Web3 and Polkadot

Back in the early 2000's the internet featured read-only, static, basic web pages. The online connected world at the time was only the beginning of virtual data, identities, and more. The internet during this time can be viewed as its first version (Web1).

As social media platforms and online businesses began to emerge, the internet transformed into its next iteration - the Web2. This upgraded internet, which we use today, features dynamic, interactive web pages, where users can read and write information and publish their own for others to see. However, this version of the web comes with downsides, dealing with data control, privacy issues, and the consequences of trusting centralized entities storing our data on their servers. This is where Web3 comes into the picture.

Web3 is transforming applications hosted on centralized infrastructure into decentralized applications (dApps) powered by trust-free blockchain protocols. The goal is to transform the internet into a decentralized web, where users control their data and identity in a trust-free environment. The Web3 movement aims to remove intermediaries and build trustless infrastructure. Web3 is an interactive and collaborative web where users can read, write, and own data.
THE WEB3 MOVEMENT

To learn more about the Web3 movement, check out this video from the Web3 Summit

Data Ownership

In web3, ownership is achieved and validated through cryptography. Each user has a digital identity bound to a set of cryptographic keys usually based on the public key cryptographic scheme, i.e., the famous public and private key pair.

Unlike Web2 which is driven by email IDs, phone numbers, and passwords, users onboarding to Web3 just need to generate a key pair. The public key can be the identity that can be shared with anybody to send you messages or assets, while the private key is used to access your account, sign messages, transfer funds, edit identity details, etc. Keeping your private key secure is essential to avoid identity theft or consequent loss of funds. Currently, scams are one of the main factors hindering web3 adoption. No legitimate person or entity will ever ask you to share your private key, and those who attempt to do so are likely trying to steal your digital identity and anything you own related to it.

To mitigate risks of key mismanagement (for non-custodial accounts, i.e. when you have custody of your keys) there are account abstraction solutions that separate the key management from the user experience. To mitigate key hacks, there are cold wallet solutions where the private key is generated and stored on dedicated devices with secure elements that are not exposed to the internet (see Ledger), or dedicated applications that can be installed on air-gapped devices such as phones (see Polkadot Vault). For custodial accounts, you trust third parties to manage your keys and give you access whenever needed.

To summarize, data ownership comes from the fact that any message you sign with your private key comes from your digital identity, and the signature proof can be cryptographically verified. Unless someone else stole your keys, you and only you are held accountable for signing the messages and are responsible for the information on your account. Transferring an NFT between two accounts is essentially a transfer of ownership.

Trustless Environment

Cryptography also brings the possibility of building a trustless environment where we do not have to trust third parties, or have any relationship between the sender and receiver of a message. We do not need to trust centralized entities since we can verify who wrote the message and who owns what just by using cryptography. Trust is embedded in the code. Well-audited and reviewed code ultimately provides a solid, trustless environment.

Data Immutability

But what if the data we own can be easily modified or tampered with after they have been signed and stored?

Here is where blockchain technology plays an important role. Blockchain networks comprise of distributed state machines where increments of data are stored within blocks that build on each other using hash functions. For example, the hash of block $N + 1$ contains data of that block together with the hash of the previous block N. This creates the situation where if you modify the content of block N you will change the hash of block $N + 1$, $N + 2$, etc. essentially breaking the chain. Although it can be possible to add an invalid block (a block with invalid transactions) or censor certain transactions, if the blockchain network is nor sufficiently decentralized. In decentralized proof-of-stake blockchains like Polkadot such attacks are financially expensive, and attempting to do it can get you slashed.

So, with blockchain as a means of storing data and transactions permanently without an option to modify them, we can ensure what we cryptographically sign with our digital identity is set in stone digitally.

Data Retrievability

But what if our data are stored in a blockchain, but that blockchain is run on a centralized server or by different computers belonging to the same operator?

That server or those computers can be easily shut down, the blockchain can be stopped from running and its data wiped out. This can be achieved from the inside by the malicious network participants or from the outside by regulatory rules and other forces. Though blockchain offers immutability, there would be little sense in using a centralized blockchain to prove ownership as it can possibly cease to exist in the future.

Data retrievability is dependent on how resilient the blockchain is. Resiliency is achieved through elements such as decentralization, economic incentives, and on-chain governance to ensure the network can sustain on its own.
DATA RETRIEVABILITY VS. DATA AVAILABILITY

Data retrievability is the ability of nodes to retrieve historical information from the blockchain. Historical data is not needed to verify new blocks; it is only required for synching full nodes from the Genesis block or serving specific historical requests.

Data availability assures full nodes can access and verify the full transactions associated with a specific block. It does not necessarily imply that the data is accessible forever. For more information about data availability on Polkadot, see the dedicated section on the parachain protocol page.

Decentralization

Having multiple nodes belonging to numerous independent identities increases network resiliency and thus data retrievability.

Blockchain is a state machine, and consensus must be achieved on every single state transition by every node on the blockchain network. In Proof of Work (PoW) based blockchains, which let any node in the network produce a block, consensus is achieved probabilistically by building on the longest chain (at the cost of energy-intensive computations). Proof of Stake (PoS) based blockchains like Polkadot enable deterministic consensus by allowing only a limited number of privileged nodes to produce blocks. A PoW blockchain can be considered centralized if a single entity can capture 51% of network nodes. Similarly, a PoS blockchain can be considered

centralized if a single entity controls more than one-third of nodes, as a two-thirds majority is required to arrive at a deterministic consensus. Different blockchains have different levels of decentralization.

Nowadays, most of the nodes cannot be run on consumer-grade hardware. Node running equipment is typically rented through service providers. Resiliency is also achieved by ensuring nodes run on as many different providers as possible and avoiding a significant share of the nodes being run under the same provider in the same geographic region. A legislation change or a natural disaster could impact a considerable fraction of the nodes and potentially stop the network. Pokadot level of decentralization can be explored through the Polkawatch app.

The One Thousand Validator Programme aims to incentivize the creation of new validator nodes to increase the level of node decentralization.

Decentralized Storage

Blockspace is limited and valuable. Not all data we have can be stored on the blockchain. Large files like pictures, music, movies, etc., typically will never be held on the blockchain. But where can we stored those files? To stick to the web3 vision, we need a resilient and decentralized storage solution.

The most important thing is that the proof of ownership is stored on the blockchain through the hashes of data and metadata. The files are uploaded on decentralized storage networks hosting protocols like IPFS.

Stake Allocation

In Proof-of-Stake blockchains, security is dictated by how much stake is locked on-chain (financial security). In a decentralized network, you want to ensure that the difficulty level for a financial attack to happen is equally difficult across all nodes. Polkadot's election algorithm makes sure that the stake is maximized across all active validators, and the variance in stake across validators is minimized as much as possible.

Economic Incentives

Strong incentives are essential to incentivize network participants to run nodes and secure the network. Strong incentives are possible because blockchain is a trustless system where there are no intermediaries between who sends a message and who receives it. Such incentives, coupled with punishment for bad behavior, ensure that most of the participants make the interest of the network and work together to improve it.

But from where are those incentives coming from? Polkadot's native token DOT is inflationary. Inflation is used to pay validators for running nodes and reward nominators for providing the necessary stake to secure the network. Depending on the staking rate, part of the inflation is diverted to the treasury.

Governance and Treasury

In Polkadot an on-chain treasury together with an open governance model allow to access funds in a fully decentralized manner without any bank transaction whatsoever. This opens up the possibility to come to a decision through on-chain voting mechanism, promoting a sense of community and creating an independent socio-economical environment.

Decentralized Access Points

But what if we have data we own stored on a resilient blockchain, but the only way to access the blockchain is through an RPC server? Whoever is behind the server or an attacker could present us data that is not the truth. How can we trustlessly verify that the data is true?

Here is where light clients play a key role. Light clients are clients that can sit on a web browser and can fetch data directly from blockchain. The figure below shows the architectural difference between web2 and web3 applications.

In web2 applications, data are stored on a centralized server, while in web3 applications, data (or better data proofs) are stored on the blockchain. With light clients, it is possible to access blockchain data through a full node and verify the validity of such data. They efficiently synchronize (*warp sync* in case of Polkadot) with a full node to obtain (Merkle Root) commitment of the latest chain state, and hence can trustlessly verify any response by full node against the commitment. In this way, we can always verify that the data we see is the truth, which is done automatically by the light client. Polkadot has a browser-embedded light client Substrate connect that uses the smoldot codebase. Most web3 applications today access blockchain data through a centralized RPC server.

Interoperability

The Web3 landscape's expansion into a multi-layered ecosystem highlights the need for interoperability. Blockchains compete and differentiate themselves based on decentralization, throughput, and specific use case focus. Some aim for a single high-performance base-layer blockchain, while others focus on decentralization through

layer-2 networks. With such diverse approaches, it's crucial for distinct on-chain environments to interoperate, especially for developers building cross-chain applications and traditional systems interacting with multiple blockchains.

Various cross-chain interactions are employed to achieve interoperability, including token swaps, token bridges, native payments, contract calls, and programmable token bridges. Each mechanism serves specific functions, such as facilitating the exchange of tokens between different blockchains or enabling smart contract interactions across chains. Other interoperability solutions validate the state of a source blockchain and relay transactions to the destination blockchain, which is essential for completing cross-chain interactions.

Polkadot 1.0

Since the release of Bitcoin in 2009, blockchain projects increased exponentially to the order of tens of thousands. Different projects have different value propositions, suggesting that the future will be multi-chain and that inter-chain communication will be crucial to establish collaborations and leveraging each other strengths.

Polkadot 1.0

Polkadot 1.0 reflects the status of Polkadot in 2023 at time of the release of the Polkadot runtime v1.0.0. This sections focuses on Polkadot 1.0 and some philosophical digressions about network resilience and blockspace.

Polkadot is a Layer-0 blockchain that brings to the multi-chain vision the following innovations and initiatives:

- Application-specific Layer-1 (L1) blockchains (or parachains). Polkadot is a sharded network where transactions are processed in parallel with each shard. Polkadot shards can be heterogenous (i.e. they do not need the same state transition function as in the proposed Ethereum sharding architecture). This allows to build L1 chains designed explicitly around their application and value proposition.
- Shared security and financial scalability of L1 chains. Any L1 chain attached to a Polkadot core can benefit from Polkadot shared security model. This means the Polkadot Nominated-Proof-of-Stake (NPoS) mechanism along with its consensus

mechanism, will secure L1 chains out-of-the-box without having to bootstrap security on their own.
- Secure interoperability. Any L1 chain attached to Polkadot (as well as L2 chains built on top of them) can benefit from Polkadot's native interoperability and will thus be able to communicate and exchange value and information with other parachains.
- Truly resilient infrastructure. This is achieved by keeping the network decentralized without compromising scalability and throughput and through on-chain treasury funds that can be accessed through governance referendum. Those funds guarantee constant sponsorship for events, initiatives, educational material, education, software development, etc.
- Fast development and deployment of L1 chains. This is achieved through the modular and flexible Polkadot SDK Substrate.
- Fostering next-gen of Web3 core developers. This is achieved through different initiatives such as:
    - The Polkadot Blockchain Academy
    - Substrate Builders Program
    - Polkadot Developer Heroes Program
    - Edx Courses
    - Rust and Substrate Courses (coming soon)

Polkadot's Representation

Polkadot has a Relay Chain acting as the main chain of the system. The Polkadot relay chain has been represented as a ring surrounded by multiple parachains attached to it. Based on Polkadot's design, as long as a chain's logic can compile to Wasm and adheres to the Relay Chain API, then it can connect to the Polkadot network as a parachain.

Parachains construct and propose blocks to validators on the Relay Chain, where the blocks undergo rigorous availability and validity checks before being added to the finalized chain. As the Relay Chain provides the security guarantees, collators - full nodes of these parachains - don't have any security responsibilities, and thus do not require a robust incentive system. This is how the entire network stays up to date with the many transactions that take place.

The Cross-Consensus Messaging Format (XCM) allows parachains to send messages of any type to each other. The shared security and validation logic of the Relay Chain

provide the environment for trust-free message passing that opens up true interoperability.

In order to interact with chains that want to use their own finalization process (e.g. Bitcoin), Polkadot has bridge parachains that offer two-way compatibility, meaning that transactions can be made between different parachains.

Polkadot's Additional Functionalities

The Polkadot relay-chain also manges crowdloans, auctions, staking, accounts, balances, and governance. Parachain slots or cores are leased in 6-month chunks for a maximum of two years, and crowdloans allow users to trustlessly loan funds to teams for lease deposits in exchange for pre-sale tokens. There is no other way you could use Polkadot 1.0.

Polkadot's Resilience

Decentralization is a crucial aspect of blockchain networks, but there is a trade-off between:

- having an over-decentralized network that struggles to reach consensus and consumes a lot of energy to operate, and
- having a network that reaches consensus fast at the expense of being centralized, making it trivial to manipulate or attack.

Ideally, a network should be decentralized "enough" to make it practically impossible for someone to exert manipulative or malicious influence on the network. So, decentralization is a tool while the goal is resilience, which is achieved by additionally providing on-chain treasury and governance mechanism allowing continuous incentives for the network's participants without relying on intermediaries or centralized entities.

Currently, Polkadot 1.0 achieve resilience through the following strategies:

- Nominated Proof of Staking (NPoS) where the stake per validator is maximized and evenly distributed across validators.
- The 1KV program aims to incentivize new operators to become network participants and further increase physical (how many validator nodes per service provider) and social decentralization (how many validator nodes per operator). Those can be explored with the Polkawatch App.
- An on-chain treasury and governance (see: OpenGov where every decision goes though public referenda and any token holder can cast a vote.

Polkadot's Blockspace

The design and realization of Polkadot 1.0 allowed its creators to enable commoditization of blockspace.

A blockchain is a way to store data. The storage unit is the block, and once a block is finalized onto the chain, it is practically impossible to modify the data within that block. In addition to being tamper-proof, public permissionless blockchains like Polkadot store data that are visible to everybody (i.e. public), and anybody can become a network participant permissionlessly.

Blockspace is the capacity of a blockchain to finalize and commit operations. It represents a blockchain's security, computing, and storage capability as an end product. Blockspace produced by different blockchains can vary in security, flexibility, and availability.

- Security, intended as how secure the blockspace is. In Proof-of-Stake (PoS) networks, this is directly related to how much stake is locked on validator nodes, how much variance in stake there is between validators (i.e. how easy it is to attack a single validator), and how many validators there are securing the network (i.e. how easy it is for colluding validators to exert influence on the network). Additionally, it is also important to look at how many validators are owned by single operators (this will determine the degree of social centralization of the network), and how many validators run on the same service provider (this will determine the degree of physical centralization of the network).
- Flexibility, intended as how flexible the blockspace is, what can be done with it, and what type of data can be stored. Data quality plays an important role depending on the type of network. One might avoid having situations in which poor quality data flood blockspace hindering the prompt execution of vital processes.
- Availability, intended as how available blockspace is and how difficult it is to access it. It should not be too difficult to get your hands on it so that any business model can thrive using it. Ideally, a marketplace must drive the blockspace price based on demand, with secondary market options to ensure the usage of "second-hand" blockspace.

Polkadot has been designed around those core blockspace principles. However, its design can be further improved such that the tasks which are currently managed on the relay chain, such as balances transfers, staking, and governance, can be delegated to system parachains to increase flexibility and to focus the use of the relay-chain to provide shared security and interoperability. Blockspace is only accessible through slot auctions, but an auction winner has access to a "freighter of blocks" regardless it is

needed or not. This creates high entry barriers and it can lead to waste of energy and resources.

For more information about blockspace the this interview to Robert Habermeier as well as this article by him.

A Perspective Shift: Upcoming Polkadot Features

As with many other projects before Polkadot, at some point in time after achieving the initially-planned goals, a perspective shift allows you to understand better what your project is about and what you actually have built. This allows you to "run the extra mile" and achieve more than what was originally planned.

The quote below by Marcel Proust must remind us that sometimes a perspective shift is crucial in *understanding* the world, and perhaps it is more important than *seeing more* of the world.

The only true voyage of discovery, the only fountain of Eternal Youth, would be not to visit strange lands but to possess other eyes.

Thus, if we start to see Polkadot with *other eyes* we can truly envision its potential and what it could become.

Polkadot is perfecting its implementation through RFCs to continue being a decentralized, secure, ubiquitous computing engine to power the next generation of Web3 applications.

Polkadot Direction

INFO

The material on this page is based on Gavin Wood's talk at Polkadot Decoded 2023.

Understanding what Polkadot 1.0 is about and the philosophy behind it will help us to envision the future direction of the Polkadot ecosystem toward abstraction and generalization.

Polkadot as a Computational Resource

Polkadot has been abstracted and generalized beyond what was originally proposed and envisioned in the whitepaper. Polkadot is:

- About Blockspace (the underlying resources that chains need), not chains.
- A platform to build applications rather than chains and for people to use those applications. Fundamentally, Polkadot is not a platform to host chains, and so far, chains happened to be one way to build applications and grow Polkadot's utility.
- A provider of resilient general-purpose continuation computation, where the term *continuation* refers to a broad, long-running task that can do something, pause, continue (or do something else) later.
- A multicore computer where chains that continuously operate in parallel on different cores are called parachains. Currently, one core is reserved for one chain through slot auction mechanism, although one core can be reserved on-demand to multiple chains at different periods (see parathreads). At the time of writing (mid 2023), there are around 50 cores independently operating in parallel on Polkadot.

From now on *application* will be used as a general term to describe anything that can use a Polkadot core to access secure and decentralized computation.

Summary

If we see Polkadot as a service provider of trustless and resilient computation through cores as well as secure interoperability between core-powered applications, the future development of Polkadot can be directed towards the following main changes.

A paradigm shift from:

- being a chain-focused ecosystem where each parachain owned an execution core at all times (acquired through fixed parachain slots), which allowed a simple and secure, sharded execution environment
- to being an application-focused ecosystem where we remove the assumption that each application owns a core, and instead that all cores are a resource to be consumed and used as needed by all applications.

Previously, securing a parachain slot was a competitive process through an auction mechanism. With coretime rental, there is no need for slot auctions anymore. Teams can purchase instantaneous coretime or reserve bulk coretime as required. This greatly decreases the barrier-to-entry for software tinkerers and parachain teams.

On top of those main changes, agile core usage and coretime allocation will allow any application to access Polkadot's computation based on their needs without wasting

valuable blockspace. Accords will improve cross-chain communication and the security guarantees of XCM messages. Finally, Polkadot will scale by moving on-chain logic into its system parachains, allowing it to have more bandwidth for the parachains protocol and accords.

From Slot Auctions to Coretime Marketplace

The end product of blockchains is Blockspace. Applications need to access Polkadot's blockspace, and the entry points to blockspace are the cores. Thus, applications will need to reserve some time on cores or Coretime to gain the right to access Polkadot's secure blockspace and interoperability for a finite period.

Cores must be agile and general: they can change what job they run as easily as a modern CPU. It follows that the procurement of those cores must be agile as well.

The slot auction mechanism is not agile, creates high entry barriers, and is designed for long-running single applications (i.e., the original Polkadot vision proposed in the whitepaper).

We depart from the classic lease auctions and propose an agile marketplace for coretime, where essentially coretime becomes a commodity that can be tokenized, sold, and traded. This setup maximizes the agility of Polkadot and lets the market figure out the best solution needed for applications to be successful.

Applications can reserve bulk coretime and instantaneous coretime depending on their needs. Bulk coretime rental will be a standard rental of coretime through a broker system parachain at a fixed price for a fixed period of time. Instantaneous coretime rental will be available through ongoing sale of coretime for immediate use at a spot price. This system lowers the barrier to entry for prospective builders.

For example, revenues from coretime sales can be burnt, used to fund the Treasury, or used for a mix of those options. The topic is currently under discussion. For more information, see RFC-0010 and RFC-0015.

From Chain- to Application-centricity

Polkadot 1.0 was a chain-centric paradigm consisting of isolated chains able to exchange messages. This was not fundamentally different from having completely different chains connected to bridges, with the only difference of having the relay-chain securing the network, providing message-passing capability, and doing some extra

tasks such as crowdloans, auctions, staking, accounts, balances, and governance. Having a chain-centric system will ultimately end in chain-centric application and UX.

The true innovation of Polkadot is about leveraging the unique value proposition offered by different chains and using those chains' collaborative potential to build inter-chain applications to solve real-world problems. Those applications will thus need to span across chains.

Increasingly fewer tasks will be handled by the relay-chain that will focus efforts only on primary tasks: securing the network and providing secure message-passing capability. System parachains will be used to take over secondary relay-chain tasks such as staking, governance, etc.

XCM and Accords

XCMP is the transport layer for delivering XCM messages. It gives the transportation method and a secure route but not a framework for binding agreements.

XCM is a format, a language of intention abstract over functionality common within chains. It creates an expressive language of what you intend to do or want to happen. XCM messages are transported between different chains using XCMP. Ideally, in a fully trustless environment, strong guarantees ensure chains faithfully interpret XCM messages. We can have a secure mode of delivering messages that can be interpreted across protocols, but still messages might be misinterpreted. These guarantees can be achieved with accords.

An Accord is an *opt-in* treaty across many chains, where treaty logic cannot be changed or undermined by one or more of those chains, and Polkadot guarantees faithful execution of this logic. Accords will be specific to a particular function, and any chain that enters the accord will be held to it and will service that particular function. To lower the entry barrier, accords can be proposed permissionlessly, but because they are opt-in, the accord proposal will take effect until chains agree and sign up.

To sum up, accords ensure that the receiver faithfully interprets XCM messages securely sent via XCMP channels. Accords are the missing piece of the puzzle to achieve a fully trustless and collaborative environment between applications.

Polkadot is the only ecosystem where accords can properly exist because it has a homogenous security layer that provides a specific state transition function for each logic component. This allows patterns of cooperation between multiple logic components (i.e., trans-applications) that would not be possible to achieve over bridges.

Accords will be implemented using SPREE technology.

Core Usage in Polkadot 1.0

In Polkadot 1.0, applications produced blocks at a fixed rate of 12 seconds, whether needed or not. This led to inefficient energy allocation and economic incentives for producing full blocks under heavy traffic and empty blocks under light traffic.

The figure below shows the core usage for Polkadot 1.0, where the horizontal axis is time, and each row represents a core. Colors show different parachains, each using one core (i.e., one parachain, one core formula).

The above setup allowed a simple and secure, sharded execution environment.

However, to achieve full efficiency, blocks must be produced when needed, and the system must target full block capacity, lowering the probability of incentivizing validators to build blocks half full or, worse, empty.

Agile Coretime Allocation

In Polkadot 1.0, coretime is a fixed two-year period on one specific core. Here, we remove this limitation and generalize coretime usage to meet different application needs.

Split Coretime

Owners of coretime can split or trade it. An application A1 can run on core C1 for a finite period and then another application A2 can run on that core, or application A1 can continue running on another core C2. Some applications might stop running for some time and resume later on.

Strided Coretime

Ranges can be strided (i.e., applications can take turns on a core) to share costs or decrease block production rate, for example.

Combined Coretime

An application can be assigned to multiple cores simultaneously. Some applications can have a permanent core assignment and an intermittent one, for example, in a period of high demand to send multiple blocks to multiple cores at the same time slot to reduce latency.

Agile Core Usage

In Polkadot 1.0, one core is assigned to one application (in this case, equivalent to a parachain). Ideally, core affinity (i.e., which application operates on which core) is unimportant (see below). Cores do not have any higher friendliness to one application than another.

Here, we remove the assumption that each application owns a core and instead that all cores are a resource to be consumed and used as needed by all applications in the ecosystem.

Compressed Cores

The same core can secure multiple blocks of the same application simultaneously. Combining multiple application blocks in the same relay chain core will reduce latency at the expense of increased bandwidth for the fixed price of opening and closing a block.

Shared Cores

Sharing cores with other applications to share costs but with no reduction in latency. Note that this is different from the split coretime where one core is used by multiple application at different times to share costs at the expense of higher latency.

Agile Composable Computer

All the above options of agile coretime allocation and core usage can be composable and enable the creation of an agile decentralized global computing system.

Thus, this new vision is focused on Polkadot's resource, which is secure, flexible, and available blockspace that can be accessed by reserving some time on a core. Agility in allocating coretime and using cores allows for maximized network efficiency and blockspace usage.

Polkadot's Resilience

Systems that have yet to be engineered with decentralization, cryptography, and game theory in mind, are breakable and prone to cyber-attacks. Polkadot is basing its resilience on different pillars:

- Preponderance of light-client usage: Centralized RPC servers are common but susceptible to attack and not trustless decentralized entry points to using blockchain-based applications. Light client usage on Polkadot is possible through Smoldot.
- Zero-Knowledge (ZK) Primitives: They can have a problematic effect on censorship and centralization as having a big state transition function boiled down to a single proof of correct execution is not currently a scaling solution to build resilient systems. However, a library of richly featured and high-performance ZK primitives ready for specific use cases is being built. The first use-case will be used to improve privacy for on-chain collectives such as the Polkadot Technical Fellowship.
- Sassafras consensus: New forkless block-production consensus algorithm replacing BABE and where block are not produced unless they are expected to be finalized. This will provide several benefits, such as:
    - Improved security, parachain performance, and UX from being forkless
    - Preventing front-running attacks through high-performance transaction routing where transactions are included in blocks in one hop instead of being gossiped, and transaction encryption.
- Internode Mixnet: Shielded transport for short messages that
    - avoids leaking IP information for transactions, and
    - introduces a general messaging system allowing users, chains and off-chain workers, smart contracts, pallets, and anything else existing within a chain to exchange messages containing signatures, intentions, etc.

- Social Decentralization: Resilience is achieved by including many participants contributing to the system and coming to decisions through on-chain governance. Involving as many people as possible ensures resilience against spending becoming systemically misjudged and appropriately directs wealth for spending treasury funds, salaries, and grants. Another crucial way of decentralizing the network is ensuring experts on which the maintenance of the system relies upon are incentivized and recruited over time by the Polkadot network and not by organizations within the Polkadot ecosystem.

Frequently Asked Questions (FAQs)

INFO

This FAQ focuses on technical questions for users interested in developing applications for Polkadot. If you have a more general question, you may wish to search for the answer on our support Knowledge Base or the main Polkadot network FAQ. If you have a question that is not answered, please feel free to ask on the Polkadot Watercooler Element channel or contact Polkadot Support.

Polkadot Launch

The Genesis block of the Polkadot network was launched on May 26, 2020 at 15:36:21 UTC, as a Proof of Authority (PoA) network, with governance controlled by the single Sudo (super-user) account. During this time, validators started joining the network and signaling their intention to participate in consensus.

The network evolved to become a Proof of Stake (PoS) network on June 18, 2020. With the chain secured by the decentralized community of validators, the Sudo module was removed on July 20, 2020, transitioning the governance of the chain into the hands of the token (DOT) holders. This is the point where Polkadot became decentralized.

The final step of the transition to full-functioning Polkadot was the enabling of transfer functionality, which occurred on Polkadot at block number 1_205_128 on August 18, 2020, at 16:39 UTC.

On August 21, 2020, Redenomination of DOT occurred. From this date, one DOT (old) equals 100 new DOT.

Polkadot Roadmap

For more information on the Polkadot roadmap please visit the official Polkadot website.

Validators

How do I apply to be a validator?

There is no central authority that decides on validators, so there is not per se an *application* that you can fill out. Registering as a validator is permissionless; in order to become one you must only set up a validator node and mark your intention to validate on chain. For detailed instruction on how to do this you can consult the Kusama validator guide on validating for Kusama or the Polkadot validator guide for validating on Polkadot.

However, once you've set up a validator and have registered your intention it does not mean that you will be included in the *active set* right away. The validators are elected to the active set based on the results of an election algorithm known as Phragmén's method. Phragmén's method tries to accomplish two goals: 1) select n members from a larger set based on stake-weighted votes and 2) equalize the stake backing each validator as much as possible.

You will likely want to campaign your validator to the community in order to get more backing. You are looking for *nominators* that will put up their tokens to increase the stake for your validator. For validators who cannot acquire the minimum stake from the community, Parity and Web3 Foundation also run a joint program called Thousand Validators that will nominate validators if they apply and fit the requirements.

How are validators rewarded?

Validators are rewarded from the inflation of the Relay Chain, transaction fees, and tips. However, they only take a percentage of the former two. More details can be read on the page for validator payouts.

What is the minimum stake necessary to be elected as an active validator?

The minimum stake that is necessary to be elected as an active validator is dynamic and can change over time. It depends not only on how much stake is being put behind each validator, but also the size of the active set and how many validators are waiting in the pool.

There are a few ways to estimate the minimum stake.

One way can be to navigate to the Polkadot Apps Targets tab. The value at the top of the screen saying "Lowest" is the least staked validator. You need at least this much + 1 to enter the set.

You can also use some tools some to perform estimations.

- Offline Election can provide exact results of running an election on the current set of validators using the same Rust code that is ran in Polkadot.
- Validator stats script can give you an estimate that is based on the currently elected set, as well as some statistics about Kusama validators.

Why will Polkadot have only 1000 validators while other projects have hundreds of thousands?

Polkadot's goal to have 1000 validators is set to be something that is practically achievable in the short term with high confidence of good performance in a live environment. Furthermore, validators in Polkadot are not the only stakers, and if we consider the number of stakers that can be possible on Polkadot the number can scale up to hundreds of thousands. Since validators are performing critical consensus work to maintain the security of the chain including all of its shards, a more modest number of validators is estimated to start. Upon later improvements, such as implementing signature aggregation for finalization messages, the number of validators could reasonably scale up. However, increasing validators above one thousand remains a goal for later iterations of Polkadot.

It is also worth mentioning that one thousand validators is more than the number of validators of similar PoS chains with comparable levels of economic security as Polkadot. The closest contenders are operating with around 150 validators, while Polkadot is already securely running with 297 validators.

Additionally, other projects sometimes have a different definition of *validator* that approximates more closely to remote signing keys without the full operation of a validating node. On Polkadot, each validator is running their own validating node and performing full verification of the Relay Chain, voting on finality, producing blocks in their decided slots, and verifying parachain state transitions. Other projects may consider validators and "validating nodes" as separate entities.

Finally, individuals may participate in the block production process indirectly by nominating validators. In this way, individuals who are not running a node can still share in staking rewards.

Relay Chain

What is the block time of the Relay Chain?

Both the Kusama and Polkadot networks are currently operating at a rate of one block every six seconds.

This may be changed in the future. It may go as low as two to three seconds after optimizations, or potentially increase in order to handle the capacity of the parachain networking in a live environment.

Does Polkadot have smart contracts?

No - and yes. The Polkadot Relay Chain does not implement smart contracts natively. The reason for not having smart contracts on the Relay Chain is part of the design philosophy for Polkadot that dictates that the Relay Chain should be the minimal logic required to accomplish its job.

However, Polkadot will be a platform for other chains that *do* implement smart contracts. It's possible for parachains to enable smart contract functionality and then benefit from the security and interoperability features of Polkadot. Additionally, existing smart contract chains can connect to Polkadot as a parachain, or via a bridge.

While the Polkadot Relay Chain does not implement smart contracts directly, undoubtedly there will be parachains that do. So it's better to say that the Polkadot *ecosystem* has smart contracts versus "Polkadot has smart contracts."

How will the Polkadot Relay Chain connect to external chains in the ecosystem?

One of the cornerstone interoperability technologies being researched and developed for deployment on Polkadot is cross-chain bridges. Bridges come in a variety of flavors with varying levels of trust associated with them. Polkadot is predominantly researching the trust-minimized flavor that imposes economic costs on the operators of the bridge, and therefore makes it economically secure. Bridge efforts are being worked on in concert with other projects in the ecosystem. Eventually, there will be bridges between Polkadot and most of the other major chains.

What is Polkadot's Transactions Per Second (TPS)?

Polkadot is a heterogeneous sharded network comprising a relay chain and numerous parachains, which are all individual blockchains built on Substrate executing in parallel.

Hence, the Transactions Per Second (TPS) of Polkadot is a number that encompasses all the transactions on the relay chain as well as parachains. As the transactions on these Substrate-based blockchains are weights based, it makes sense to use TPS as a measure for the network performance if all the transactions carry the same weight. Performance benchmark tests show that Substrate-based blockchains can achieve over 1000 TPS for balance transfer transactions. Assuming Polkadot is running over 100 parachains; the projected TPS is well over 100,000. With asynchronous backing upgrade, the TPS is expected to increase tenfold to 1,000,000.

It is essential to realize that TPS is inherently a subjective measurement with numerous factors that can contribute to it. It's hard to gauge the usefulness of TPS in isolation (when compared to other chains), as it depends on what a transaction does for a particular network. To view how Polkadot measures TPS see the Polkadot sTPS (Standard Transaction Per Second) to consider precisely how benchmarking was performed for Polkadot.

DOT

What is the difference between DOT (old) and new DOT?

The DOT (old) unit on Polkadot was at twelve decimal places, otherwise known as 1e12 Plancks. On 21 August, 2020, Denomination Day, the DOT (old) value was redenominated to 1e10 (10_000_000_000, or ten billion) Plancks, meaning that the new DOT was valued at ten decimal places. Following the redenomination, the new DOT is called DOT.

What is the inflation rate of the DOT?

The inflation rate is approximately 10% per year.

A portion of the inflation is rewarded to validators for performing their duties, while another portion may go directly to the treasury. The exact percentage that goes into both varies and is based on the amount of DOT that are staked. Please see the article on inflation for more information.

Why can't crowdloaned DOT be staked?

DOTs contributed to a successful crowdloan campaign by a parachain are bonded for the entire lease period, which is two years on Polkadot. The crowdloaned DOT cannot be used for any other DOT utility functionalities like staking and democracy. In

exchange to the lost staking rewards or liquidity of DOTs, the parachain team may offer rewards to the contributor.

The utility of crowdloaned DOT is to provide a lease for a parachain. The utility of staked DOT is to secure the network through a reward/slash mechanism. Allowing crowdloaned DOT to be staked results in complex consequences like applying a slash on crowdloaned DOT that was meant to be bonded for the entire lease period of a parachain. In a way, the inaccessibility of crowdloaned DOTs and the lack of staking rewards for the entire lease duration encourages the contributors to back projects that are valuable to the ecosystem.

Governance

What prevents Polkadot governance from failing?

Polkadot's governance has already been shown to work. Examples can be found in the runtime upgrades that have successfully taken place through on the testnets as well as in a real economic environment on Kusama and Polkadot itself.

It is fair to say that the field of on-chain blockchain governance is still new, and no one can claim to know exactly what the optimal version of on-chain governance is yet. However, Polkadot takes a brave step forward in pioneering thought-through mechanisms for evolving a blockchain.

Blockchains need a method to adapt and evolve. Therefore, an on-chain governance system was necessary for the long-term success of Polkadot. Ultimately, it is the token holders that are responsible for preventing Polkadot's governance from failing by using their economic value and conviction to sway the progression of the protocol.

What prevents Polkadot governance from becoming plutocratic?

A savvy reader might have noticed that the answer to the previous question endowed the token holder with the ultimate responsibility to ensure that Polkadot's governance does not fail. By following the train of this assertion, one might assume that Polkadot's governance is susceptible to becoming ruled by a few large token holders (called *whales* in trading parlance) and therefore become a mere plutocracy (rule of the rich).

There are several other mechanisms that are built-in to the governance system to resist this plutocratic tendency. One of these mechanisms is called conviction voting, and imbues greater voting power to token holders who are willing to lock their tokens on the protocol for longer lengths of time. Longer lock-ups display *conviction* in a vote.

Conviction voting could allow a highly determined minority to overrule the vote of an apathetic majority in certain situations. Another mechanism is known as Adaptive Quorum Biasing. This makes proposals have a varying threshold for approval or rejection based on what part of the governance protocol the proposal originated in. For details on the subtleties of Polkadot's governance system, please see the governance page.

Parachains

How do parachain economics work?

Parachains have the flexibility to implement their own monetary system or incentive structure for collators. However, this is not strictly necessary. Since the collator's job is to continue to give recent state transitions to the validators on the Relay Chain who validate each transition, the security of the parachain and the Polkadot network is completely separate from parachain economics. Parachains need collators to continue to progress, so it wouldn't be unreasonable to see them incentivize collator nodes in some way, but the specific mechanism is completely up to parachain implementers.

Are parachains ephemeral? What happens when a parachain loses the next auction?

Parachains are not ephemeral. As long as someone is keeping the data for a parachain, the parachain can move between being a parachain, a parathread, or a separate sovereign chain at different points of its lifetime. Especially with parathreads, parachains can be decommissioned to only produce blocks when their usage and throughput makes it necessary.

When a parachain loses an auction for renewal, that parachain has a few options. In most cases, becoming a parathread instead would be a suitable choice. Parathreads are still secured by the Relay Chain, but don't need to hold a parachain slot and can produce a block when its economically feasible for them. For more on parachains please see the parachains page and for more on parathreads see the parathreads page.

Networking

What is libp2p?

Libp2p is a modular and extensible networking stack that is used by IPFS, Substrate, and many other projects. It is a collection of peer-to-peer protocols for finding peers and connecting to them. Its modules have logic for content routing, peer routing, peer discovery, different transports, and NAT traversals. It is intended to be used by applications for building large scale peer-to-peer networks by only selecting the parts of the protocol suite that are needed.

The Rust implementation of the specification was built and primarily maintained by a team of contributors at Parity Technologies. The Go and JavaScript versions are maintained by Protocol Labs as well as community contributors. A Nim version of the library also exists. Libp2p as a whole is an open source project that is actively developed and expanded on various code repositories hosted on their GitHub.

Does Polkadot use libp2p?

Yes, since Polkadot is built with Substrate. Substrate uses a networking protocol that is based on libp2p (specifically the Rust libp2p library). However, Substrate uses a mix of standard libp2p protocols and protocols that are homegrown and not official libp2p standards. Of the standards protocols, those which are shared with other implementations of libp2p such as IPFS, are connection-checking (ping), asking for information on a peer (identity), and Kademlia random walks (kad).

Of the protocols that are custom to Substrate, there are the legacy Substrate stream, a request-response for getting information on blocks (sync), a light client protocol, a notification protocol for transactions, and block announcement. For detailed information on how Substrate uses libp2p and the standard and custom protocols, please see the networking documentation.

How does libp2p differ from IPFS?

The Interplanetary File System (IPFS) is a peer-to-peer hypermedia protocol used primarily for storage of files. It allows one to upload a file onto the network and share it with its content addressable URI. IPFS, like Substrate, is an application of libp2p and exists higher on the technology stack. Although both IPFS and Substrate use libp2p, it cannot be said that Substrate "uses" IPFS since besides sharing the underlying library for networking there is no native integration between the two applications.

Kusama

What is the minimum amount of KSM / DOT I can have in my account?

Please see information about Existential Deposits.

What are the transfer fees for Kusama?

It is important to note that the cost of transferring KSM is dynamic. Currently, the minimum cost of transferring KSM is 0.01 KSM (the base fee), although this can be changed via governance. However, actual transaction fees will vary based on a variety of factors. Specifically, fee calculation follows the following formula:
base_fee + (tx_length * length_fee) + WeightToFee(weight)

Please see the fee calculation page in the Substrate documentation for more detailed information.

Answered by Gav: Reason for using asynchronous rather than synchronous communication? Difference in terms of TPS?

First some terminology. Cross-chain communication has two paradigms: active and passive. They might also be named "mutable" and "immutable" or "push" and "pull" or "write" and "read". Active is where one chain, actually sends a message into another chain which would then have some effect in that receiving chain. Passive, on the other hand, is where one chain simply reads some recent information about some other chain and takes some action itself.

In wholly synchronous systems like Ethereum, both are synchronous. One smart contract sends a message to another, and is halted until any effects are caused and it receives any reply from them. Achieving wholly synchronous inter-chain communication is something that I threw out of the window 30 seconds into the design of Polkadot. For active communication, Polkadot's parallelisation means that it must be asynchronous, so token transfers between chains, for example, will need to be async.

For passive communication however, there is no such requirement needed. Given any witness data, Polkadot parachains will be able to interpret other parachains' recent states instantly and trustlessly.

Transaction throughput doesn't really come into it - it's all about the fact that it's an architecture built for large-scale parallelism from the ground-up.

Answered by Gav: Can I run multiple Validators with the same Session Key?

Gav@riot:polkadot watercooler:

you can absolutely run multiple validators with the same session key.

However, you need to absolutely ensure that only one is set to be validating at any given time. you can probably author a sophisticated script to ensure that this is the case.

However, the better way is to just to stay away from mainstream providers that many other validators might be using.

polkadot will slash coordinated offline-attacks heavily, but it will barely slash isolated incidents at all.

so as long as you can be sure that when you go down, few other validators will be down with you, then it should be fine

- (though you still don't want to go down often, ofc!)

given how unlikely it is for major cloud-providers to go offline, you might also just consider a paging service; if ever a watchdog detects your node has fallen offline, you get paged and can quickly (but manually) check and switch over to a second host. obviously, you'll need to be very careful to ensure that when the first node comes back up it doesn't resume validating.

polkadot consensus will allow for validators and possibly collators to know each others IP addresses, this is necessary for high-performance networking to function at all

if DoS is an issue then it'll need to be solved at the validator side.

in particular, you won't be able to assume that nobody will know the address(es) of you validator node(s)

Build on Polkadot

Request of Commit (RFC)

With the release of Polkadot runtime 1.0, Polkadot's codebase is in the hands of the community. Anyone can open a Request For Commit (RFC) to propose and discuss

changes to the network protocol, runtime logic, and public interfaces, and other technical matters.

To submit an RFC, follow the instructions here.

RFCs can only be approved and merged by III-Dan members of Polkadot Technical Fellowship via on-chain voting mechanism. Definitive approval or rejection is done by issuing the `RFC_APPROVE(xxxx, h)` or `RFC_REJECT(xxxx, h)` on-chain remark from the Fellowship origin on the Polkadot Collectives parachain, where `xxxx` is the RFC number and `h` is the hash of the raw proposal text.

For example, the first RFC RFC-1 about Agile Coretime was proposed by Gavin Wood on the 30th of June 2023 and merged on the 12th of August 2023. Subsequently, the code for the Agile Coretime Broker pallet was added to the Substrate FRAME system.

Polkadot SDK

INFO

For more information about building on Polkadot, see the Builder's Guide.

The Polkadot Software Development Kit (SDK) includes all the tools needed to build on the Polkadot ecosystem. The main repositories include:

- Implementation of a node for the Polkadot network in Rust, using the Substrate framework
- The Substrate SDK
- Cumulus - Parachain Development Kit

The programming language used for development is Rust.

Introduction

Polkadot consists of a main chain called the relay chain and multiple sharded chains called parachains. The relay chain is maintained by validators that are selected through the NPoS scheme and is responsible for producing blocks of the relay chain (via BABE) and keeping the state of all the parachains. These validators need to vote on the consensus, see GRANDPA, over all the parachains blocks. For parachains, there are additional actors called collators and fishermen that are responsible for parachain block

production and reporting invalid parachain blocks respectively. In the figure below an example cut-out of Polkadot with part of the relay chain, one parachain, three validators and five collators are shown.

Validators are assigned to parachains, which are responsible for validating parachain blockd and keeping them available via the A&V scheme. Moreover, another feature of Polkadot is enabling interchain messaging among parachains, called XCMP.

The security goal of Polkadot is to be Byzantine fault tolerant when the participants are rational. Rewards are given out when validators behave correctly and validators misbehaviour is punished via the Slashing mechanisms. More details on incentives and economics are reviewed here.

Furthermore, Polkadot has a decentralised governance scheme that can change any Polkadot design decisions and parameterisation. Details on low-level cryptographic primitives can be found here and Polkadot's networking schemes is in progress.

For other information regarding the project please refer to the wiki page.

We also provide implementation level specification of the protocol in the polkadot specification website.

High-level properties

This section describes properties (guarantees) that users can expect of Polkadot.

State Transition Properties

Polkadot speaking in abstract terms provides a number of connected finalising state machines. Connected means that a state transition of one machine can affect a transition of another machine. The state machines are final, since most networks participants agree on their state after some time. We would also like to enable adding, removing and changing of the state machines as the time goes on to ensure utility. The research focuses on how to enable having such publicly available system in the face of possible adversarial conditions. The public can use the system by interacting with state machines that they are interested in via the internet. Each state machine can provide different functionalities and behave in different ways (have a different state and state transition scheme).

So let us start with abstract state machines. A state machine has a certain state type and state transition type. As the time goes on, state transitions occur.

The data that determines the state transitions is structured as bundles of transactions - individual small state transitions triggered by the users of the system. Each bundle is called a block. In order to achieve its properties, ensures that those blocks are hash connected forming joint data structure.

1 Utility

Each state transition should bring some utility to the system participants. In order to ensure that this is the case:

- state machines should provide some utility to participants
- state transitions processed by these state machines reflect well the state transition needs of participants.

To ensure that the state machines provide utility we should ensure that there is a mechansim that enables participants to decide what state machines should be included and how they should change to reflect participant needs. This mechanism is the Polkadot governance scheme.

To ensure that useful state transitions are processed by those state machines, we will want to ensure that useful transactions get included in Polkadot blocks. Polkadot will have a transaction fee mechanism on the relay chain to ensure that transactions issued by parties willing to pay a reasonable price for them are included. There will also be a certain portion of each block that is dedicated to certain high-priority transactions, such as misbehaviour reporting. The utility of the parachain state transitions has to be ensured by the state transition function of a given chain.

2 Validity

The notion of validity in Polkadot is determined by a state transition validation function (STVF). Each chain in the ecosystem has to have one implemented. In order for all

nodes to be able to run this function it is being distributed as deterministic WebAssembly (Wasm) code which can be executed by the Polkadot Host.

The blocks are produced by parachain collators, then they get validated using the STVF by the subset of validators responsible for the given parachain to finally get included in the Polkadot Relay Chain. During this process validators, parachain collators and other parties are free to challenge claims of validity to trigger additional check, these parties are referred to as fishermen.

3 Finality

Finality of the Polkadot network state machines is achieved via a combination of a block production mechanism with eventual probabilistic consistency (BABE scheme) and GRANDPA finality gadget.

This approach allows for block production (thus transaction confirmations) to be fast, while allowing for as fast as possible economic finality with compact proofs.

4 Availability

In order for the critical data from all chains to remain reachable by users and subsequent block producers, Polkadot makes use of an erasure coding based availability scheme.

5 Messaging reliability

Besides ensuring all the above properties for all parachain, a crucial element of Polkadot is that these state machines are able to affect each others state transitions. This is done via the Cross-Chain Message Passing (XCMP) scheme.

6 Size

To ensure that the state transitions can be processed and stored by the network their size has to be reasonable. Mechanisms such as transaction fees and block limits are there to limit the storage size and computation required for each block.

Light client

The protocol is being designed with light client support in mind with existing Substrate implementation supporting one.

7 Bandwidth

To ensure usability in realistic network conditions a reasonable bandwidth requirements have to be maintained.

The Polkadot blockchain will implement nominated proof-of-stake (NPoS), a relatively new type of scheme used to select the validators who are allowed to participate in the consensus protocol. In this note we give an intro to NPoS, and a peek inside the research carried out at the Web3 Foundation. We also explain the peculiar way in which validators get elected. So how does NPoS work in Polkadot?

Validators and nominators

About once per day, the system elects a group of entities called validators, who in the next few hours will play a key role in highly sensitive protocols such as block production and the GRANDPA finality gadget. Their job is demanding as they need to run costly operations, ensure high communication responsiveness, and build a long-term reputation of reliability. They also must stake their DOTs, Polkadot's native token, as a guarantee of good behavior, and this stake gets slashed whenever they deviate from their protocol. In contrast, they get paid well when they play by the rules. Any node that is up to the task can publicly offer itself as a validator candidate. However, for operational reasons only a limited number of validators can be elected, expected to be hundreds or thousands.

The system also encourages any DOT holder to participate as a nominator. A nominator publishes a list of validator candidates that she trusts, and puts down an amount of DOTs at stake to support them with. If some of these candidates are elected as validators, she shares with them the payments, or the sanctions, on a per-staked-DOT basis. Unlike validators, an unlimited number of parties can participate as nominators. As long as a nominator is diligent in her choice and only supports validator candidates with good security practices, her role carries low risk and provides a continuous source of revenue. There are other special roles in the Polkadot network, but we focus only on the relation between these two roles.

The NPoS scheme

This nominator-validator arrangement gives strong security guarantees. It allows for the system to select validators with massive amounts of aggregate stake—much higher than any single party's DOT holdings—and eliminate candidates with low stake. In fact, at any given moment we expect there to be a considerable fraction of all the DOTs supply be staked in NPoS. This makes it very difficult for an adversarial entity to get validators elected (as they need to build a fair amount of reputation to get the required backing) and very costly to attack the system (because any attack will result in large amounts of DOTs being slashed).

Our NPoS scheme is much more efficient than proof-of-work (PoW) and faster than standard proof-of-stake (PoS). Networks with deterministic finality must have a limited validator set (the size can be changed with governance). NPoS allows for virtually all DOT holders to continuously participate, thus maintaining high levels of security by putting more value at stake and allowing more people to earn a yield based on their holdings.

The election process

How to elect the validators, given the nominators' votes? Unlike other PoS-based projects where validators are weighted by stake, Polkadot gives elected validators equal voting power in the consensus protocol. To reflect this fact, the nominators' stake should be distributed among the elected validators as evenly as possible, while still respecting the nominators' preferences. At the Web3 Foundation research team, we use tools ranging from election theory to game theory to discrete optimization, to develop an efficient election process that offers fair representation and security, and can be applied in the future to any blockchain using NPoS. We explore these objectives below, together with some examples.

Fair representation. In the late 19th century, Swedish mathematician Lars Edvard Phragmén proposed a method for electing members to his country's parliament. He noticed that the election methods at the time tended to give all the seats to the most popular political party; in contrast, his new method ensured that the number of seats assigned to each party were proportional to the votes given to them, so it gave more representation to minorities. The property achieved by his method is formally known as proportional justified representation, and is very fitting for the NPoS election because it ensures that any pool of nodes is neither over-represented nor under-represented by

the elected validators, proportional to their stake. Our heuristics build on top of Phragmén's suggested method and ensure this property in every election.

The illustration represents a typical input to the election process, with nominators on the left having different amounts of stake, and connected by lines to those validator candidates on the right that they trust (for simplicity, validators have no stake of their own in this example, though they will in a real scenario). Suppose we need to elect k = 4 validators. The fair representation property roughly translates to the rule that any nominator holding at least one k-th of the total stake is guaranteed to have at least one of their trusted validators elected. As the total stake is 40 DOTS and a fourth of it is 10 DOTS, the first two nominators are guaranteed to be represented by a validator. In the image below we see three possible election results: one that violates the fair representation property and two that achieve it.

Security. If a nominator gets two or more of its trusted validators elected, we need to distribute her stake among them, in such a way that the validators' backings are as balanced as possible. Recall that we want to make it as difficult as possible for an adversarial pool to get a validator elected, and they can achieve this only if they get a high enough backing. Therefore, we equate the level of security of an election result to *the minimum amount of backing of any elected validator*. For the last two election results with fair representation, we provide stake distributions which show that they achieve security levels of 6 and 9 respectively.

The election result on the right achieves a higher security level, and clearly does a better job at splitting the nominators' stake into validators' backings of roughly equal size. The goal of the NPoS election process is thus to provide a result that achieves fair representation and a security level that is as high as possible. This gives rise to a rather challenging optimization problem (it is NP-complete), for which we have developed fast approximate heuristics with strong guarantees on security and scalability.

We are excited about the technical developments brought forward by Polkadot, and the possibilities enabled by NPoS and other highly efficient schemes being developed in the blockchain space. To learn more about Nominated Proof-of-Stake, visit our Wiki pages, and read our research paper.

Block production

The relay chain in Polkadot is built with the underlying proof-of-stake (POS) block production mechanism by validators. The currently deployed mechanism is a hybrid of BABE and Aura. We plan to replace BABE+Aura with Sassafras in the future.

BABE: A PoS protocol provides a way to elect validators to produce a block in the corresponding time slot. BABE's election is based on verifiable random function (VRF) of validators invented by David et al. for Ouroboros Praos i.e., if a VRF output of a validator is less than a pre-defined threshold, then the validator is legitimate to produce a block. So, one validator or more than one validator or no validator can be elected. This election mechanism is completely private. In other words, no one can guess who is elected until the elected validator publishes a block. The privacy property is very critical for the blockchain security because it is indispensable for achieving security against an adaptive adversary who can corrupt any validator at any time. The drawback of this election mechanism is that no validator will be elected in a significant amount of time. So, validators waste these times by doing nothing which causes slightly worse (and uneven) throughput. Therefore, we fill the empty slots with blocks generated by validators who are deterministically selected by Aura. Aura's election mechanism is not private so it is not secure against an adaptive adversary. For example, the adversary can prepare a DDOS attack on the elected validator by Aura to prevent him to publish his block because the adversary knows who is elected beforehand. Therefore, filling the empty slots with Aura blocks is not a solution in the adaptive adversarial model to prevent empty slots. Nevertheless we note that BABE+Aura is secure (safe and live) in the adaptive adversarial model - the security reduces to the BABE's security. It just does not prevent theoretically to have empty slots that we need to have a better throughput in the adaptive adversarial model.

1. Overview

In Polkadot, we produce relay chain blocks using our Blind Assignment for Blockchain Extension protocol, abbreviated BABE. BABE assigns block production slots using roughly the randomness cycle from Ouroboros Praos [2].

In brief, all block producers have a verifiable random function (VRF) key, which they register with the locked stake. These VRFs produce secret randomness, which determines when they produce blocks. A priori, there is a risk that block producers could grind through VRF keys to bias results, so VRF inputs must include public randomness

created only after the VRF key. We therefore have epochs in which we create fresh public on-chain randomness by hashing together all the VRF outputs revealed in block creation during the epoch. In this way, we cycle between private but verifiable randomness and collaborative public randomness.

The main differences of BABE from Ouroboros Praos [2] are the best chain selection mechanism and slot synchronization assumption i.e.:

1. BABE's best chain selection is based on GRANDPA and longest chain.
2. Block producers in BABE do not have access to a central authority (e.g., Network Time Protocol (NTP)) to count slots instead, they construct their own clock to follow the slots.
3.

GRANDPA is the finality (consensus) algorithm for Polkadot. Here we first present a high-level overview, as an "extended abstract". Details are presented in the full paper directly below that.

We also have an alternative version of the full paper available on arxiv, which is more polished and a bit shorter.

What is implemented in the Polkadot software and deployed in practise, we refer to as "Polite GRANDPA" which includes optimisations required for efficient real-world performance in practise. These are not covered in the papers below for brevity, but we go into the details later here on this page. The high-level concepts and design principles remain the same as GRANDPA.