WTF is Web Token Forwarding? A Brief Explanation of the Protocol

Nanak Nihal Khalsa
email myemail@somewhere.com

Lilly Hansen-Gillis
email myemail@somewhere.com

Kinshuk Kashyap
email myemail@somewhere.com

Caleb Tuttle
email myemail@somewhere.com

Kushal Kahar
email myemail@somewhere.com

Shady El Damaty
email myemail@somewhere.com

March 14, 2022

1 Introduction

We introduce decentralized verified credentials (DVCs) as opposed to verifiable credentials (VCs). Verified credentials promise to help onboard Web2 users to Web3 and ameliorate the unique and severe flaws of Web3 in its early stages. $10bn has been lost with lost wallets, $3 billion has been lost in 2021 alone due to hacks [1], and roughly $1bn of NFT sales on major NFT platforms are fraudulent [3, 2]. The WTF protocol can fix these problems with a novel verification solution. Furthermore, Verified credentials are key for in bringing crypto-economic 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 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 vital consequence is that smart contracts can use DVCs. They need not be verified off-chain; their existence in the WTF Protocol on-chain hashmaps is enough proof that they are verified.

3 Existing Identity Protocols

Currently, multiple decentralized protocols for identity exist with unique capabilities and shortcomings. Some with radically different approaches include:

- Ceramic
- Proof of Humanity
- On-chain resume / badge services
- Magic Link
- Traditional KYC services

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

Proof of Humanity provides a handful benefits such as a Universal Basic Income (UBI) token and Sybil resistance. However, its incentive structures do not fully protect from attacks (an attack can cost only $10 [4]) and it requires a significant amount of work from the user verifying, and an additional user.

On-chain resume / badge services provide pseudonymous credentials. They are intended to show accomplishments and functional well for doing so, but they are mostly limited to resume-related, rather than security- and identity-related uses.

Magic Link claims to be 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.

Traditional KYC services provide deep checks into not just identity but also legal status. However, they require significant overhead and are not automated, like the WTF protocol. The WTF will be made interoperable with traditional KYC services.

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 this case, the data is the readable aspect of the JWT, albeit base64url encoded. This 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.

Diagram Here

HMAC signatures, such as those returned by Twitter SSO, need a proxy to convert them to a blockchain-compatible signature. We provide Asymmetric Signing Service (ASS) for this. But relying entirely on ASS for all credentials would centralize the protocol to relying on the ASS server, so we recommend limiting use of ASS except when needed, e.g. for Twitter. We are also migrating ASS to a secure enclave so it can at least be private and trustless, despite requiring a centralized proxy. More people are encouraged to create ASS servers to make it decentralized. Readers are encouraged to reach out to us at wtfprotocol@gmail.com if they are interested in providing an ASS node and in exchange for WTF tokens, source code, 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 (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 JWT with public key, hash it, and submit the result
2. Wait for the current block to be finalized
3. Submit the JWT for verification

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.

First is the question of whether users know blockchain data is public. The purpose of our protocol is to allow users to publicly link their Web2 accounts to their blockchain address – like ENS but with a twitter handle or email address instead of domain name. If users are unclear about this purpose, they may be mistakenly put private accounts on a public blockchain. Thus, we also prompt users clearly with the data that will be made public to ensure they consent.

Second is the question of whether it is safe to publicly reveal OpenID JWTs. We believe it is safe since OpenID JWTs have an 'aud' claim that restricts their audience to the WTF protocol. Thus, revealed JWTs cannot be used to login to other websites.

6 Future Steps

The WTF protocol will enable far more than just verifiable credentials. Our team is 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 the use of 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 (WT-FA) to provide decentralized, on-chain two-factor authentication. DeFi protocols and smart contract wallets could then use WT-FA with just a single line of code to prevent unwanted transfers upon hacks.

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] Author, *title* https://go.chainalysis.com/rs/503-FAP-074/images/Crypto-Crime-Report-2022.pdf

[2] Author, *title* https://dune.xyz/rchen8/opensea

[3] Author, *title* https://twitter.com/opensea/status/1486843201352716289

[4] Ted Suzman *title* `https://hackmd.io/@RoboTeddy/SkFEYwptd`