Web3 wallet as basis for single signon across a collection of dApps

Bring your own Wallet!

Gary Mawdsley CEO Lockular Limited

May 2023[1]

[1] Updated May 2024 to refresh the language

This white paper.

Web3 Authentication

In the context of Web3 applications, especially those interacting with blockchain networks like Polkadot, a Web3 wallet is a crucial tool for authentication. Here's how it generally works:

- Identity Verification: The wallet holds the user's private keys, which are used to sign transactions and prove ownership of a blockchain address. This address acts as a unique identifier (or pseudo-anonymous identity) on the blockchain.
- Authentication: When a user wants to interact with a decentralized application (dApp) on Polkadot, they typically sign a message or a transaction using their private key. This process effectively authenticates the user without the need for traditional login credentials (like a username and password).
- Interaction: Post-authentication, in general the wallet facilitates interactions with the blockchain, such as transferring tokens, voting in governance, or participating in other dApp-specific activities.
- Security and Control: Using a Web3 wallet gives users full control over their data and interactions. Unlike traditional web applications, where the application stores user data, in Web3, the data is often stored on the blockchain and controlled directly by the user through their wallet.

Popular Web3 wallets that can interact with the Polkadot network include Polkadot.js, Talisman, and Subwallet. These wallets support the unique features of Polkadot, such as interacting with multiple parachains and participating in consensus mechanisms like staking and governance.

In summary, Web3 wallets are not just tools for holding digital assets; they are essential for secure and direct interaction with decentralized applications on blockchain networks like Polkadot.

Role based access control (RBAC)

Traditional authentication systems like keycloak can be used to not only define logins with usernames and passwords but also define apps and related roles and access rights. In the specific case of Keycloak this information would be stored in a postgres DB falling under the guardianship of system admins.

In a Web3 environment, traditional role-based access control (RBAC) and permissions management, as seen in systems like Keycloak, need to be adapted to fit the decentralized and non-custodial nature of blockchain technology. Here's how Lockular achieves similar functionalities using a Web3 wallet approach:

- Smart Contracts for Access Control: Instead of a centralised server managing roles and permissions, smart contracts are deployed on the blockchain that define and enforce access rules. These smart contracts can hold the logic for who can access what resources based on their blockchain address or other on-chain credentials.

Lockular's WAM dApp records team structures and are domain dependent. A common smart contract providing RBAC is used to supplement a domain specific HR structure.

- Decentralized Identifiers (DIDs): DIDs are used to create verifiable, self-sovereign identities that are linked to blockchain addresses. These identities can carry verifiable credentials that include user roles and permissions. Lockular DApps then verify these credentials against the rules defined in the smart contracts to grant or deny access.

This is the case in defining division level access to IP Blocks in the Lockular secure EDA solution.

- Token-Based Permissions: Implement token-gating, where access to certain features or areas of a dApp is granted based on the tokens or NFTs held by a wallet. This can be used to represent membership, roles, or specific rights within the application.
- On-Chain Governance for Role Management: With this approach it is possible to utilise on-chain governance mechanisms to allow community voting on changes to roles or permissions, aligning with the decentralised nature of blockchain applications. This opportunity is not yet utilised by the suite of Lockular dApps but would provide the means by which a supply chain could collectively manage access.
- Integration with Off-Chain Systems: On a similar note, for applications that require a blend of traditional and blockchain-based authentication, Web3 wallets can ne integrated with off-chain systems using oracles or other bridging technologies. This allows you to maintain some centralised control while leveraging the security and user control of blockchain technology. An example here is access to parcel carrier systems in a a supply chain work

Bring your own Wallet

In summary Lockular uses some of the above in creating its decentralized access control system that provides similar functionalities to traditional systems like Keycloak but is adapted to the decentralized and user-centric nature of Web3.

Lockular dApps

Web3 wallets sit at the heart of Lockular's supply chain provenance suite:

- Lockular Provenance Tracking Fileystem
- Workflow Actor Management
- Secure Artifact Store
- SecureWorking
- NFT Marketplace Platform

Multi-Sig

In addition to utilising the signature from the signed-in user wallet, some of the subsystem components also contribute platform signatures. This is the means by which provenance records are enshrined in a blockchain by multiple cooperating parties.

Considerations in the design from the team

Fundamentally it is pointed out that keycloak provides both authentication and authorisation. In any remodelling we have to recognise this and reimplement where wallet is simply the focus of authentication and WAM provides the authorisation. Currently keycloak does a authentication and some authorisation.

- Login process for dApps often sets up cookies - we need to define what they are and how they are created.
- Over time a person may utilise more than one wallet. Therefore RBAC info should be disconnected from a specific wallet in WAM (perhaps a SID - wallet relationship).
- SecireWorking is not one single dApp but a land page app and an IDE app - session info has to be considered carefully after login.
- the IDE capability run on a kubernetes pod and specifically instances of entities in the CRD relate to session information.

References