WTF is Web Token Forwarding? An Explanation of the Fastest and Most Versatile Blockchain-Native Protocol for Identity

Nanak Nihal Khalsa
myemail@somewhere.com

Lilly Hansen-Gillis
myemail@somewhere.com

Kinshuk Kashyap
myemail@somewhere.com

Kushal Kahar
myemail@somewhere.com

Jakub Smekal
myemail@somewhere.com

Shady El Damaty
myemail@somewhere.com

March 27, 2022

1 Introduction

We introduce decentralized verified credentials (DVCs) as opposed to verifiable credentials (VCs). DVCs promise to help onboard Web2 users to Web3 and ameliorate the unique and severe flaws of Web3 in its early stages. $10b has been lost with lost wallets, $3b has been lost in 2021 alone due to hacks [1], and roughly $1b of NFT sales on major NFT platforms are fraudulent [3, 2]. The WTF protocol can fix these problems with a novel verification solution. Furthermore, DVCs are key for in bringing on-chain incentives to creative works, such as scientific papers, books, songs, and videos. Organizations are currently implementing the WTF protocol to bring such incentives to decentralized science (DeSci). A guiding principle for the WTF protocol is removing single points of failure from both Web3 and Web2.

2 VCs vs. DVCs

Decentralized verified credentials reduce the burden of verification on both the individual and the verifier. This allows easier adoption from both users and dApps by substantially lowering the amount of work they must do to verify credentials. A key consequence is that smart contracts can use DVCs. They need not be verified off-chain; their existence on-chain is proof that they are verified.

3 Existing Approaches to Identity

Multiple approaches for identity in Web3 exist with unique capabilities and shortcomings. Some with radically different approaches include:

- Ceramic
- Social attestation networks, e.g. Proof of Humanity & BrightID
- On-chain resume / badge services, e.g. Rabbit Hole & Project Galaxy
- Centralized federated identity, e.g. Magic Link & traditional SSO
- Enterprise DID and SSI services, e.g. Evernym, BloomID, & Mattr
- Traditional KYC services, e.g. Blockpass & Stripe

Ceramic's advantage is scalable databases well-suited for storing identities. It supports verifiable credentials, but its method of verifiable credentials puts burden on the users and the verifiers. The WTF protocol's DVCs are interoperable with Ceramic, and later integrations are planned to further increase interoperability.

Social attestation networks such as Proof of Humanity and BrightID provide a handful benefits such as a Universal Basic Income token and Sybil resistance. However, they are game-theoretically insecure and thus cannot be used for serious applications where bad actors can reap benefits (an attack can cost only $10 [4]). They also require a significant amount of effort from both the users and verifiers, further preventing scaling. Although the protocols have demonstrated they are nonviable, they were important steps to sole a critical problem and should be lauded for their innovative efforts.

On-chain resume / badge services provide pseudonymous credentials. They effectively show accomplishments but they are limited to skill- and experience-related, rather than security- and identity-related uses.

Centralized Federated Identity achieves a unified, quasi-self-sovereign identity. Google SSOs are centralized and were not blockchain-compatible until being used with Web3Auth and the WTF protocol (both of which perform disparate tasks using Web2 SSOs. In fact, a Web3Auth+WTF protocol integration would enable powerful integrated onboarding, ID verification, and wallet recovery user experiences). An interesting case of centralized federated identity is Magic Link, which markets itself as decentralized but rather further centralizes Web2 SSOs by storing a key linked to all of them on AWS. This increases the single-point of failure for applications, opposite to the WTF protocol's ethos of security by decentralization. *Enterprise DID and SSI services* such as Evernym, BloomID, & Mattr provide some benefits of DIDs for organizations. However, they do not address any of the need for identity on decentralized blockchains like the WTF protocol.

In fact, they are fundamentally incompatible with such public blockchains without significant reliance on custom oracle solutions. They require a certain reliance on, and adoption of, their ecosystems, whereas we are baked into EVM blockchains which are already adopted.

Traditional KYC services provide deep checks into not just identity but also legal status. Efforts are being made to put these on-chain. We are integrating KYC service API responses with the WTF protocol to add them as one of many DVCs.

4 WTF Protocol: How it Works

The WTF protocol works by forwarding web tokens to smart contracts which verify their data is truly signed by the OpenID (or other asymmetrically-signed identity) provider. Single sign-on (SSO) services (login with Google et al. buttons) return JSON web tokens (JWTs) with RSA or HMAC signatures. RSA signatures used by Google, Facebook, ORCID, and others have the benefit of being verifiable by anyone. ECDSA signatures could also work and be far easier to implement on EVM chains but unfortunately, few servers sign JWTs with Elliptic Curve signatures.

RSA Signatures are verified by sending the JWTs to our smart contracts, which use the 0x05 modular exponentiation precompile to compare the signature to a padded hash of the data which is signed (in line with traditional RSA signature verification). For JWTs, the data to be hashed, padded, and signed is the readable aspect of the JWT, base64url encoded. Signature verification is similar to how all blockchain transactions work; they must be signed by the party transacting. Here, instead of checking that a transaction is signed by the sender, we check that credentials are signed by Google, Facebook etc. given their public keys. Thus, our contracts provide automated verifiable credentials, verified by the public key of the entity which issued them.

HMAC signatures, such as those returned by Twitter SSO, need a proxy to convert them to a blockchain-compatible signature. We provide Proxy Asymmetric Signing Service (PASS) for this. But relying entirely on PASS for all credentials would centralize the protocol to relying on the PASS server, so we recommend limiting use of PASS except when needed, e.g. for Twitter. We are also migrating PASS to a secure enclave so it can at least be private and trustless, despite requiring a centralized proxy. More people are encouraged to create PASS servers to reduce centralization. Readers are encouraged to reach out to us at wtfprotocol@gmail.com if they are interested in providing a PASS node and in exchange for WTF tokens and help setting up a node.

Once a blockchain-compatible asymmetric signature for credentials has been obtained, further steps are needed to proceed. A nave implementation

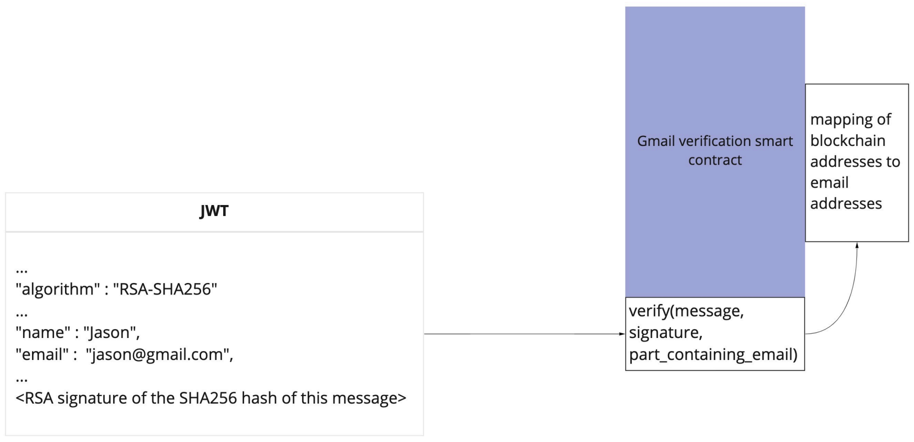
JWT
...
"algorithm" : "RSA-SHA256"
...
"name" : "Jason",
"email" : "jason@gmail.com",
...
<RSA signature of the SHA256 hash of this message>
Gmail verification smart contract
mapping of blockchain addresses to email addresses
verify(message, signature, part_containing_email)

Figure 1: JWT verification process. The mappings are bidirectional hashmaps which allow O(1) lookups by credentials or blockchain addresses

(checking the hash of the data against a an encrypted signature), would allow impersonation via frontrunning. The mempool would reveal a pending transaction, and one could sniff the JWT from it, replaying it in a transaction from ones own address. Thus, we employed a commit-reveal pattern for proving knowledge of a JWT amongst multiple blocks. This works in the following three-step process:

1. XOR hash(JWT) with public key, and submit the hash of that result
2. Wait for the current block to be finalized
3. Submit the plaintext JWT for the smart contract to verify it.

The smart contract can then XOR the JWT with the user's public key, then hash it. Then, it may check the result was not only known but also linked with the user's public key in a previous block. In this block, the JWT remained unknown to all but the user because it was (XORed and) hashed before it was shared. After these steps confirm a JWT belongs to the submitter of the transaction, its signature can then be verified as mentioned previously.

5 Underlying Security of this Protocol

A few questions come up in the storing of JWTs on-chain. This is a protocol in alpha-stage dealing with an uncharted territory of on-chain verification of web2 credentials. We do not recommend storing credentials for any sensitive accounts on-chain until past the alpha version, despite having the protocol being informally reviewed by security experts.

First is the question of whether users know blockchain data is public. We addressed this by clearly prompting users with the data that will be made public to ensure they consent to sharing it. We have thereby reduced the probability of user error to a negligible and tolerable amount.

Second is the question of whether revealing OpenID tokens can allow for impersonation of the user. We have a uniquely identifiable "aud" claim in the JWT so that no other website implementing OpenID will accept the tokens. Any secure website will reject JWTs submitted with "aud" claim that restricts their audience to the WTF protocol. Thus, revealed JWTs cannot be used to login to other websites that check JWTs. The only exceptions are atrociously insecure websites – no serious institution would allow these tokens. An analogy would be a bar that accepts Legoland Driver's license as a form of ID. Any service implementing the OpenID protocol only allows JWTs made for itself, not for Legoland.

6 Future Steps

The WTF protocol will enable far more than just verifiable credentials. We are working with the Lit Protocol to provide what may seem like magic at first: private credentials that are stored and verified on-chain, yet only provided to who the user grants on-chain permission to. This will be a critical step in decentralized self-sovereign identity, where users can control which aspects of their identity are public to which people, without relying on Web2 tech giants.

Securing the blockchain from its points of failure is our next use case. Decentralization is lauded as removing single points of failure. However, it is rarely mentioned that one of the biggest flaws in blockchain is that it adds two major points of failure: the user and the smart contract writers. The user can lose access to a seed phrase, or the smart contracts can be insecure. These cases happen often and cause tens of billions of dollars in losses and theft. The WTF protocol can be used to secure these. We are working again with the Lit protocol to provide decentralized seed phrase backup and recovery. We are also developing a similar, interoperable protocol called Web3 Two-Factor Authentication (WTFA) to provide decentralized, on-chain two-factor authentication. DeFi protocols and smart contract wallets could then use WTFA with just a single line of code to prevent unwanted transfers upon hacks. These uses are enabled by WTF protocol's multi-factor threshold proof of identity: proving your ownership of multiple Web2 accounts, and external 2FA devices such as phones or FIDO2 devices for the highest security, can be used as secure proof of identity.

7 Conclusion

The WTF protocol enables DVCs, a type of decentralized credential that can be used to increase blockchain adoption, reduce plagiarism of creative content such as NFTs, solve the pain points preventing protocols for decentralized science/music/ebooks/videos from being built, prevent damage from hacks, prevent losses of seed phrases, and enable truly self-sovereign and private credentials. We cannot tackle these simultaneously so are focusing on identity use-cases first. However, the protocol has promise to drastically increase the use-cases, security, and adoption of blockchain in general.

References

[1] Chainalysis. (2022, February). *The 2022 Crypto Crime Report.* Chainalysis. Retrieved March 27, 2022, from `https://go.chainalysis.com/rs/503-FAP-074/images/Crypto-Crime-Report-2022.pdf`

[2] @rchen8 (2022, March 27). Dune Analytics. *OpenSea.* Retrieved March 27, 2022, from `https://dune.xyz/rchen8/opensea`

[3] OpenSea. (2022, January 27). *OpenSea Tweet.* Twitter. Retrieved March 27, 2022, from `https://twitter.com/opensea/status/1486843201352716289`

[4] @RoboTeddy. (2021, June). *Proof of humanity: The cost of attack.* HackMD. Retrieved June 2021, from `https://hackmd.io/@RoboTeddy/SkFEYwptd`