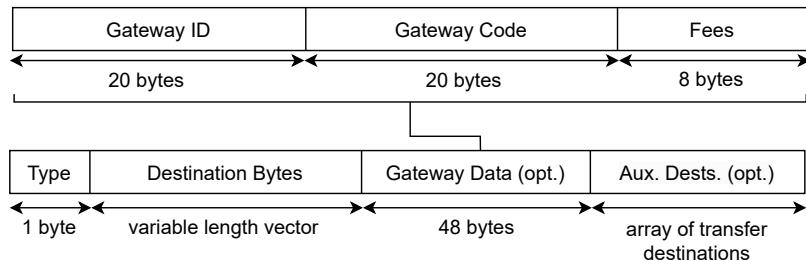
Transfer Destination also supports the use of flags to indicate additional attributes of a destination:

- **FLAG_DEST_AUX (64):** Indicates the presence of auxiliary destinations, allowing for the specification of additional destinations within a single Transfer Destination construct.
- **FLAG_DEST_GATEWAY (128):** Specifies that the destination is associated with a gateway, relevant for cross-chain transactions or interactions with external systems.

Serialization and Deserialization Process

The core functionality of Transfer Destination revolves around the ability to serialize and deserialize destination information. This process ensures that destination data can be efficiently transmitted across networks or stored, maintaining integrity and compatibility across different implementations.

Key Components in Serialization



- **Type:** A numerical value indicating the destination type, serialized directly as part of the destination data. Includes flags.
- **Destination Bytes:** The specific bytes associated with the destination, which may represent an address, identifier, or other relevant data, depending on the destination type.
- **Gateway ID & Gateway Code:** Optional components used for gateway destinations, serialized when present to include external system identifiers.
- **Fees:** Associated fees, if applicable, serialized to ensure the correct processing of transactions, particularly for complex or cross-chain operations.
- **Auxiliary Destinations:** An optional list of additional Transfer Destination constructs, serialized to support nested or multi-part transactions.

Considerations

The serialization and deserialization mechanisms must adhere to a standardized format to ensure interoperability. Implementations in different programming languages should focus on accurately reflecting the structure and logic outlined, ensuring that Transfer Destination constructs are universally compatible, regardless of the underlying platform or language used.

Application and Significance

Transfer Destination is a foundational component for blockchain developers, enabling the definition of flexible, interoperable, and sophisticated transaction

pathways. Its design facilitates a wide range of blockchain operations, from simple transfers to complex cross-chain and multi-step transactions, making it an essential tool in the development of decentralized applications and systems.

Implementation Examples

The Transfer Destination construct is implemented in a number of codebases across the Verus ecosystem:

- [The VerusCoin core GitHub repository \(as CTransferDestination\)](#)
- [The verus-typescript-primitives utility library \(as TransferDestination\)](#)

← [Introduction](#)

Introduction

Addresses & Transactions

Introduction

Public addresses

Private addresses

Public-to-public

Public-to-private

Private-to-public

Private-to-private

Transfer Destination

In Verus, there are two kinds of addresses: **public** and **private**. This doc guides you through understanding these address types and how they're involved in transactions.

Understanding the distinctions between public and private addresses in the Verus ecosystem is crucial for making informed decisions about transaction privacy.

Public addresses

You can use the following public address types to send and receive funds.

Address type	Details
R-address	A R-address is a public cryptocurrency address derived from a private key. The private key is essential for accessing and controlling the R-address.
VerusID@	A VerusID is a public friendly-name cryptocurrency address. It is controlled by a primary address, which is a R-address.
i-address	An i-address is a public cryptocurrency address derived from a VerusID.

Public addresses and their transactions are visible on blockchain explorers because they're recorded on the public ledger, unlike private z-addresses.

Private addresses

You can use the following private address type for confidential exchange of funds and data.

Address type	Details
z-address	A z-address is a private cryptocurrency address derived from a private key. The private key is essential for accessing and controlling the z-address. Only the person who has the private key can spend and see its balance. Or provide visibility in the balance through a viewing key .

Balances and transactions associated with private addresses are confidential. They do not appear on the blockchain or any explorer.

i A VerusID can contain a pointer to a z-address. You can then send coins to `VerusID@:private`.

Only native currencies

Z-addresses can only contain the native blockchain currency. Simple token currencies or basket currencies can **not** be held in a z-address.

Transaction scenarios

Public-to-public

`VerusID@` , `R-address` , or `i-address` `VerusID@` , `R-address` , or `i-`

address

- Sender's address and the amount sent are visible
- Recipient's address(es) and received amount(s) are visible

Public-to-private

VerusID@ , R-address , or i-address → z-address or VerusID@:private

- Sender's address and the amount sent are visible
- Recipient's address(es) and received amount(s) are **not** visible

Note: Correlating transactions by time and amounts between public and private addresses is still possible, potentially linking two public addresses based on transaction patterns.

Private-to-public

z-address → VerusID@ , R-address , or i-address

- Sender's address and the amount sent are **not** visible
- Recipient's address(es) and received amount(s) are visible

Note: Like public-to-private transactions, there remains a possibility of correlating transactions based on timing and value, potentially linking two public addresses.

Private-to-private

z-address → z-address or VerusID@:private

- Sender's address and the amount sent are **not** visible
- Recipient's address(es) and received amount(s) are **not** visible

Transfer Destination →

Defining a PBaaS-blockchain

Launch Blockchains

- Introduction
- [Defining a PBaaS-blockchain](#)
- VerusID Namespace
- Examples of blockchain launches

Information here is not complete. Need help setting up a blockchain launch? 😊

Go to the Verus Discord [#pbaas-development](#) channel. The community is happy to assist! ↗

There are many options to choose from when defining your blockchain. Combine them in the `options` parameter for different use cases.

#	Options	Details
1	OPTION_FRACTIONAL	Allows reserve conversion using base calculations when set
2	OPTION_ID_ISSUANCE	Clear is permissionless, if set, IDs may only be created by controlling ID
4	OPTION_ID_STAKING	All IDs on chain stake equally, rather than value-based staking
8	OPTION_ID_REFERRALS	If set, this chain supports referrals
16	OPTION_ID_REFERRALREQUIRED	If set, this chain requires referrals
32	OPTION_TOKEN	If set, this is a token, not a native currency
64	OPTION_SINGLECURRENCY	For PBaaS chains or gateways to potentially restrict to single currency
128	OPTION_GATEWAY	If set, this routes external currencies
256	OPTION_PBAAS	This is a PBaaS chain definition
512	OPTION_GATEWAY_CONVERTER	This means that for a specific PBaaS gateway, this is the default converter and will publish prices
1024	OPTION_GATEWAY_NAMECONTROLLER	When not set on a gateway, top level ID and currency registration happen on launch chain

#	Options	Details
2048	OPTION_NFT_TOKEN	Single satoshi NFT token, tokenizes control over the root ID

VerusID Namespace

To create a blockchain of a specific name, you need a VerusID of the same name. The controller of this VerusID is the only one who can create a blockchain of that name, and they can only do so once.

Examples of blockchain launches

Limit for all currency supplies (10 billion)

10 billion (-1 SATOSHI) with 8 decimal places (999999999.9999999) is now the recommended absolute limit for all currency supplies, including over time with conversions and extended tail emissions for blockchains.

Blockchain 1

```
./verus -chain=VRSCTEST definecurrency '{
  "name": "PBaaSChain",
  "options": 264,
  "currencies": ["VRSCTEST"],
  "conversions": [1],
  "eras": [
    {
      "reward": 1200000000,
      "decay": 0,
      "halving": 0,
      "eraend": 0
    }
  ],
  "notaries": [
    "Notary1@",
    "Notary2@",
    "Notary3@"
  ],
  "minnotariesconfirm": 2,
  "nodes": [
    {
      "networkaddress": "111.111.111.111:10000",
      "nodeidentity": "Node1@"
    },
    {
      "networkaddress": "111.111.111.112:10000",
      "nodeidentity": "Node2@"
    }
  ],
  "gatewayconvertername": "Bridge",
  "gatewayconverterissuance": 1000000
}'
```

```

        }'
        '{
            "currencies": [ "VRSCTEST", "PBaaSChain", "USD" ],
            "initialcontributions": [ 371747.20398827, 0, 1000000 ],
            "initialsupply": 3000000
        }'
    }
}

```

Blockchain 2

```

./verus -chain=vrsctest definecurrency '{
    "name": "v2",
    "options": 264,
    "currencies": [ "vrsctest" ],
    "preallocations": [
        {
            "allnotary1@": 800000
        }
    ],
    "conversions": [ 1 ],
    "eras": [
        {
            "reward": 76800000000,
            "decay": 0,
            "halving": 525000,
            "eraend": 0
        }
    ],
    "blocktime": 10,
    "idregistrationfees": 25,
    "notaries": [
        "allnotary1@",
        "allnotary2@",
        "allnotary3@"
    ],
    "startblock": 9500,
    "minnotariesconfirm": 2,
    "nodes": [
        {
            "networkaddress": "45.76.168.133:20022",
            "nodeidentity": "allnotary1@"
        },
        {
            "networkaddress": "149.28.95.28:20022",
            "nodeidentity": "allnotary2@"
        },
        {
            "networkaddress": "45.63.58.171:20022",
            "nodeidentity": "allnotary3@"
        }
    ],
    "gatewayconvertername": "Bridge",
    "gatewayconverterissuance": 800000
}
'
'{
    "currencies": [ "VRSCTEST", "v2" ],
    "initialcontributions": [ 2500, 0 ],
    "initialsupply": 800000
}'

```

Need help setting up a blockchain launch? 😊

Go to the Verus Discord #pbaas-development channel. The community is happy to assist! 🚀

← [Introduction](#)

Introduction

Launch blockchains that are fully interoperable, customizable and independent. Each blockchain launched has the following L1 features.

Launch Blockchains

Introduction

Launch without programming needed
DeFi liquidity pools and fractional currency baskets
Bridge converter launches
Crowdfunding currency and blockchain launches
Interoperable, multichain network
Defining a PBaaS-blockchain

Feature	Details
L1 Multi-currency	Consensus layer basket currencies (e.g. liquidity pools) & simple token currencies, decentralized crowdfund mechanisms
L1 DeFi	Consensus layer AMMs, MEV-free, no front/back-running, solving all conversions simultaneously within a block & fully decentralized marketplace
L1 VerusID	Consensus layer self-sovereign identities, namespace for currencies, tokens & PBaaS-blockchains, friendly-name addresses, revocable, recoverable, NFTs, profiles, can be bound to any type of data
L1 Privacy	Consensus layer zk-SNARKs privacy technology
51% hash attack resistant	50% proof-of-work, 50% proof-of-stake. Read the paper here.
75 - 800 TPS	High performance blockchain, adjustable block times 10 secs - 2 mins
Scaling	Scale out by deploying more interoperable, interconnected PBaaS-chains

It costs 10,000 VRSC to launch a PBaaS-chain. 5000 goes to the Verus block producers, 5000 goes to the block producers of the newly launched chain.

Launch without programming needed

Enables any user with VerusID to create their own token currency or even full fledged, multi-currency, VerusID-issuing 50% PoW/50% PoS, [51% hash attack resistant](#) blockchain that can send and receive from the Verus chain which launched it.

All PBaaS chains run from the same daemon, and projects may choose to join the worldwide Verus community in improving the daemon. In doing so, they will start with a complete, multi-currency, VerusID-capable blockchain with DeFi capabilities that is merge-mineable and stakeable with other blockchains in the Verus network.

DeFi liquidity pools and fractional currency baskets

Any VerusID owner may define basket currencies, with one or more reserves

backing the currency, at a fractional percentage ranging from 5% to 100% backing.

The Verus DeFi protocol ensures that all currency conversions that use a particular basket currency, and are mined into one block are solved and priced simultaneously, addressing the problems of miner extracted value (MEV) and front-running, while providing fee-based DeFi integrated incentives to miners and stakers, ensuring smooth consensus operation and fee conversion capabilities by integrating DeFi basket currencies directly into the consensus and cross-chain bridge protocols.

Bridge converter launches

Launch of a world class, worldwide, merge-mineable blockchain along with a fully decentralized or centralized “bridge” converter liquidity pool as part of defining a new blockchain.

Bridge converter currencies have the same flexibility as other 100% asset backed or partially asset backed basket currencies, but is bound to the launch of the new blockchain, runs on the new blockchain, and all fees generated via cross-chain fee conversions or general use of the liquidity pool are earned on the new blockchain with no rent going back to the Verus blockchain, only seamless connectivity.

Crowdfunding currency and blockchain launches

Set required minimum levels of worldwide participation in your preferred currencies on chain. If by the start time of your blockchain, minimums are not met, all participants will automatically get a refund of all of their pre-conversions, less the network fees.

The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral pre-allocations to select IDs that increase the fractional reserve ratio to issue currencies, similarly price neutral carve-outs of proceeds, and pre-launch discounts for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

Interoperable, multichain network

The Verus multi-currency, multi-chain network allows the creation of an unlimited number of interoperable blockchains in the Verus network.

Notary IDs, specified at chain definition, provide decentralized blockchain-specific bridge confirmation, enabling public blockchains available to the world for merge mining and staking, as well as private, internal blockchains, which are easy to setup with easy bridging of public currencies into an organization and onto their internal private network and back, with all features and currencies of the public chain but none of the access.

There is no limit on the number of blockchains that may continuously operate and interoperate on the Verus network. While there is some overhead for cross notarization, the model for the Verus blockchain network is fractal, enabling an unlimited number of simultaneously operating, interoperable blockchains.

[Defining a PBaaS-blockchain →](#)

Launch currency with 1:1 mapping of ERC-20

Launch Currencies

Introduction

Launch currencies

Launch currency with 1:1 mapping of ERC-20

Export currency to Ethereum (as ERC-20)

When defining a currency it can be mapped to an ERC-20 1:1. The currency on Verus and the ERC-20 on Ethereum are then always interchangeable 1 to 1. [More on the Verus-Ethereum Bridge](#).

↔ Verus-Ethereum Bridge For Testnet!

[Access the Verus-Ethereum Testnet Bridge](#) (⚠ Goerli testnet)

↔ Verus-Ethereum Bridge For Mainnet!

[Access the Verus-Ethereum Mainnet Bridge](#)

Need help setting up a currency launch? 😊

[Go to the Verus Discord #pbaas-development channel. The community is happy to assist!](#)

Defining the currency

To create a currency of a specific name, we need a VerusID of the same name. The controller of this VerusID is the only one who can create a currency of that name, and we can only do so once.

The cost for a VerusID on the Verus is `100 VRSC` (or 80 when using a referral). The cost to launch a currency is `200 VRSC`. Before launching we need to have enough VRSC in the namespace VerusID.

You can also use a .vETH subID. The cost to register a .vETH subID is an amount of Bridge.vETH (0.01 vETH worth).

In our example we have a namespace `MyUSDC` with which we want to launch a currency that is mapped to the Ethereum `usdc` ERC-20 (on Goerli testnet, [see contract address](#)).

Below is the command to map a currency 1:1 to an ERC-20 on Ethereum. The `address` field is the Ethereum smart contract address of the ERC-20 we want to map to.

```
./verus -chain=VRSCTEST definecurrency '{
  "name": "MyUSDC",
  "options": 32,
  "systemid": "veth",
  "parent": "vrsctest",
  "launchsystemid": "vrsctest",
  "nativecurrencyid": {
    "type": 9,
    "address": "0x98339D8C260052B7ad81c28c16C0b98420f2B46a"
  },
  "initialsupply": 0,
  "proofprotocol": 3
}'
```

After we put in the command, we get returned a HEX. We use this HEX to launch the currency on the network. Use the command below to launch the currency:

```
./verus -chain=VRSCTEST sendrawtransaction "HEX"
```

json

Now we have to wait a few blocks for the currency to be available on the network.

Export to Ethereum

The last step is to export the currency to Ethereum so we can see it there too. 

[Read it here](#)

[← Launch currencies](#)

[Export currency to Ethereum \(as ERC-20\)](#) →

Export currency to Ethereum (as ERC-20)

Launch Currencies

Introduction

Launch currencies

Launch currency with 1:1 mapping of ERC-20

Export currency to Ethereum (as ERC-20)

Any currency on Verus can be exported to Ethereum as an ERC-20. The currency can then be used in the complete Ethereum ecosystem, and on the Verus network. Sending to and from Verus and Ethereum couldn't be easier. [More on the Verus-Ethereum Bridge](#).

↔ Verus-Ethereum Bridge  For Testnet!

 [Access the Verus-Ethereum Testnet Bridge](#) (⚠ Goerli testnet)

↔ Verus-Ethereum Bridge  For Mainnet!

 [Access the Verus-Ethereum Mainnet Bridge](#)

Exporting the currency

Now, let's export a currency from Verus to Ethereum as an ERC-20 over the non-custodial bridge. We must have enough funds to pay for the export.

The command to export a currency to Ethereum as an ERC-20:

```
./verus -chain=VRSCTEST sendcurrency "myVerusID@" '[{  
    "address": "0x85a7dE2278E52327471e174AeeB280cdFdC6A68a",  
    "currency": "myCurrency",  
    "amount": 0,  
    "exportto": "veth",  
    "exportcurrency": true,  
    "feecurrency": "veth"  
}]'
```

json

Let's break the command down with what you can change.

myVerusID@ is the from- and change-address. Can be a VerusID, R-address or i-address. The fee to pay for the export comes from here, and if you paid too much fees the rest will be returned here.

address You can fill in any ETH-address here, it is actually not important what is here.

currency is the name of the currency you wish to export as an ERC-20. E.g. **MyCurrency** , **MyCurrency.vETH** .

Wait for notarization

After the bridge has been notarized to the blockheight where you have exported the currency, you can choose it from the token dropdown on the [Bridge website](#)

[mainnet](#) or [Bridge website](#) [Goerli testnet](#)

← [Launch currency with 1:1 mapping of ERC-20](#)

Introduction

Launch Currencies

Introduction

Basket currencies (e.g. liquidity pools)

Simple token currencies

Ethereum ERC-20

Crowdfund mechanisms

Launch currencies

Launch currency with 1:1 mapping of ERC-20

Export currency to Ethereum (as ERC-20)

Launching currencies on Verus, and any other PBaaS-chain (Public Blockchains as a Service), is better, faster, cheaper and more secure than any EVM-like protocol out there. There is no coding involved.

There are two types of currencies that can be launched with the Verus Protocol. Basket currencies (e.g. liquidity pools) and simple token currencies. Both can be issued decentralized, or centralized with minting and burning capabilities.

When a currency is launched, subIDs can be created from it. SubIDs are powerful objects on Verus (and other PBaaS-blockchains). They are exactly the same as [VerusIDs](#), yet can not launch blockchains or currencies.

Basket currencies (e.g. liquidity pools)



Basket currencies function like automated market makers (AMMs), they have reserves. A reserve can be any currency or token on the Verus network (also bridged). Have a look at the simplified image. If anyone has currency X or Y, they can convert to the basket currency, or convert from reserve to reserve. If anyone has the basket currency, they can go to currency X or Y. A basket currency can have 1 and up to 10 currencies in its reserves.

The basket currency supply is dynamic, depending on how much is converted to the basket currency (supply minted), or back to its reserve(s) (supply burned).

A basket currency can be 100% backed by its reserves, 5%, or anything in between. This is called the reserve ratio, or the weight. The lower the reserve ratio, the more volatile the currency is when people are converting into or out of the basket currency. The value of the basket currency is directly linked to what is in the reserves and what the reserve ratio is.

When a centralized version of this currency is created, the owner of the rootID can mint currencies into existence, while automatically lowering the reserve ratio. Or they can burn currencies and automatically raise the reserve ratio. Anyone can also just burn the currency at will without raising the reserve ratio.

The conversion fees are incredibly low, 0.025% when converting to and from a basket currency, and 0.05% when converting from reserve to reserve currency. These fees go directly to the worldwide miners and stakers of the protocol, and/or they are accrued into the reserves making the basket currency worth more.

Conversion type	Fee	Fee goes to
Basket currency ↔ reserve	0.025%	0.0125% added to reserves of the basket currency, 0.0125% to the block reward for miners and stakers
Reserve ↔ reserve	0.05%	0.025% added to reserves of the basket currency, 0.025% to the block reward for miners and stakers

Because all currency conversions are solved simultaneously inside a block, giving all participants the same price, the protocol is MEV-free (no front-running, back-running, sandwich attacks etc.). The protocol doesn't have any of the problems EVM-like account-based systems have. Verus DeFi is fair, cheap and has no rent-seekers.

Every(!) currency and token on the Verus network (also mapped ERC-20s!), can be used as reserves for basket currencies. As you might start to understand now — basket currencies are unique currencies that can not be found anywhere else and offer an enormous amount of opportunities for value creation.

Simple token currencies

Simple token currencies are just currencies without any reserves. They are not as exciting as the basket currencies, yet still offer much value. With all the parameters that can be added, subIDs created and decentralized crowdfund mechanisms, these currencies can support a lot of use cases that are difficult to do with alternative protocols.

The supply of this type of currency is static when it's a decentralized version. When it's a centralized version, the owner of the rootID can mint tokens into existence, and anyone can burn them.

This option is also used to create currencies that are mapped to Ethereum ERC-20s. Which means you can send those ERC-20s over to Verus, or from Verus to the ERC-20. This is made possible with the non-custodial Verus-Ethereum Bridge. You can read more about it [here](#).

And of course, a simple token currency can be one of the reserves in a basket currency.

Ethereum ERC-20

The Verus-Ethereum Bridge makes it possible for currencies and tokens to be sent over to Ethereum, and back to Verus. It is a true non-custodial bridge. All tokens and currencies flowing over the bridge are never in anyone's custody, and are proven and verified by consensus rules. [Everything on the Verus-Ethereum Bridge here.](#)

Any currency and token, on Verus or any other PBaaS-blockchain, can be exported to Ethereum as an ERC-20. They can then be used in the Ethereum ecosystem.

Also, any already existing ERC-20 token can be mapped one-to-one as a Verus currency. Meaning any ERC-20 token can live on the Verus blockchain (or any other PBaaS-chain) and take advantage of all the L1 features.

Crowdfund mechanisms

All currencies can be launched through decentralized crowdfund mechanisms. [Set required minimum levels](#) of worldwide participation in your preferred currencies. If by the start time of your currency or token, minimums are not met, all

participants will automatically get a refund of all of their preconversions, less the network fees.

The launch options also provide for maximum participation in one or more currencies, pre-launch discounts, price neutral **pre-allocations** to select VerusIDs that increase the reserve ratio to issue currencies, similarly price neutral **carve-outs of proceeds**, and **pre-launch discounts** for early participants. Using VerusIDs, launches can also include vesting schedules in the pre-allocations as well.

[Launch currencies →](#)

Staking

Economy

- Network Economy
- Mining
- Staking**
- Solo staking
- Pool staking

Staking is accessible for everyone. Use Verus holdings to secure the network. It does not matter how much \$VRSC you have, there are no minimum requirements to start staking.

On the Verus blockchain UTXOs (unspent transaction outputs) are staking, not balances. A large UTXO has more chances of winning a block than a small UTXO.

[What is a UTXO?](#)

Solo staking

There are a few rules you need to know before you can start staking.

Rules
Wallet Running
Full Node Required (native mode in Verus Desktop)
Staking Enabled
UTXO Eligible After 150 Blocks
Minimum of 0.00000001 VRSC in Wallet

With solo staking you either win a full block or you win nothing.

Need help with staking?

Join the Verus Discord #staking channel. The community is happy to assist! [↗](#)

Pool staking

There are two ways to do pool staking with Verus. One is keeping control over your funds in combination with VerusID, the other is sending your coins to a pool operator and trusting your funds with them.

A non-custodial staking pool: [Synergy Pool](#)

Economy

Network Economy

Mining

[Solo vs Pool Mining](#)

[Solo Mining](#)

[Pool Mining](#)

Suitable Devices

Mining Software

CPU

Mobile

GPU

Mining Pools

Staking

Mining

Solo vs Pool Mining

Solo Mining

Mine solo through Verus Desktop to receive full block rewards. Depending on hashrate it may take a while before winning a block. No additional software is needed to get started.

Pool Mining

Participate in pool mining to receive regular rewards. You will need to set up a few things before you can start.

Quick Comparison

	Solo mining	Pool mining
Full Node	yes	no
Regular Rewards	no	yes
Setup Difficulty	easy	intermediate

Suitable Devices

Mine Verus with various devices. Profitability indication means the electricity usage vs hashrate.

Device	Profitability Indication
CPU (processor)	high
GPU (graphics card)	medium
Mobile Phone	high
ARM (not RP4)	high
FPGA	not possible
ASIC	not possible

Hashrate Comparison

Compare hashrates from different devices. See [Community Reported Hashrates](#)

Mining Software

Need help with mining? 

Go to the Verus Discord #mining channel. The community is happy to assist! .

CPU

Software to mine with a CPU (processor).

CCminer

OS	Name	Version	Download
Windows	CCminer	v3.8.3a	Download 
macOS (Apple silicon)	CCminer	--	not (yet) available
Linux	CCminer	v3.8.3a	Download 
ARM	CCminer	v3.8.3a	Download 

Mobile

Start mining with your phone.

VerusMiner [Download here !\[\]\(d86c3a3eecf8736ae80a0b746d223709_img.jpg\)](#)

GPU

No software yet.

Mining Pools

Pool Name	Fee %
Verus Pool  Fees donated to the Verus foundation	5%
LuckPool 	1%
ZergPool 	0.5%
CiscoTech 	1%
LePool 	1%
Zhuao 	1%
AlphatechIT 	0.2%
Wattpool 	0.5%
011Data 	0.5%
vipor.net 	0.1%

Pool Name	Fee %
cloudiko.io	0%

[← Network Economy](#)

[Staking →](#)

How to bridge from Verus to Ethereum

Verus-Ethereum Bridge

What is the Verus-Ethereum Bridge

How to bridge from Ethereum to Verus

How to bridge from Verus to Ethereum

Verus Mobile

Verus Desktop

CLI

Export currency to Ethereum (as ERC-20)

Launch currency with 1:1 mapping of ERC-20

Bridge VRSC, ETH, DAI, MKR, Bridge.vETH and any other bridged assets from the Verus blockchain to the Ethereum blockchain.

To bridge from Verus to Ethereum gas fees have to be paid to use the smart contract on the Ethereum side. It can be very expensive.

You can estimate the bridging costs through Verus Mobile or if you are on a computer use the following command: (⚠ KEEP THE `1` AT THE END OR YOU WILL SEND THE ACTUAL AMOUNT)

```
run sendcurrency "*" '[{
  "currency": "veth",
  "exportto": "veth",
  "address": "0x71518580f36FeCEFFe0721F06bA4703218cD7F63",
  "amount": 1,
  "refundto": "john doe@"
}]' 1 0.0001 1
```

json

Or use:

```
curl --location 'https://YOUR-API-SERVER' \
--header 'content-type: text/plain;' \
--data-raw '{
  "jsonrpc": "1.0",
  "id": "curltext",
  "method": "sendcurrency",
  "params": [
    "*",
    [
      {
        "currency": "veth",
        "exportto": "veth",
        "address": "0x71518580f36FeCEFFe0721F06bA4703218cD7F63",
        "amount": 1,
        "refundto": "john doe@"
      }
    ],
    1,
    0.0001,
    true
  ]
}'
```

json

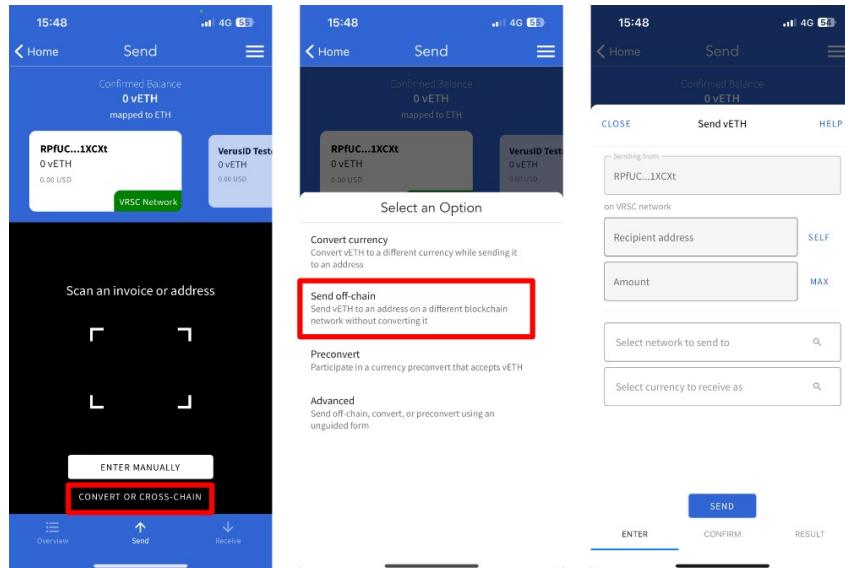
Then you look for the number after `i5w5MuNik5NtLcYmNzcvaoixooEebB6MGV`, it is the fee you have to pay in VRSC to cross from Verus to Ethereum.

⌚ The bridging from Verus to Ethereum can take 30 - 60 mins.

Learn how to go from Verus to Ethereum with [Verus Mobile](#), [Verus Desktop](#) or [CLI](#).

Verus Mobile

To bridge over from Verus to Ethereum using Verus Mobile you can go to vETH (or DAI.vETH, MKR.vETH, VRSC or other bridged assets). Then click on "Send", then "Convert or Cross-chain", then "Send off-chain".



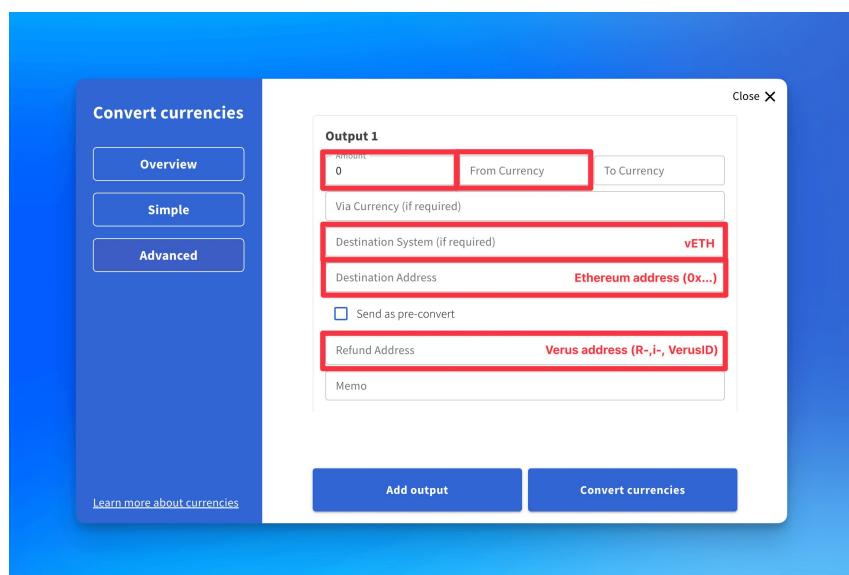
Recipient Address is your Ethereum address.

Select network to send to select **vETH**

Then follow the next steps.

Verus Desktop

To bridge over from Verus to Ethereum using Verus Desktop you can press "Convert Currencies" and go to the "Advanced"-tab. Fill in the red highlighted inputs.



In the **From Currency** you can put `veth` , `dai.veth` , `mkr.veth` , `vrsc` or other bridged assets. Not capital sensitive.

Destination System is `veth` for Ethereum.

Destination Address is your Ethereum address.

Refund Address is your Verus address (R-,i- or VerusID).

CLI

To bridge over from Verus to Ethereum using CLI. You can also use this command with the built-in command-line interface in Verus Desktop, under Settings -> Coin Settings, just replace `./verus` with `run` .

```
./verus sendcurrency "*" '[{  
    "currency": "veth",  
    "exportto": "veth",  
    "address": "ETH-ADDRESS",  
    "amount": 100,  
    "refundto": "VERUS-ADDRESS"  
}]'
```

json

currency: `veth` , `dai.veth` , `mkr.veth` , `vrsc` or other bridged assets. Not capital sensitive.

exportto: keep as `veth`

address: your Ethereum address

refundto: your Verus address (R-,i- or VerusID)

You can also add `"feecurrency": "veth"` to pay the fees in vETH. Standard is in VRSC.

← [How to bridge from Ethereum to Verus](#)

[Export currency to Ethereum \(as ERC-20\)](#) →

What is the Verus-Ethereum Bridge

Verus-Ethereum Bridge

What is the Verus-Ethereum Bridge

What can the Verus-Ethereum Bridge do

What makes the Verus-Ethereum Bridge secure

Bridge.vETH currency

.vETH subID

Contract addresses

How to bridge from Ethereum to Verus

How to bridge from Verus to Ethereum

Export currency to Ethereum (as ERC-20)

Launch currency with 1:1 mapping of ERC-20

The Verus-Ethereum Bridge (fully operational since Oct 20, 2023) allows for the secure transfer and conversion of cryptocurrencies between Verus and Ethereum. It's **trustless** and **non-custodial**, meaning it doesn't require users to trust a third party with their funds, and **no single entity has control over the assets being transferred**.

The Bridge stands out because it avoids common security issues found in other cryptocurrency bridges by using the decentralized network of miners and stakers to verify and account for funds crossing the bridge.

👉 Access the Verus-Ethereum Bridge [with MetaMask or WalletConnect](#), or download Verus Mobile for [iOS](#) and [Android](#).

What can the Verus-Ethereum Bridge do

The trustless and non-custodial Verus-Ethereum Bridge can be used for the following things:

What	Details
Conversions	Convert VRSC, ETH (vETH), DAI (DAI.vETH) & MKR (MKR.vETH) with each other into any direction (on/to Verus & Ethereum), or to and from the Bridge.vETH currency.
Launch mapped currencies	Launch currencies on Verus that are 1:1 mapped to any ERC-20. Learn how
Export currencies as ERC-20	Export any launched currency (simple tokens, basket currencies) on Verus as an ERC-20. Learn how
Cross-chain sends	Send any tokens, basket currencies (e.g. liquidity pools), mapped currencies etc. that are exported to Ethereum across the bridge.
Export tokenized ID control	Export a tokenized VerusID to Ethereum as ERC-721.
Map VerusID to Ethereum NFT	Launch a tokenized ID with a mapping of an Ethereum ERC-721 or ERC-1155.

Then there is the bridge currency Bridge.vETH, a 100% backed currency with 4 currencies in its reserves (VRSC, ETH, DAI, MKR). The Bridge.vETH currency function is to make the bridging of assets simple. From wherever you send it converts the fees that you need seamlessly. [More on Bridge.vETH below](#).

What makes the Verus-Ethereum Bridge secure

The Verus-Ethereum Bridge is different because the assets are never in anyone's

custody. This is done through the seamless cooperation between the block producers (worldwide miners and stakers), community notary witnesses, [the Bridgekeeper software](#) and the [Ethereum smart contract](#). At each step during cross-chain transactions the assets are verified and proven by consensus rules, with safeguards in place to prevent hacks.

Every 10 blocks the block producers create a notarization (when there is traffic over the bridge). They create these digital receipts for both Verus and Ethereum. The digital receipts, called "notarizations", contain, among other things: the "stateroot" ([Merkle Mountain Range](#) for Verus, [Merkle Patricia Trie](#) for Ethereum), the blockheight, blockhash and the gas price for Ethereum. The notarizations have to be agreed to by the block producers (miners and stakers) and are then mined into the Verus blockchain.

 [Read more here](#) on how the Verus Protocol handles cross-chain communication (PBaaS-chains and more) in a decentralized and provable way.

Safeguards against bridge hacks

Threats caused by malicious notary witnesses, or stolen keys to drain funds are not viable against the Verus-Ethereum Bridge. To successfully mount an attack on the bridge, if a majority of witnesses were colluding or got their private keys stolen the following would need to happen:

- Colluding, malicious witnesses.
- Fake block producers with more combined hash and stake power than the publicly validated Verus blockchain.  [Verus Paper: A Provable Hybrid Solution to 51% Hash Attacks](#)
- Developers helping them by creating an alternate protocol for the shadow chain.

These requirements are very close to the requirements of attacking any blockchain. The bridge even provides a way to defend against such an unlikely scenario.

The notary witnesses are also monitoring notarizations, and if they were to sign for something that they themselves do not agree with, they can auto-revoke their identities, using the VerusID protocol, which cannot be stopped by an attacker unless they have stolen both the keys for the notary ID and those for its revocation ID as well. This serves as a prevention for stolen key attacks, ensuring that notaries are extremely hard targets to compromise.

Bridge.vETH currency

Bridge.vETH is a 100% backed currency with 4 currencies in its reserves (VRSC, ETH, DAI, MKR), [read more on basket currencies](#). The Bridge.vETH currency function is to make the bridging of assets simple. From wherever side on the bridge you send it converts the fees that you need seamlessly.

The value of Bridge.vETH increases relative to reserves when fees or interest are added to the reserves without there being new Bridge.vETH minted.

 Accrued fees	Details
Dai Savings Rate	5% interest (at the time of writing, the rate is subject to change by MakerDAO) is earned automatically when holding Dai in the DSR (Dai Savings Rate) contract. Dai in Bridge.vETH and in the complete Verus ecosystem get this savings rate. The DSR is being passed through 100% to the DAI reserves of Bridge.vETH.
.vETH subID registrations	A .vETH subID costs 0.01 vETH worth of Bridge.vETH. When registering the subID the Bridge.vETH is burned, meaning the Bridge.vETH supply decreases.
Conversion fees	50% of the conversion fees go into the reserves of Bridge.vETH: 0.025% when it's a reserve to reserve conversion, 0.0125% when it's a reserve to Bridge.vETH conversion (or vice versa).
Cross-chain send fees	A share of the cross-chain send fees go into the reserves of Bridge.vETH.

Bridge.vETH had an initial supply of 100,000. During the preconversion timeframe which lasted 10 days, anyone could add VRSC, ETH, DAI and MKR into its reserves. When the preconversion period ended everyone got their share of the 100,000 Bridge.vETH, distributed by the protocol.

The supply of Bridge.vETH is dynamic. The currency gets minted when people convert VRSC, ETH, DAI or MKR to Bridge.vETH, and the currency gets burned when they convert from Bridge.vETH back to VRSC, ETH, DAI or MKR.

Converting currencies using Bridge.vETH (or other basket currencies) has many advantages. It is MEV-resistant, has no smart contract risks due to protocol level security and has low fees (max. 0.05%). [More on Verus DeFi here](#).

See statistics for Bridge.vETH here: verus.io/eth-bridge

.vETH subID

Register a .vETH subID to launch a currency with a 1-to-1 mapping of an ERC-20. The cost is 0.01 vETH worth of Bridge.vETH.

[Learn here how to register a VerusID / subID.](#)

Contract addresses

What	Contract address
Verus-Ethereum Bridge smart contract mainnet	0x71518580f36FeCEFFe0721F06bA4703218cD7F63
VRSC token address	0xBc2738BA63882891094C99E59a02141Ca1A1C36a
Bridge.vETH token address	0xE6052Dcc60573561EcEf2D9A4C0FEA6d3aC5B9A2

[How to bridge from Ethereum to Verus →](#)

Setup verus-cli

Download verus-cli for Windows, macOS and Linux

The CLI (command-line-interface) wallet is an alternative to Verus Desktop, and is used through the terminal.

In the packaged file you find `verusd`, `verus`, `fetch-bootstrap`, `fetch-params` and accompanying text files to [verify](#) signatures.

Run `verusd` to start the blockchain. The `d` stands for daemon. A daemon is a program that runs as a background process.

Run `verus` to interact with the blockchain.

New System

Run `fetch-params` before using `verusd` when you are running a new system. It downloads the zcash parameters needed to get started.

Bootstrap Blockchain

Downloading the blockchain can take up a long time, to speed it up you can bootstrap the blockchain. Run `fetch-bootstrap`.

Extract the packaged file on your computer, preferably where you can find it back easily.

Use Commands

In your terminal, go into the directory where you extracted verus-cli.

Start Verus Blockchain

```
./verusd
```

Start Testnet Blockchain

```
./verusd -chain=VRSCTEST
```


Set up Verus Vault in Verus Desktop (flags)

Guides

Overview

Get a Verus address

Setup verus-cli

[Set up Verus Vault in Verus Desktop \(flags\)](#)

Vault with TimeLock

Vault with DelayLock

Set up Verus Vault in Verus Desktop (easy method)

Divert staking rewards to different wallet

Claim refunds on the Verus-Ethereum Bridge



Lock your funds with Verus Vault

Verus Vault is not yet accessible with clickable interfaces. You can still set up Vault in Verus Desktop. Here's how.

What do you need:

- Latest version Verus Desktop
- VerusID (on the Verus mainchain, or when PBaaS is live on any other chain)

With Verus Vault you can lock funds in your VerusID. When your funds are locked in the Vault you can not spend them anymore, they cannot leave the VerusID. You can still always continue to stake and receive coins.

You can lock a VerusID in two different ways that cannot be circumvented by anyone, except the `revocation and recovery authorities` together.

Lock type	How it works
TimeLock	Locks the funds and unlocks until a predetermined number of blocks have passed.
DelayLock	Locks the funds and unlocks with a delay. Funds can not be spent until an unlock has been requested + a predetermined number of blocks have passed.

Get Started

We need to access the commandline interface in Verus Desktop. Go to `settings` (the cogwheel top right corner), then select `Coin Settings`. Here we can fill in the commands to set up your Vault.

Vault with TimeLock



Now let's put a Timelock on a VerusID. For a TimeLock you need to know the blockheight of the blockchain. Let's say the blockchain's blockheight is at `1,000,000` blocks. You want to lock your VerusID for 1 year. 1 year is `508994` blocks.

Long-Term Locking

For long-term locking it's best to take an average block time of 62

seconds. Yet there are some variables that make it difficult to predict an exact time, leap years for example. Over long-term there are an average of 1394.5484 blocks per day.

- Under `timelock` you put `1508994`
- Under `name` you put your VerusID name without `@`
- Under `primaryaddress` you put the R-address which is the primary address of your VerusID (you can find this address in the 'VerusID'-tab, open your VerusID and click 'ID Info')

So in our example your VerusID is locked for approximately for 1 year. After that period of time the funds can be spent again.

```
run updateidentity
'{{
  "name": "youridentityname",
  "minimumsignatures": 1,
  "primaryaddresses": ["primary-address-comes-here"],
  "flags": 0,
  "timelock": 1534360
}}'
```

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs. In the code snippet are the minimum requirements to update your VerusID with a lock.

Revoke & Recover

Remember: you can always revoke and recover a locked VerusID.

Vault with DelayLock



Now let's put a Delaylock on a VerusID. This means that you lock the identity, and when you request the identity to be unlocked, a predetermined number of blocks have to pass before you can actually spend the funds again.

Set the DelayLock

Let's say you want to put a Delaylock of 1 week. 1 week is `10,080` blocks (1440x7). This will lock the identity, and when you requested an unlock, it takes 1 week (or 10,080 blocks) before the funds can be spent again.

- Under `timelock` you put `10080`
- Under `name` you put your VerusID name without `@`
- Under `primaryaddress` you put the R-address which is the primary address of your VerusID (you can find this address in the 'VerusID'-tab, open your VerusID and click 'ID Info')

```
run updateidentity
'{{
  "name": "youridentityname",
  "minimumsignatures": 1,
  "primaryaddresses": ["primary-address-comes-here"],
```

```
"flags":2,  
"timelock":10080  
}'
```

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs. In the code snippet are the minimum requirements to update your VerusID with a lock.

Revoke & Recover

Remember: you can always revoke and recover a locked VerusID.

Request an unlock

Above you locked a VerusID with a DelayLock. Now let's request an unlock. When an unlock has been requested you have to wait the predetermined number of blocks before you can spend the funds again.

Fill in your VerusID `name`, the `primary address` and set `flags` to 0. Now you only have to wait for your predetermined number of blocks (10,080 in our example) to run out so you can spend the funds again.

```
run updateidentity  
'{  
  "name": "youridentityname",  
  "minimumsignatures":1,  
  "primaryaddresses":["primary-address-comes-here"],  
  "flags":0  
'}
```

json

You can copy and paste this code snippet into the commandline interface of Verus Desktop and edit the necessary inputs for your needs. In the code snippet are the minimum requirements to unlock your VerusID.

← [Setup verus-cli](#)

[Set up Verus Vault in Verus Desktop \(easy method\)](#) →



Privacy, Community, Power

Verus consensus non emitur

“True consensus cannot be bought” - anonymous

Michael J. Toutonghi

Michael F. Toutonghi

Alex R. English

June 12, 2018

June 12, 2018

Abstract

The Verus Project aims to establish a secure, privacy-centric, fairly-distributed cryptocurrency. But – beyond this currency – Verus seeks to become much more than a zero-knowledge privacy coin, one with two completely new highly-decentralizing proof of work and proof of stake algorithms.

In addition to payments, decentralization, and privacy features, Verus Project plans include its direct use as a currency for provisioning scalable and secure public blockchains as a service (PBaaS), for Verus applications built upon these parallel chains to scale. What this will do is simple: It will enable all people – as well as all nodes on the Verus network to participate in and benefit from a decentralized, blockchain service economy.

This paper details the Verus vision and describes the function of Verus as its own platform, and also as a member of the Komodo platform ecosystem, in the context of its first applications. Verus core applications will provide a foundation to build additional applications and services, which will leverage Verus’ automatically created blockchains, called autochains. Autochains – or PBaaS – will be provisioned and notarized by the Verus blockchain miners and stakers, in exchange for Verus currency.

Autochains will be validated through proof-of-stake by their user populations. In addition to extremely scalable, dynamic, publicly-secured autochain applications, this will add a dynamic isolation and security component to applications that can also create, manage, and verify transactions on the main Verus chain or any other Komodo-compatible, Crypto-Conditions [17], Interledger Protocol [25] enabled blockchains.

The ways we apply this technology to our world – to our biggest contemporary challenges – has the potential to completely remake the fabric of our society.

Preface

Human progress leapt forward with the invention of money. Money enabled worldwide, trade-based economies to move from a primitive, barter-based system to a consensus-based valuation and accounting of verifiable, storable, divisible, and scarce commodities as early currencies of exchange.

Even in recent times, actual or perceived scarcity and authenticity of source, whether genuine gold, an original giant coin of Micronesia [27] – or government backed notes with trusted fiscal management and the ability to exchange for oil – provide the foundation of value upon which, ultimately, human resources are bought and sold.

What blockchain does is straightforward. It enables the creation of cryptographic tools and applications that do nothing less than provide us with a new future of programmable money. This programmable money integrates directly with existent accounting functions and, eventually, with all automated services. Blockchain technologies provide humans with a new, verifiable solution to ensure scarcity, authenticity of source, inbuilt transferability, fungibility through privacy, and programmable rules. More than that, fully peer to peer, public blockchains disintermediate and provide tools that offer the opportunity to revitalize our economic systems and societies, potentially creating a more equitable and honest economic framework for all.

About This Document

This vision paper is not intended as a technical reference, but as a vehicle to communicate our vision, our plans, and our thinking – as we work on realizing the true potential of Verus.

All technologies described are based on a clear understanding of methods intended to implement each solution but – as with any project – the specific implementation details will be refined as we actually realize their functionality. To supplement this vision paper, we will follow soon after its initial release with a white paper, describing the new technologies developed for Verus Coin’s initial launch, what benefits they provide, and provide details of their implementation. We will continue to supplement this vision paper with white papers, as appropriate, for each completed phase of Verus Coin’s development, in the future.

As of this point, we have completed Phase 1, having released Verus Coin with zero knowledge privacy, two brand new algorithms combined for simultaneous CPU-mineable proof of work (POW) and fair proof of stake (POS) consensus, in-wallet mining and staking on PCs, leverage of and support for advanced Komodo platform technologies, and wallets and miners for every major PC operating system.

1 Introduction

Information vies for our attention in today's digital world, trying to convince us of what product to use or which politician to believe. We are watched and measured as we react to that information, in order to convince us what to buy or to believe. This is an active process that exploits our lack of online privacy – combined with weaknesses in human psychology – in order to make profit; extracting value from, even affecting our behavior, learning about and influencing each of us individually to open our wallets or add our voices to another's agenda.

No single person can digest and fully understand – let alone verify – a fraction of the information thrust at us in daily life. We are told to give up on the notion of privacy and to trust networks of companies with our most private data, our identity, our credit histories, our location, our habits. We are also told that our voices are lost in the digital sea of information. Yet how is this the same sea through which we are laser-targeted based on correlating our behaviors to learn so much about us, individually? What if these technologies could be turned towards the benefit of society, first – and then allowed to support businesses, in that context?

We on the Verus team believe it **is** possible to support businesses and governments requirements with digital systems that: 1) respect your privacy, 2) give you control over your data, and 3) enable you to speak your mind with the anonymous authority of an authorized voter or member of a community, in a way that can directly be heard and affect actual change.

Before we explain how, it would help to understand the shared beliefs behind our vision for Verus. These beliefs underpin everything we build into the Verus network:

1. We believe that every human has ideas, knowledge, and value to contribute to our society. By using technology to reward people for their contributions, we can enable each of our verifiable, yet anonymous voices to be heard as a collective truth.
2. We believe that those who contribute positively over time to the system should be rewarded for that contribution and provided with ongoing incentive.
3. We believe that a world-scale, peer-to-peer system that can enable humans to be queried directly, confidentially, verifiably – and in a transparent manner – can directly provide populations of people and the world with valuable, transformative tools.

With Verus, we will introduce digital tools to enable us all to build a better world together. We will monetarily incentivize – with our technologies – behavior that strengthens communities and institutions. This is the thing that's missing in the online world today: Fiscal incentive for communal behavior built into the very fabric of its function.

The Verus Project's tools will make it easy to create an identity – or multiple identities – on the Verus multi-blockchain system, which can accumulate value and even have multiple personas, each to represent a facet of your identity as a whole. This reflects how we might express identity in our personal or professional lives, where some situations call for provable credibility, yet others require no more information than what you might reveal when encountering a casual stranger.

Each identity will have its own, unique address. Unless its owner reveals information to link two or more identities, it is cryptographically hard – meaning virtually impossible – to correlate one identity to another. At the same time, the owner of an identity can still cryptographically verify statements made about identities under their control, attesting to identity properties (such as passport, age, height, citizenship, photo, etc.) as strongly as is possible with today’s digital technology. As part of the Verus vision, to be described in more detail in later phases, we intend to support fully decentralized verification of identities that can provide as strong verification as today’s centralized systems. At the same time, we believe it is important for practical reasons to enable compatibility with centralized forms of identity and to enable people to optionally support KYC in identities. To enable a smooth bridge between centralized and decentralized identity systems, today’s ID systems, including biometric and government issued IDs, will be supported via centralized or decentralized verification to enable use of Verus identity in situations that require conformance to know-your-customer (KYC) regulations. These forms of ID, however, are not required to establish or use even strong, decentralized identities on the Verus network.

Verus autochains, will operate parallel to the main Verus chain and enable large-scale applications – such as polls or elections – to run simultaneously without concern for congestion or excessive fees. Autochains will enable poll application users to provision their own secure blockchains just by using the application – spawning dynamic parallel chains that can process thousands (or potentially millions) of transactions per second when needed. Autochains will operate by proof of stake, enabling each chain to have security isolated to its direct user population. Autochains will also be backed by Verus notarization and block time synchronization – providing the full weight of its PoW/PoS security layer as well as the Komodo platform’s delayed proof of work (dPoW), to provide notarization all the way back to the full power of the actual Bitcoin blockchain.

For **Phase II**, what we expect to be an extended development phase, we will work to implement autochains and their first application in the world. We will eventually use them to create, secure, scale, and perform polls for everything from classifying online content, to identifying real public opinion, to actual, real-time elections for an organization or – conceivably – a government.

Our goal is to make these polls easy to use from a PC or mobile device, yet industrial strength and suitable for serious, secure elections. They will leverage the latest cryptographic technologies for privacy – known as zero-knowledge proofs – to preserve confidentiality.

They will be:

- **Confidential** — No one but the voter knows who or what they voted for – unless the voter discloses. Results of the vote can be withheld until the vote is complete, at which time they can be released to everyone, simultaneously.
- **Verifiable** — Only voters that are authorized to vote can vote. Each voter can vote only once. Each voter can look at the released results and see that their vote was counted.
- **Transparent** — Anyone can validate the number of votes counted, and the number of votes, each person or selection received.

- **Secure** — Our novel autochains – dynamic, security isolated, proof-of-stake (PoS) blockchains – are layered over proof-of-work (PoW) and delayed proof-of-work (dPoW) security, all the way back to the real Bitcoin blockchain through Komodo.

By default, the first layer of security is run by the actual voters, themselves. Together, this provides unprecedented layered security both in the autochain and in delayed proof-of-work, leveraging the network with the most hash power in the world.

We like to think that – on a Verus autochain – a 51% attack is called winning a poll.

- **Scalable** — Each poll is conducted on its own, automatically-created-and-validated blockchain, operating under its own visible validation rules.

2 Our Vision for Verus Foundational Applications

Throughout history, understanding what a population truly thinks or feels has been an invaluable capability. The manipulation of that understanding can, and has, repeatedly changed the course of history.

The Verus Project plans to use highly decentralized blockchain technology to enable all people to safely, anonymously, and confidently express their opinions – on any issue. Verus users will be able to share their knowledge in a public forum, query a population of humans (or eventually both AIs and humans), quickly and effectively, and – importantly – earn cryptocurrency in exchange for these contributions to collective knowledge.

A primary goal of the Verus Project is to enable societies and organizations to make decisions based on a more honest understanding of the public’s actual beliefs. To achieve this goal, we consider the need for both confidentiality and transparency, to ensure that – first and foremost – the system can be trusted. It is also critical to prioritize decentralization and develop a core platform that can leverage fully transparent blockchain technologies, while running many applications that respect privacy. With Verus, these applications can leverage zero-knowledge succinct non-interactive arguments of knowledge – or zk-SNARKs – the most reliable and tested iteration of proven, zero-knowledge privacy technology available today.

Our novel approach ensures that this technology can be used at scale, across any population size, throughout the world. Using the tested-at-scale infrastructure and tools of the Komodo platform, and Verus planned development of PBaaS – the Verus developers will build human organizational tools, including voting and fully self-sovereign identity with a reputation system that respects privacy. The goal, here, is a significant one, and these tools are planned in a way to allow entire human populations to transparently and directly share their knowledge and opinions – without the risk of privacy violation, spin or censorship.

Verus tools will also leverage human (and eventually non-human) intelligence at scale to solve previously challenging problems in a decentralized manner – financially rewarding those who contribute to their proper function.

To serve all people in the world simultaneously with these public, peer-to-peer tools, Verus introduces autochains. Autochains are a novel scaling model for blockchain systems. Autochains will enable full node miners to provide public blockchain provisioning services. These services will occur in transient, parallel chains, chains that are isolated from congestion, disruption, or interference from the main Verus blockchain. As an autochain operates, it is

almost completely parallel and isolated from the main Verus chain, except for the capability of posting transactions and proofs to Verus, for results and coordination across applications or people.

By combining privacy, polls at scale, and identity, Verus can be used to post polls and ask almost any question of a population or subpopulation of people, receive an honest – and, importantly – verifiable answer, and still respect the privacy of the respondents. When even the first support for polling is active on the Verus network, the Verus community will be able to vote on the long term direction of the Verus Project, itself.

Over time, Verus polls – used for purposes such as determining the accuracy of news online, for research, or even for political polling, will be combined with machine learning to enable all of us, as a society, to benefit directly from our collective intelligence without today's risks to our individual privacy. Verus will combine auditability and verifiability of transactions with the option of verified, confidential participation – meaning societies and organizations can directly understand what people think, feel, and believe, as populations – without taking control of their personal data or targeting some individuals, based on their answers or beliefs.

Ultimately, our vision for the Verus Project is to enable us all to directly participate in our own worldwide economy. Verus applications will enable any individual to speak up with the power of an authorized participant – and with the confidence of anonymity – to produce a verifiable, transparent, and honest flow of information throughout the globe. It will transform the way that anonymity functions in our current technological environment – prioritizing cooperation and providing the financial incentive for that cooperation to occur.

3 Important Verus Concepts

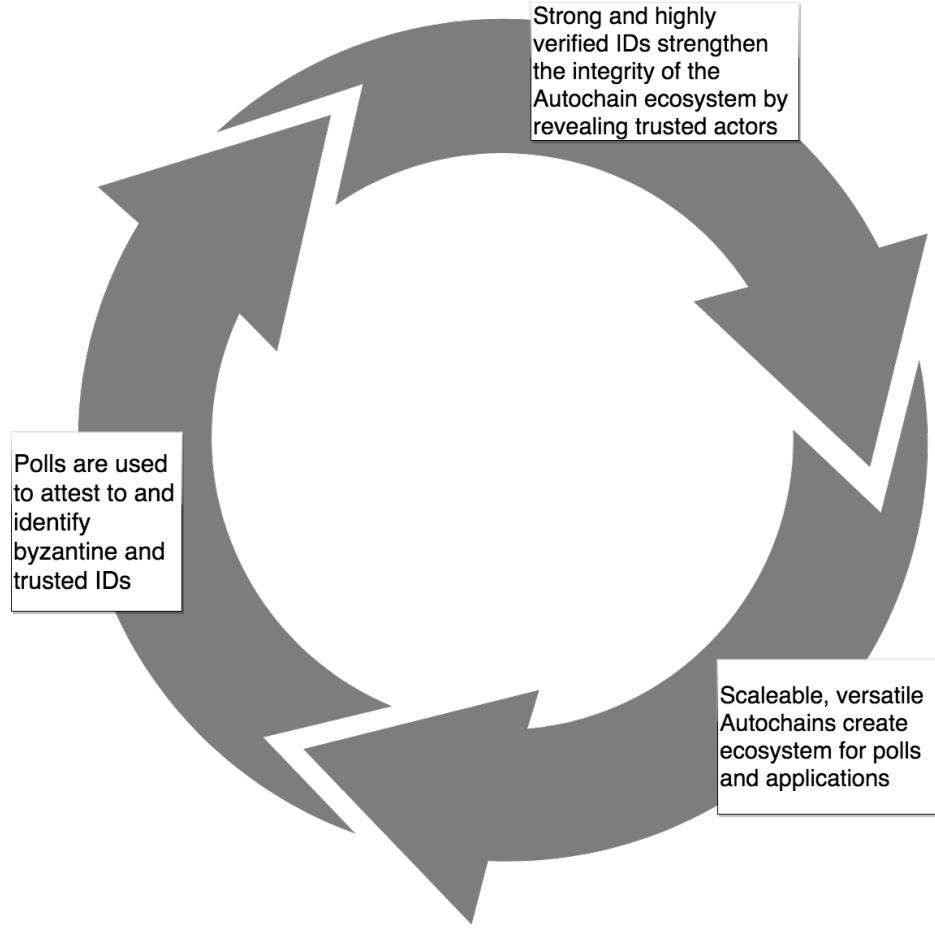
The longer term Verus Project vision of a better society through public blockchain services relies upon three fundamental pillars –where the correct operation of each strengthens the others in the form of a virtuous circle.

These pillars are:

1. Extreme transaction throughput and scalable decentralized applications with autochains, which provide public blockchains as a service (PBaaS)
2. Selectively strong, private identity, and
3. Open application support with a foundation of polls, voting, and lottery selection services.

Each of these pillars will help us process human knowledge, understanding, and/or opinion on any content or topic. Together, they will enable direct querying – with confidential and truthful – answers across an entire population. It will create a secure, public platform that respects privacy, and can potentially serve as the foundation of a more respectful society.

It is also important to know about the the concept of **Verus Virtue** when reading further in this document. The best way to think about Virtue for the purposes of this document is as a separate currency that can be earned, but not purchased or sold, and is part of the



evolution of the Verus proof-of-stake algorithm, slated for later phases of the Verus project. Proof of Virtue is a technology that will be described in much greater detail and implemented in later phases of the Verus project.

3.1 Autochains

With Verus, we plan to create a platform upon which we will build voting and even election systems that support our vision of enabling a better society through blockchain.

To run simultaneous polls across disparate populations on one blockchain-based system, we need another dimension of information – one that retains privacy, and is anchored to a primary store of value, Verus.

One approach to creating another dimension of meaning would be a system of “coloring.” The Ethereum blockchain system and some layers over Bitcoin or Bitcoin compatible coins use a version of a coloring system. [\[5\]](#)[\[6\]](#)[\[7\]](#)[\[8\]](#). While, at first glance, this seems like a reasonable approach, it is actually a suboptimal solution, at best.

With Ethereum and the ICO experiences of 2017 [\[9\]](#)[\[10\]](#), we now know what happens when you create one blockchain and overload it with transactions: Congestion and unnecessarily high fees. Developers have proposed many solutions to this problem of scaling – including a move to proof-of-stake systems and sharding (effectively trying to separate the various

functions on a blockchain), or even a move to large systems of parallel chains with a common design.

Yet instead of implementing a “solution” with colored coins sharing one blockchain, we have decided to take a completely new approach to this problem. Our approach will not only provide an extra dimension of information and an unlimited number of token types, or “colors”, but it will do so with highly scalable PBaaS autochains, which also reward Verus miners and node operators as provisioning agents. This is a novel method of scaling, one which provides significant scale and security benefits through isolation of transaction processing for each application instance on its own blockchain.

Rather than thinking of Verus as a single blockchain, it is more appropriate to think of it as a blockchain-based system, one that is rooted in a primary value chain.

Autochains are exactly what they sound like – automatically instantiated blockchains that are validated initially – and when notarizing – by Verus miners, and otherwise, only by their users, often a disjoint set from the whole of Verus users. Autochains, once instantiated, can be used indefinitely or can have a finite lifetime to be used for a specific purpose. In this document, we will often discuss autochains in the context of a few concrete applications that we intend to implement on the Verus network. It is important to keep in mind that their versatility is **basically unlimited**. Due to their security being inherited from the main Verus chain, autochains eliminate perhaps the biggest disadvantage that creating consensus-based distributed ledger applications suffers from: Weak defense against attacks in their earliest stages.

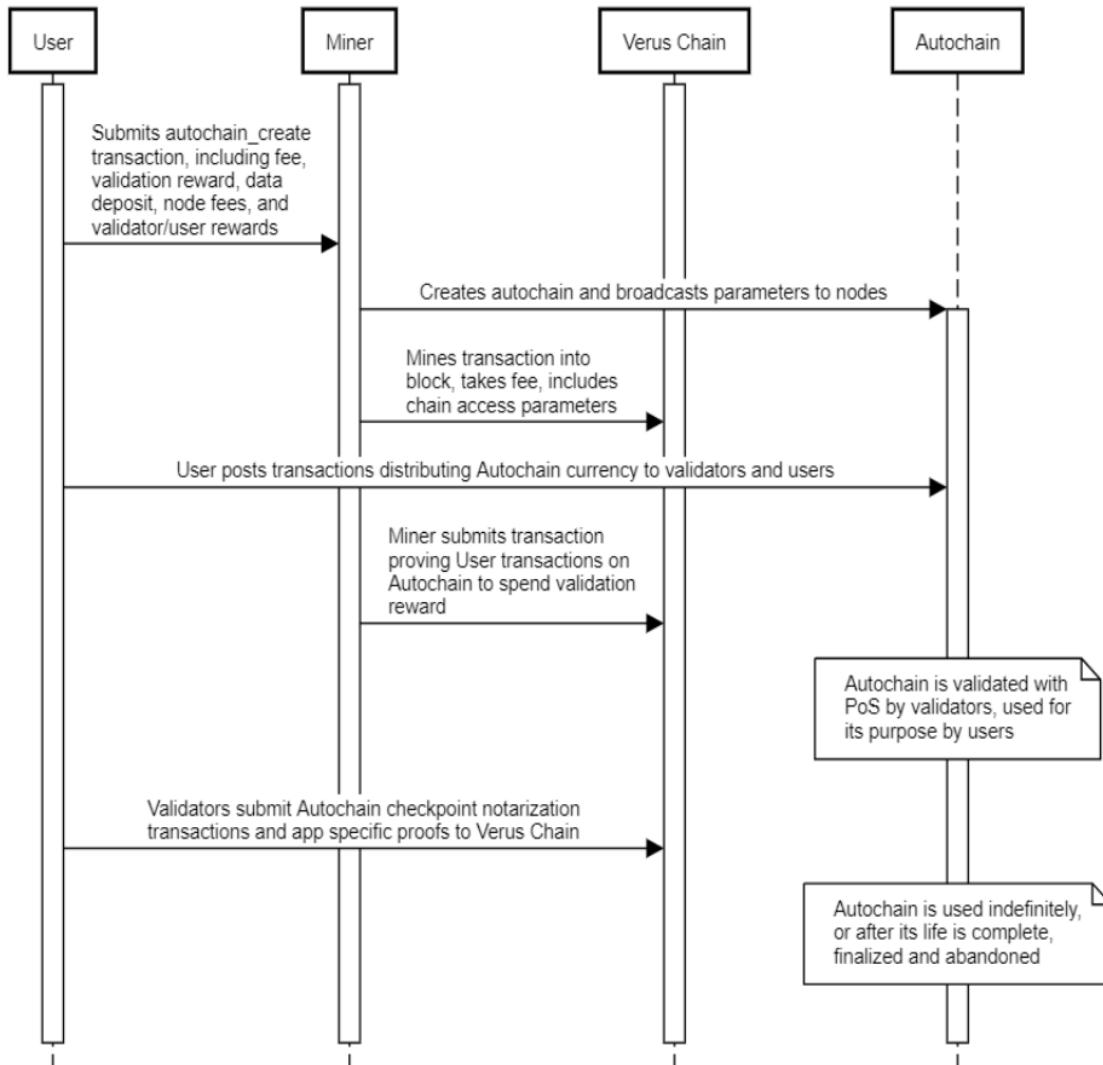
In **Phase II**, Verus will define an autochain provisioning protocol, and specific voting protocols, based on blockchain transactions. These protocols will include zero knowledge proofs, Crypto Conditions [17], and Interledger Protocol [25] technologies to provide proper incentive for all participants to engage in the automatic creation and use of on-demand, isolated or public blockchains, ones that rest on a secure, public foundation.

The autochain protocol will also provide the ability to place a value on the data generated by the autochain’s operation. This value will then either be released to the public or paid out, through rewards, to the participants in the creation of that data. This creates a model that incentivizes release of learning and information to the public – also making applications which do so much less expensive to run.

For Verus poll applications, validation will default to a form of proof-of-stake (PoS), where all members of the chain are stakers. This is because that security model matches exactly the interests of the majority of voters in having an accurate record. Because PoS does not involve powerful mining equipment or competition to solve a cryptographic puzzle, it can operate on much less powerful hardware – even on mobile devices. Since voters, themselves, secure the chain simply by running the voting software, poll chains are generally not affected by activity on the main Verus chain (except when posting interledger transactions). In addition, an autochain’s operation, independent of scale, adds no overhead or congestion to the main Verus chain’s transaction load – besides the transactions to provision the poll chain and post results back to the main chain itself.

Since we want to be able to create autochains from clients that may not be running full nodes or even own stable IP addresses, full nodes will be incentivized to provide reliable static node support for autochains. Once they provide this support, they will be recognized

Verus Autochain Lifecycle



by validators on the autochain. They will then be able to use this recognition to boost their reputation for selection in future lotteries.

By advertising autochain node support on the main blockchain – along with a stable payment address via a transaction – the node IP can be selected in a random lottery. The Eigentrust ratings of all nodes can then be recorded by all other nodes that serve the chain at regular intervals. Nodes which consistently offer better QoS will get better ratings. Nodes that get poorly rated will still share in the node rewards for the current blockchain – but will be less likely to be selected in the next node lottery.

3.2 The Virtue Autochain

3.2.1 Roots of Trust in Distributed Consensus

The Bitcoin [14] proof of work implementation introduced the world to systems that establish trust, not due to belief in any individual human's behavior, but based on the tendency of humans to act in their own self-interest.

The Bitcoin protocol rewards winners of a contest, referred to as "miners", in statistical proportion to the computing power they employ to "mine" for Bitcoin, as compared to the total computing power of all miners. For every block of validated transactions added to the Bitcoin blockchain, someone wins a competition to earn the right to define that block while adhering to specific, verifiable rules. The competition involves verifiably solving a statistical mathematical puzzle that is specific to exactly that block, meaning the same work cannot be reused on any other block. The winner of each competition earns the right to accurately process the next block of pending Bitcoin transactions and to claim a specific measure of Bitcoin, called the "block reward" (plus fees of all transactions in the block that was processed).

Mining competition serves to determine the amount of computing power a would-be attacker would have to control in order to mine blocks that were not earned according to the intended rules, execute transactions that might otherwise be rejected, and prevent certain transactions from executing altogether. Based on the way that miners achieve consensus on what is the correct chain, forging a false Bitcoin chain to achieve this would require an attacker to control more than 50% of the total power of all miners – both honest and byzantine.

Since, right now, it is likely infeasible for any single actor to mount such an attack against Bitcoin, the largest network of hash power today, this type of competition, called proof-of-work (PoW), currently manages hundreds of billions of dollars of value, sends transactions of that value to anyone, anywhere, at any time, and does this all without any company, bank, or trusted third party, of any kind.

In addition to enabling secure management of large sums of money, PoW has also sparked a mining arms race and significant investment in computing power to claim Bitcoins and other cryptocurrencies, creating the equivalent of the world's largest distributed supercomputer doing nothing but the same calculations, albeit on different data, over and over again. As a result, one side effect of the public blockchains' PoW security system is massive power consumption and significant ecological cost.

Due to this, a great deal of research and development has gone into alternative consensus mechanisms [11] [12] [13]. Most serious PoW alternatives center around the concept of proof-of-stake (PoS) – the idea, that by putting something at stake commensurate with the value being transacted in the transaction flow, a rational staking person, known as a "validator," will choose good behavior within the ecosystem, so as not to lose their "stake."

Even though large systems are being built that depend on 100% PoS, there remains controversy over its robustness when users have incompatible interests, or when an attacker's stake is not valued as highly as the perceived value to be gained by corrupting the system.

Most modern PoS and PoW systems make a **fundamental assumption** about all miners; that they are either 'byzantine', and intend to compromise the system as a whole, or

‘rational’, in statistically the same proportions for all participants equally. One need only consider that if there was a way to learn a more accurate statistical function for each participant’s probability of being **either** byzantine or rational, using such a function to determine who participates in the system’s validation would improve its resistance to attack.

In fact, any accurate method of recognizing those who were attempting to strengthen the system – and thus who could statistically be more trusted than a byzantine or even average participant – could both decrease the power consumption and ecological cost of blockchain security and further strengthen resistance to attack. This then brings up the question: **“How do we recognize trustworthy participants?”**

Some methods studied and proven effective are the Eigentrust reputation management algorithm [20], and its improved derivative the Eigentrust++ mechanism implemented in the NEM blockchain [22]. Originally designed by Sep Kamvar, Mario Glosser and Hector Garcia-Molina in 2003, the Eigentrust algorithm is built to function on peer-to-peer networks. It aims to isolate byzantine actors in said networks by assigning each peer a public, global trust value based on their history of activity – clearly displaying a form of “rating” for each peer. Those with lower ratings are shown not to be trustworthy, and thus, are interacted with less by their peers.

Simply put, the algorithm assumes that if any given peer a trusts any given peer b , then it would also trust the peers trusted by b . Every peer calculates a local trust value for each other peer it has interacted with, based on the either satisfactory or unsatisfactory transactions it has had with each one. These local trust values are determined by each peer, and when peer a wants to know if they should trust peer b , say, before making a transaction, peer a would ask all other peers it knows to report on their local trust values of peer b and weigh their responses according to the trust values peer a has for each of them.

3.2.2 Proof of Virtue (PoV) Reward System

Using a model similar to the EigenTrust algorithm, Verus will introduce the idea of Proof of Virtue (PoV) **enhancement** to the PoS algorithm initially released in **Phase I**. In PoV, we intend to weight staking contests with both Verus stake and “Virtue”, a special “currency”, expected to be tracked on a Virtue autochain, and used as a **trust rating** of identities within the network.

In a similar way to the “amount of stake” used when staking, Virtue will add another component to the probability of being selected to process a block of transactions on the Verus Network. Its probabilities upon introduction will not change returns on the Verus PoS system, but addresses will gradually be able to improve earnings with a Virtue weighting – **based on recorded activity and behavior**.

In order to ensure that Virtue is both a rare and valued property, which drives correct behavior, it will neither be purchased, nor sold. Verus holders will have opportunities to earn a small measure of Virtue, which can then be further increased by staking the Virtue itself, when attesting to a fact for the network, validating information, or providing another measurable benefit to the network.

Since Virtue will have intrinsic value on the Verus network, there will be methods for transferring it through wallet ownership – for cases of probate or other necessity – but such behavior as a tool for trading in virtue will not provide a secure or intended method of

exchange.

The Proof of Virtue model will effectively be a **modified** form of PoS that is not based just on monetary value, but also on long term measurement of contribution, which will affect trustworthiness and earning power in the Verus network. The intent is to provide opportunities for more people to participate in the growth of a **positive functioning network** and to prevent potential attackers from being able to easily buy their way into an attack, adding yet another layer of security.

We expect to keep the PoW component as part of Verus security system for an indefinite period of time. We also expect to leverage the dPoW security of Komodo and its notarization into the Bitcoin blockchain as well.

3.3 Polls, Voting, and Elections

3.3.1 The Importance of Secure Polling

A great deal of research has delved into the best way to achieve confidential, verifiable, and transparent (CVT) elections electronically – with a few notable systems actually used in real life situations. These attempts, however, have been plagued with significant limitations, limitations that almost always risked either revealing the identity of a voter and thus eliminating confidentiality, or allowing attackers to impersonate legitimate voters, and put the validity of the vote into question [1].

This was shown in one of the first majorly adopted North American electronic voting systems, the Diebold AccuVote TS, which was announced in the year 2000 [4], at a time when, as a result of the Florida 2000 presidential election, the general public began to recognize numerous flaws in the widely-used punch card voting system [2]. Despite multiple studies discussing electronic voting systems – studies which clearly warned of security risks such as insider threats, issues with auditing, and network vulnerabilities – by 2004, the Accuvote TS system was deployed for major political elections in 37 US states [2], resulting in multiple serious problems that impact the legitimacy of numerous election results since, due to inherent flaws in its design that were never fully addressed [3].

Firstly, the “solution” that the system introduced to deal with voters submitting multiple ballots – and to solve the ballot anonymity problem – was to issue a single personal voting ‘smartcard’ to each vote. This smartcard, unfortunately, did not contain a complete identity, but instead contained a common election key among voters that voting machines simply checked to determine if the card belonged in the correct election [2]. The lack of any cryptographic unique identification on these cards was a significant security flaw, as user-programmable smart cards and readers almost instantly became commercially available on the internet for reasonable prices – making it extremely easy to mass produce homemade copies. Furthermore, due to the lack of a truly secure boot loader, operating system, or application, the system had numerous potential attack vectors, many of which were quite simple for an adversary. For example, any attacker with access to the operating system had the ability to modify ballot program files through the standard Windows Explorer application already included on each machine.

These serious failures in previously adopted electronic voting systems have literally **changed the course of history**, affecting the outcomes of major political elections that

decide the state of world politics. This highlights the importance of creating a system that is truly confidential, verifiable and transparent, and indicates the magnitude of one problem we plan to solve with Verus.

3.3.2 Running, Recording, and Scaling Polls - Transient, PoS Chains

Using autochains, Verus will dynamically create parallel voting chains on demand for each poll in its ecosystem. This creates improved, isolated security and allows for scaling in a virtually unlimited manner.

The Verus wallet, which will be used for voting, will also have the ability to validate blocks through proof-of-stake, blocks which, when enabled, will allow voters who are validating the election to earn rewards. These rewards will either be paid in Virtue for intrinsic polls, when available, or from outputs of the payment transaction that instantiated the poll. When a poll is complete, results are posted back to the main Verus chain. The chain used to run it can then be archived or deleted, and it is typically abandoned.

3.3.3 Poll Content and Distributed Hash Tables

While blockchains make excellent databases for permanent, non-repudiable records and public key management, they are not the best solution for storing large amounts of data, due to their massive duplication across the distributed network. Instead, a better model for distributed storage would look something like the Interplanetary File System, or IPFS, which is a peer-to-peer file system based on the Kademlia [18]19 protocol in an implementation that currently works well enough to use as storage for create/read/delete operations using hash commitments on the Verus blockchain to represent and index content.

While we are evaluating the IPFS, we are also looking at other distributed hash table solutions, such as Open DHT [26]. A key requirement for the Verus network is that mobile devices be able to participate in its operation. Open DHT is an efficient C++ library intended for use with small devices. IPFS does not yet seem to have much support for mobile at this time, though it does have a Javascript library that can be used from the browser or mobile applications [24].

Until IPFS is supplanted with another implementation, Verus applications will assume that when off-chain content is needed, it should be retrieved from IPFS, and for future compatibility, we will add a storage and version specifier. In order to store supporting poll or other content for dApps, the content owner must ensure that the content is pinned in IPFS, until it is no longer needed. By **Phase IV**, or earlier if there is significant demand, we expect to either support or provide a blockchain-metered storage solution that is more integrated and automatic to use for poll makers. Two options for such a system include a cross-blockchain integration with a decentralized payment/pinning solution, or to offer a simple service that charges in Verus to store and automatically pin content for specific lengths of time in IPFS.

3.3.4 Voting Models and Types of Polls

Verus will support numerous types of polls, those that use weighted or unweighted voting, that seek to expose truth or opinions, and polls that rate and choose collectively. In order to

create a sensible taxonomy for polls, one that people will understand, we define the following types of voting:

1. Multiple choice
2. Weighted multiple choice
3. Ranking
4. Rating

We also define the following types of polls:

1. Polls to classify
2. Polls to select
3. Polls to rank
4. Polls to rate

We do not expect Verus to support all voting and poll types in the early phases. In fact, until **Phase IV**, the Verus core team plans to focus on known requirements of multiple choice and weighted multiple choice voting. During or after completion of **Phase IV**, expanding support for voting and poll types will be higher priority.

Each type of Verus poll may support confidential or public voting, live or delayed poll results, online or offline vote authorization, and lottery selected vote authorization. We will roll out working vote models in phases, as well, initially providing solutions for weighted and unweighted selection voting, which fits well for both political and opinion polls as well as content classification and ID verification.

3.3.5 Intrinsic Polls

Once Virtue is activated, the Verus network itself will run specific, intrinsic polls that will be regularly available to identities with Virtue, based on a statistical lottery and weighted by their actual measure of Virtue. Participating in these polls, which will be used to classify content and perform basic, human verification functions, enables participants to help the system classify and rate content across Verus applications, enabling a rating system rather than censorship to allow each voter to set preferences for the ratings of polls they may wish to see.

Miners who provision autochains for these intrinsic polls will get the same rewards as miners setting up autochains for commissioned polls. The content for these polls will be taken from other polls and will be used to classify and rate those polls for easier discoverability and the user experience of poll services.

3.3.6 Privacy

There is a fair amount of confusion among the public about privacy on the blockchain. One common belief is that Bitcoin transactions are private, when in fact all Bitcoin transactions, as well as the public addresses between which value is transferred, are public information [21]. As with the Verus token and its blockchain, poll autochains will also support zk-SNARKs to ensure that votes which have not been disclosed by a voter remain provably anonymous to all other parties, while still being verifiable by the person who holds the keys to the address that cast them. Currently, based on standard smartphone hardware specifications and the memory requirements of zk-SNARK transactions, mobile phones cannot generate z-transactions. In coming releases of new Zcash technology this year, the z-transactions will be more efficient and still provide the same privacy guarantees. When these technologies are available, we will work to support them on the Verus chain, as well as use the increased efficiency to enable z-transactions and all Verus features on mobile devices as soon as possible. Until that is available, we will work to provide a mobile wallet for Verus transparent transactions.

3.3.7 Content Classification

While Verus is designed to provide invaluable tools to people across the world, any system without censorship will also inevitably allow the underbelly of humanity to show through. That means that while we would want all people to participate, we recognize that in order for that to provide the most positive experience for all participants, we must use the Verus poll system itself to recognize, rate, and classify the content on its network, enabling those who provide such a service through participation in polls to earn, while enabling the selective filtering of content based on value judgements and classification of honest populations.

While we recognize that the best early use case for such capability is in rating and classifying non-intrinsic polls themselves, the ability to classify content is a fundamental strength that Verus will increasingly develop over time. At first, we expect such classification to be applied to relatively simple tasks for humans, such as identifying toxic comments, hate speech, or categories of information. Even though these types of classification systems have largely employed machine learning systems running against privately curated training data, our vision is for Verus content classification to enable people to earn, as they classify content better than any machine learning system, but in a way that can be followed by the best machine learning algorithms, and generalized at scale.

3.3.8 Truth vs. Opinion

When classifying content, especially when you start to consider challenging classifications, such as misleading, propaganda, generally accepted fact, accurate versions of history, vs. classification of what should generally be accepted as toxic or rated content that people could reasonably answer objectively, one must consider the difference between fact and opinion. Selecting the winner of an election is a matter of expressing an opinion. Classifying content according to level of toxicity is a question of determining the factual answer to the question: what does the majority of the voting population believe the classification to be. Determining

whether an image in one photo is the same image as that from another is a question with an objective, if not determinable answer.

To ensure that Verus polls support polls that attempt to discover accepted truths, facts, and even credible emerging arguments, Verus voters will often be able to stake and either earn or lose Verus and in later phases, Virtue, by answering in a manner that matches consensus. For other polls, which are recognized as opinion polls, all answers are equally valid contributions.

3.4 Random Sampling as a Service

The Verus system will randomized selection for many of its functions, similar to its **Phase I** PoS block validation system. Blockchain lotteries will be generalized and used for selection of participants in randomized polls and in other cases where pseudo-random sampling is desired. The general principles behind Verus lotteries are the same, regardless of whether something similar to Algorand [29] for guaranteed selection or an original Verus algorithm based on difficulty, such as Verus PoS is used for selection. Lotteries use the blockchain as a random oracle and are based on the assumption that the exact value of a specific block hash from a past block in the chain cannot be modified by a byzantine participant to weigh the odds in their favor.

3.4.1 Participating in a Lottery

Since the Verus network is completely decentralized, all lotteries, including PoS, are potential, and there is no central server selecting and sending messages to those who qualified in a lottery. When lotteries are implemented as a general feature, if you would like to participate in any Verus lottery or poll that is distributing tickets via lottery, you must first determine that you qualify for the poll's requirements. You will then need to submit a transaction that proves that you have a valid claim and spend the output transaction of the lottery ticket to your address using that proof. By default, that spend will authorize you to be a validator on the autochain for the poll you are participating in, by providing you with the currency of that poll. You may validate the poll by leaving your wallet running, getting potential rewards in Verus or Virtue for both activities.

Lotteries will be useful for selecting subpopulations based on identity and claims people make about themselves or that others claim about them, which they are willing to share. While detailed discussion of Verus identity is beyond the scope of this paper's release, the Verus system will provide self-sovereign identity with the ability for identities to make very flexible claims and have them attested to by other identities. For example, a lottery may look for males between the ages of 18 to 25 among identities willing to share that information, which will even be possible to verify as strongly as through a validated passport or driver's license. Other polls may not require the same evidence backing claims. Polls may also have restricted voting, such as local political polls, where verification and authorization for the poll must be provided explicitly, often by mail to a physical address or in person in the form of a QR code or electronically delivered transaction to your Verus wallet.

3.4.2 Proof of Stake Lottery

A block validation lottery is a somewhat different form of lottery which allows you to claim the right to validate a block and its associated reward based on proof-of-stake (for autochains) or 50/50 PoW/PoS (for the main chain). The lottery requires that a prior block hash 100 blocks past combined with a qualifying UTXO of the destination public key hashed with the staking block height and then divided by the UTXO value must be under the current proof of stake validation difficulty. By dividing by the UTXO value, Verus weights proof-of-stake or proof-of-virtue in proportion to the amount of stake or virtue in the UTXO. If no one who qualifies to validate a block is online or no one responds with a submitted block validation in a certain amount of time, the blockchain will wait until someone mines a block using proof-of-work.

3.4.3 Ticket Harvesting Lotteries

There will be multiple ways to acquire tickets to participate in any particular poll. The manual ways emphasized so far may include receiving a poll with QR code in post or email or perhaps directly from someone taking a poll. In addition to these direct ways to receive a voting ticket, Verus will enable polls to be posted on the poll blockchain as transactions, the tickets for which are distributed from its outputs by lottery. This will work when the Verus ID functionality is available, and will enable polls to require presence of specific claims or attestations on an ID, as well as the number of participants to select in the poll by creating one transaction output for each, which is spendable by a cryptocondition [17] that defines the lottery conditions and random difficulty. The difficulty determines how often a particular block will match any attempt to harvest a ticket from that block.

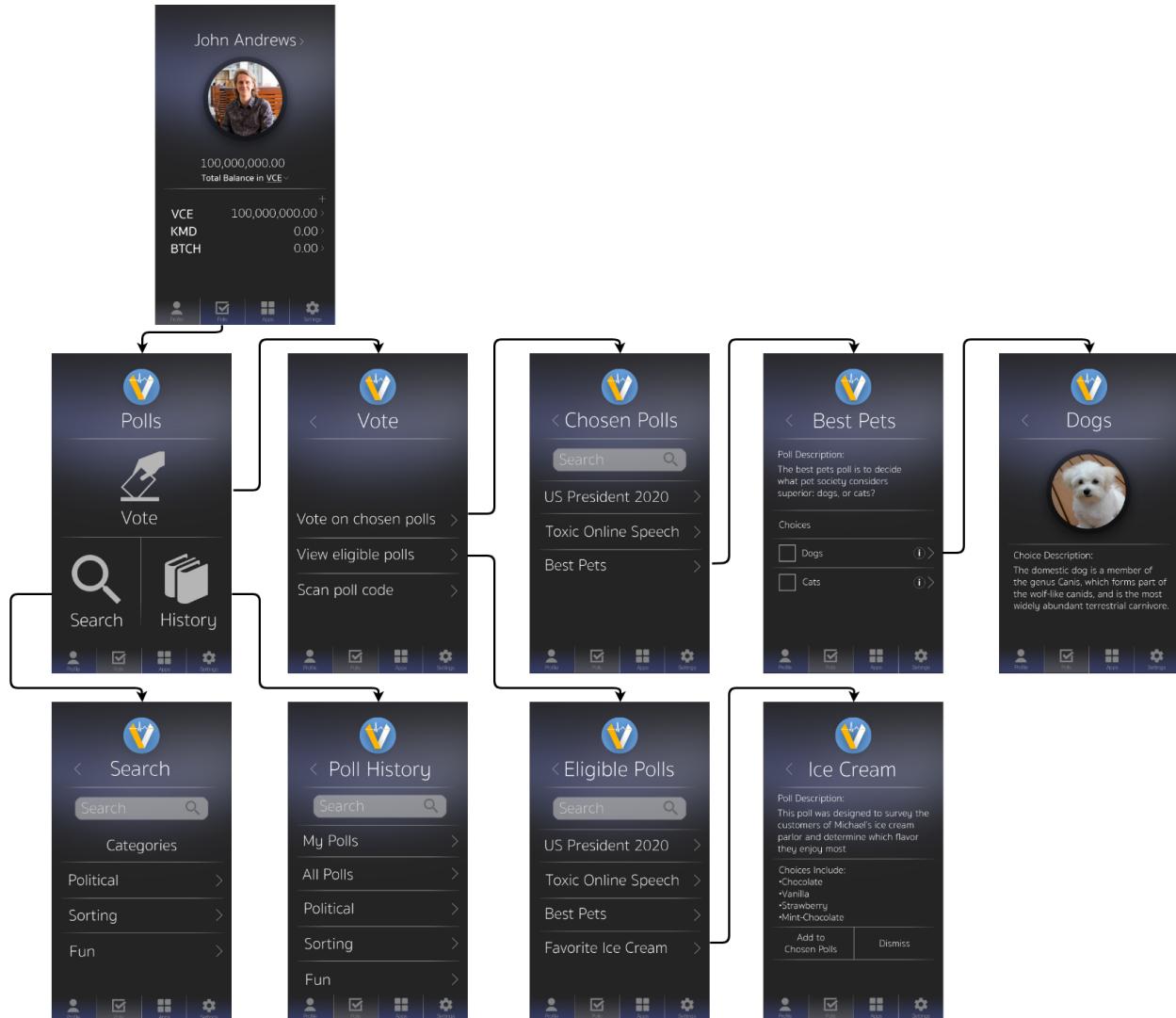
3.5 Machine Learning Integration

Verus enables humans to share and expose human knowledge through voting mechanisms to make decisions, express opinions, and classify accurately. The data used for that classification, as well as the classifications or decisions themselves provide an ideal source of training data for machine learning systems across any domain of human knowledge or opinion. As corporations gather, sequester, and learn from massive amounts of personal data in order to compete and boost their bottom line, rather than accrue to society's benefit, these massive private databases serve to affect and influence human populations by knowing more about them than others, or even than they know about themselves.

Since Verus enables humans to use voting mechanisms to classify and make decisions, and stores the results of this process in IPFS and on the blockchain, we will train machine learning algorithms on these results over time, allowing them to eventually classify and make decisions consistent with human decisions and values at an even higher scale. This will enable Verus to perform certain tasks automatically, such as the recognition of false or incorrect identification data among participants in the system, the initial classification or rating of certain content that can be overruled, but may not be appropriate for all audiences, and much more. The end result of this is a more secure and trustworthy network, built on consensus and trustless principles, leveraging worldwide human knowledge as a base of information and machines for scale.

A major Verus advantage over almost any other system in its use of Machine learning on human data is the innate privacy it provides to those whose data is used for learning, along with the default that all resulting information be made public, unless those making polls pay the Verus community a going rate to keep the data for themselves. In today's society, the goal of most machine learning systems is to match consumers with products they are likely to buy and/or manipulate them into an actual purchase, thus generating revenue. This creates a situation that can create unintended negative consequences when algorithms disregard any positive or negative effects its predictive abilities may have on society [23]. For example, if the algorithm recognizes that people with depressive episodes are more likely to gamble, and thus buy airline tickets to gambling-oriented locations, such as Las Vegas, it will advertise those airline tickets to those people. In the Verus system, machine learning algorithms will be able to learn from human beings, while being limited to accessing only the data users permit, and at the same time, being unable to easily target any specific individual.

3.6 Verus Mobile Polls



In addition to a desktop wallet that supports Verus and other Komodo platforms or compatible applications, mobile support is a high priority and will provide yet another layer of convenience for users, enabling them to use, earn, and spend Verus in everyday life. We intend to provide a Verus mobile experience that functions as a wallet, self-sovereign identity, provides easy access to the polling and earning applications, and is intended to serve as an extensible application browser, capable of supporting additional applications that leverage the Komodo or compatible ecosystems. While we have already begun thinking about and storyboarding design and development of user experiences, we do not expect any mobile experience to be ready until completion of **Phase II** at the earliest. Even when the first mobile experiences are ready, they will not include the support for private transactions that we intend to enable as a foundation for confidential, mobile polling and communications. To get a feel for the way we envision Verus experiences working in practice, below is an example of some poll screens from early designs.

4 Implementation and Roadmap

After researching options from building the technology ourselves to leveraging unreleased advanced projects, to using one of the few well thought through blockchain application platforms, we decided to start building the Verus project as a friendly fork of Komodo and its asset chain technology, enabling us to both become a contributing member of and also enhance and extend the Komodo platform as Verus builds blockchain platform technologies and real world applications of PBaaS.

By leveraging proven zk-SNARK zero knowledge privacy technologies and Komodo's delayed proof of work (dPoW), which notarizes the main Verus blockchain into Komodo's blockchain, which is then notarized into the Bitcoin blockchain, we underpin Verus with a foundation of state-of-the-art privacy, security, and interledger transaction capabilities as our baseline. Above that, we have already developed advanced features that further enhance security, improve decentralization, and prepare for the implementation of autochains and proof of virtue.

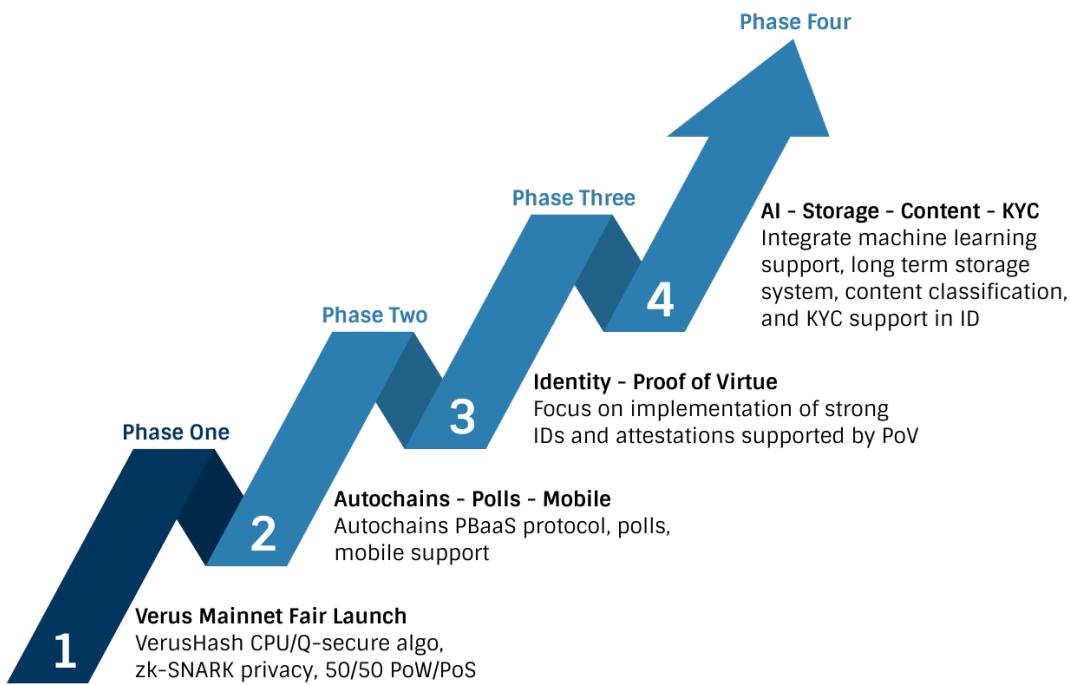
We will maintain our own open source fork of the Komodo platform and collaborate with the Komodo team when possible to develop new features or capabilities that can be accessed cross-chain by projects across the Komodo platform and even other blockchains that are simply compatible with Komodo's Interledger Protocol implementations. We will maintain our development as an open source project and contribute back to the broader open source community, as we leverage the contributions of others that have made it possible for us to begin realizing our vision as a community.

All that said, roadmaps are never perfect. Unless we significantly underpromise and prioritize a date so highly that no feature or capability is important in and of itself, targeting timeframes and milestones without specific target dates allows us to be ambitious in our goals and adapt to a changing world as we deliver. Even in **Phase I**, which we have delivered as of the writing of this paper, we adjusted our plan, and in an uncharacteristic turn of events, delivered even more than we had originally planned. We do not expect that to always be the case, and we will focus more on correct and complete phases than specific dates when possible. Sometimes, as with any major endeavor, the right choice is to recognize an

unexpected opportunity or obstacle, take two steps to the right or left, and only then proceed forward.

For the purpose of this whitepaper, we will discuss four phases of the Verus vision. We do not see these four phases as a completion of the vision as much as four phases of an ongoing mission to advance Verus and its contribution to society that we can currently express. From its inception and throughout the project, the Verus chain will serve as our fungible value chain, and use of Verus tokens on this chain will be the core value around which we continue to develop the Verus vision.

The first Verus chain, which is already available at the time of this white paper's release, includes zk-SNARKs for privacy, both transparent and private addresses, a brand new CPU-mineable hash algorithm for proof of work, a brand new proof of stake algorithm, and a unique emission schedule among fair launch cryptocurrency projects. Each phase of our project's development will introduce capabilities and experiences that provide independent value on their own, further leverage Verus Coin to power operation, and create a foundation upon which to build the next phase.



We intend and hope for the Verus project to become a worldwide, inclusive community effort, which welcomes and rewards those who contribute. Although we articulate these first four phases here, we see these phases as just the beginning, and we hope you will join us by participating in and contributing to the Verus project to make our world a better place.

4.1 Phase I – Mainnet – 50/50 PoW/PoS – Time Locks – Fair Launch

In phase one, the Verus main network began with a slow start and 15 minutes advance notice at 7:15am GMT, Monday, May 21st, 2018.

The Verus network began emitting first 0 block reward, rising linearly each block over the course of 7 days until block 10080 to its peak of 384 coins per block. The Verus emission schedule is as follows:

Era 1:

Block 0 - 10080 reward: 0 to 384, rising linearly and changing each block

Era 2:

Block 10080 - 53279 reward: 384
Block 53280 - 96479 reward: 192
Block 96480 - 139679 reward: 96
Block 139680 - 182879 reward: 48
Block 182880 - 226079 reward: 24

Era 3:

Block 226080 - 1277279 reward: 24
Block 1277280 - 2328479 reward: 12
Block 2328480 - 3379679 reward: 6
... halving indefinitely every 1051200 blocks (approximately 2 years)

We also added another fair twist on the launch that we believe will contribute to a more stable value growth in the Verus currency, without any unusual risk of dumping by any single party. During the first 5 months, Verus will have an accelerated reward curve for mining and staking, with a halving every month. During the first two months, when the block rewards are at or above 192 Verus, mined blocks will have time locked coinbase transactions, preventing spending, staking, or transferring of those coins for a period that varies from approximately 3 months after the genesis block, based on numbers of blocks, to 2 years and 3 months. These time locked coins provide a smooth release of the Verus currency supply as well as incentive to all of their owners to support the long term success of the Verus network and project.

From the very beginning of its operation, Verus operated with a dual proof of stake / proof of work mechanism for all participants. Verus mining with the VerusHash algorithm, as of this point, is a CPU-only algorithm, designed with a Haraka512 V2 [30] core to be quantum secure and to maximize performance on modern CPUs. While VerusHash was designed to be CPU-optimized and did not begin life with GPU or ASIC miners capable of beating CPUs in its mining, the Verus developers have no illusions that it is possible someone might develop either a GPU or ASIC-based miner that could be meaningfully superior to CPUs.

To ensure that such a solution if developed, remains open, jl777, lead developer of the Komodo Platform, has offered a 1 Bitcoin bounty to any developer who can make and publish in an open manner, a GPU miner for the VerusHash algorithm that can significantly outperform the current CPU miner. In order to win the bounty, the source code for the miner must be available under open source licensing. To be eligible for the bounty, any implementation must be able to perform 5x better than a modern, high thread-count, high clock-rate CPU on a GPU costing under \$1000. By bringing up the Verus network this way, we established an immediate, baseline set of functionality, above which we can build out our project and community around a functional coin and project, as we stay in sync with advancements in the underlying Komodo platform.

Verus phase one release was pre-announced on Bitcointalk [28] with zero premine, and team members mined and staked along with everyone else to generate coins. In addition to coins mined by individual team members for their own benefit, the Verus developers intend to donate most of their mined and staked earnings to a community Verus foundation along with other donating community members who will join us, in order to support the ongoing growth and project development by core developers and the community for years to come.

4.2 Phase Two – Autochains, Polls, Verus Mobile

As soon as we completed phase one, we entered phase two, which includes release of this white paper, activation of the community through donations of some of our mined coins, and a push to finish planning for, development and release of mobile support, autochain capability, and the ability to setup, and then easily run CVT polls at scale.

4.3 Phase Three – Identity, and Proof of Virtue

In phase three, we plan to further improve the mobile experiences, implement the identity system, including support for strong, decentralized identity and attestations, separating photo IDs and other photo content into components that can be separately verified in unbiased human polls. At this point, we intend to support optional KYC strong identities, via both notarization of identities as well as poll-based identity verification. This will also be the first phase that supports Verus chain validation PoV enhanced PoS.

4.4 Phase Four – Integrated Machine Learning, Content, Storage

In phase four, Verus will begin to truly leverage the foundation built in phases one, two, and three by focusing on broad content classification capabilities for off-chain content, improving storage management support in its distributed hash table implementations, supporting storage monetization in some way, and providing open source, public implementations of machine learning systems that can learn from data on the Verus network to solve real world challenges of today and tomorrow.

5 Forward Looking Statements

This paper includes predictions, statements of intent, discussion of plans, estimates or other information that might be considered forward-looking. While these forward-looking statements represent our judgment and expectation of what the future holds, this is not an offer or solicitation to purchase any product, good, service, or security. All statements herein are subject to risks and uncertainties that could cause actual results of the Verus Project's development to differ materially. Furthermore, we intend to use the Verus blockchain as our open source development platform – contributing these technologies under permissive licensing for the betterment of society, not focusing solely on profit of anyone affiliated with the Verus project. You are cautioned not to place undue reliance, especially in any financial decision, on these forward-looking statements, which are subject to modification, update, or change for legitimate reasons both within or beyond our control. By expressing our vision and goals, the Verus Core developers are not obligating ourselves to revise or publicly release the results of any revision to these forward-looking statements in light of new information or future events.

References

- [1] S. Estehghari and Y. Desmedt, Exploiting the client vulnerabilities in Inter-net e-voting systems: Hacking Helios 2.0 as an example, Proc. 2010 Electron. Voting , no. Section 4, pp. 027, 2010.
- [2] Kohno, Tadayoshi, et al. Analysis of an Electronic Voting System. IEEE Computer Society Press, 2004, Analysis of an Electronic Voting System, <http://avirubin.com/vote.pdf>
- [3] Hursti, Harri. “SECURITY ALERT: May 11, 2006 Critical Security Issues with Diebold TSx .” Black Box Voting, Black Box Voting, Inc, 11 May 2006, <http://www.blackboxvoting.org/BBVtsxstudy.pdf>
- [4] Diebold Election Systems. AVTSCE source tree, 2003. <http://users.actrix.co.nz/dolly/Vol2/cvs.tar>
- [5] Willet, JR, et al. “OmniLayer/Spec.” GitHub, JR Willet, 24 Jan. 2017, <http://github.com/OmniLayer/spec>
- [6] Stone, Andrew. “Bitcoin Cash Scripting Applications: Representative Tokens (OP_GROUP).” Medium, Medium, 16 Oct. 2017, <http://medium.com/@g.andrew.stone/bitcoin-scripting-applications-representative-tokens-ece42de81285>
- [7] Abed, Gabriel, et al. “Colu Local Network.” Colu Network, Colu Technologies DLT Limited, Jan. 2018, http://cln.network/pdf/cln_whitepaper.pdf
- [8] Assia, Yoni, et al. “Colored Coins White Paper - Digital Assets.” Google Docs, ColoredCoins, June 2017 , http://docs.google.com/document/d/1AnkP_cvZTCMLIzw4DvsW6M8Q2JC01IzrTLuoWu2z1BE/edit#heading=h.wxrvzqj8997r
- [9] “Ethereum Blockchain Congestion Triggers Outrage.” Global Coin Report, Global Coin Report, 12 Dec. 2017, <http://globalcoinreport.com/ethereum-blockchain-congestion-triggers-outrage/>
- [10] Sedgwick, Kai. “The Ethereum Blockchain Is Congested by Cats.” Bitcoin News, Bitcoin News, 4 Dec. 2017, <http://news.bitcoin.com/ethereum-blockchain-congested-cats/>
- [11] “Delegated Proof-of-Stake Consensus.” Delegated Proof-of-Stake Consensus - BitShares, Bitshare, 8 June 2015, <http://bitshares.org/technology/delegated-proof-of-stake-consensus/>
- [12] Buterin, Vitalik. “A Proof of Stake Design Philosophy – Vitalik Buterin – Medium.” Medium, Medium, 30 Dec. 2016, <http://medium.com/@VitalikButerin/a-proof-of-stake-design-philosophy-506585978d51>
- [13] Poelstra, Andrew. “On Stake and Consensus.” WP Software, 22 Mar. 2015, <http://download.wpsoftware.net/bitcoin/pos.pdf>

- [14] Nakamoto, Satoshi. "Bitcoin: A Peer-to-Peer Electronic Cash System." Bitcoin.org, <http://bitcoin.org/bitcoin.pdf>
- [15] Lee, James. "Komodo: An Advanced Blockchain Technology, Focused on Freedom." Komodo Platform, Komodo, 12 Mar. 2018, <http://komodoplatform.com/wp-content/uploads/2018/03/2018-03-12-Komodo-White-Paper-Full.pdf>
- [16] Mercer, David, and Duke Leto. "HushList Protocol Specification." Git Hub, <http://raw.githubusercontent.com/leto/hushlist/master/whitepaper/protocol.pdf>
- [17] Thomas, S., and R. Reginelli. "Crypto-Conditions." IETF Tools, 9 Jan. 2017, <http://tools.ietf.org/html/draft-thomas-crypto-conditions-02>
- [18] Maymounkov, Petar, and David Mazieres. "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric." Parallel & Distributed Operating Systems Group, <http://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf>
- [19] Benet, Juan. "IPFS - Content Addressed, Versioned, P2P File System." IPFS, <http://ipfs.io/ipfs/QmR7GSQM93Cx5eAg6a6yRzNde1FQv7uL6X1o4k7zrJa3LX/ipfs.draft3.pdf>
- [20] Kamvar, Sepandar D., et al. "The EigenTrust Algorithm for Reputation Management in P2P Networks." The Stanford Natural Language Processing Group, <http://nlp.stanford.edu/pubs/eigentrust.pdf>
- [21] "Some Things You Need to Know." Some Things You Need to Know - Bitcoin, 2015, <http://bitcoin.org/en/you-need-to-know>
- [22] "NEM Technical Reference." NEM, http://nem.io/NEM_techRef.pdf
- [23] Tufekci, Zeynep. "We're Building a Dystopia Just to Make People Click on Ads." TED: Ideas Worth Spreading, Sept. 2017, http://www.ted.com/talks/zeynep_tufekci_we_re_building_a_dystopia_just_to_make_people_click_on_ads
- [24] "IPFS js-ipfs Javascript Github Repository" <https://github.com/ipfs/js-ipfs>
- [25] Thomas S., et al. "A Protocol for Interledger Payments", <http://interledger.org/interledger.pdf>
- [26] Savoir-faire Linux Inc. "A C++ 11 Distributed Hash Table Implementation," <http://github.com/savoirfairelinux/opendht>
- [27] Goldstein, Jacob, and David Kestenbaum. "The Island of Stone Money." NPR, NPR, 10 Dec. 2010, <http://www.npr.org/sections/money/2011/02/15/131934618/the-island-of-stone-money>
- [28] Toutonghi, Michael. "[ANN] Verus (VRSC) - Zk-SNARK Privacy, CPU-Mining, 50/50 POW/POS, Fair Launch." Bitcoin Forum, Simple Machines Forum, 21 May 2018, <http://bitcointalk.org/index.php?topic=4070404.0>

- [29] Gilad, Yossi, et al. “Algorand: Scaling Byzantine Agreements for Cryptocurrencies.” MIT CSAIL, 24 Sept. 2017.
- [30] Klbl, Stefan, et al. “Haraka v2 – Efficient Short-Input Hashing for Post-Quantum Applications.” International Association for Cryptologic Research, 24 Dec. 2016.