Capstone Project: Letter of Intent

Zubayr Khalid

Capstone Project Name: One Configurable Standard (1CS)

Brief Project Description: In the blockchain ecosystem, shared ownership, data access, and asset management present unique challenges. Existing solutions like multisig wallets and static token standards often lack flexibility and user-friendliness when it comes to accessing assets by multiple wallets or accounts.
1CS (One Configurable Standard) aims to bridge this gap by introducing a novel framework for managing shared access to on-chain assets and data. Unlike traditional approaches, 1CS focuses on flexibility, enabling wallets to hold specific roles and permissions. This makes it an ideal choice for applications that demand nuanced access control, multi-party ownership, or time-based asset management.

1CS is to be designed as a general-purpose application layer on Solana, providing a flexible framework for managing permissions and roles associated with on-chain data. Key components of the 1CS methodology include:

1. **Role-Based Access Control (RBAC):**
    - Assign specific roles to wallets, such as “owner”, “admin,” “full-access,” or “time-restricted access.”
    - Store permissions dynamically in a smart contract, allowing real-time updates and revocations.
2. **Asset Enveloping:**
    - Use smart contracts to "envelope" data or assets, linking them to an access control list (ACL).
    - The ACL defines the rights of each wallet in relation to the enveloped data.
3. **Interoperability:**
    - Build on existing SPL token and NFT standards to ensure compatibility with Solana's ecosystem.
    - Enable seamless integration with wallets, dApps, and marketplaces.
4. **Customizable Framework:**
    - Design the protocol to be adaptable for various use cases, allowing developers to define roles and permissions specific to their needs.

Reason for Choosing this Project: As of now in Solana (or any other blockchain ecosystem) there is not much work on access management of data or assets. There are some separate protocols such as “delegating” (eg- https://solana.com/developers/courses/token-extensions/permanent-delegate) tokens to other wallets to transfer/ burn later, or “transferring ownership” of fungible tokens or NFTs, but there is not a single protocol or application that can deal with all these issues altogether.

There are some other web3 world problems that access management of data or assets can solve, such as - creating a “backup wallet” or creating a “will” where tokens can be accessed by “allocated” wallets after a specific time. Besides, with 1CS, normal wallets can also be used as multi-sig wallets. Another application would be creating multi-owner NFTs.

Target Audience:

1. **Shared Ownership of Digital Assets:** Users who enable multiple wallets to co-own and manage an asset without requiring multisig. Example: A shared property deed or intellectual property rights.
2. **Multi-Owner NFTs:** Users facilitating collective ownership or management of NFTs with unique roles for each wallet. Example: An artist retains primary rights while collaborators exhibit or promote the NFT.
3. **Certificates with Issuer/Receiver Roles:** Educational or industry organisations and their students or employees who require certificates or credentials to be shared between issuers and receivers, with issuers maintaining update or revocation rights. Example: Universities issuing degrees or companies providing proof of employment.
4. **Time-Based Data or Asset Access:** Normal users or committees who grant time-limited access to assets or data, automating expiration through smart contracts. Example: Event tickets, memberships, or leased digital assets.
5. **Borrowed Wallets:** New users who can trade in crypto without requiring to download or maintain wallets while their assets being stored in the used DEX protocol wallet (in exchange of some fees). Can also be used for users storing assets in web3 bank. Example: A third party wallet acting as a savings account, offering users access to funds with controlled withdrawal conditions.
6. **Web3 based Will:** Users who want to will their asset to a separate wallet. The allocated wallet will be granted access to the assets after a specific time. Example: Can be used in a Web3 based Will application.
7. **Backup Wallet:** Anyone willing to use a backup wallet for their primary wallet. Example: User creating a backup wallet/ account to their primary account where both accounts can access the assets.

Value Proposition: 1CS will allow users to maintain an Access Control List (ACL) for their assets where they can provide certain permissions to other wallets. The permitted wallets can perform certain operations on the assets such as modifying it, and if enough permission is given, add more wallets on behalf of the owner. Blinks (Dialect) can be used to notify the permitted wallets. With the ACL defined in 1CS protocol, users will be able to perform a series of different tasks (defined in the previous section), which, otherwise, they would need to use multiple protocols, applications or wallets (eg. - using multi-sig wallets to provide rights of an asset to 2 different user accounts while accounts having their wallets registered in multiple divides to secure their assets).

Marketing and Distribution: Instead of having social media marketing from the start, which does not always provide good output, unless the organizers have a good marketing budget, or already have a good community, 1CS will follow these steps in order:

1. **Launch Will app:** Create a community through the Will app. The application can spread the knowledge on 1CS to the Solana community and it can also arrange enough funds for future social media marketing.
2. **Hackathon:** After the application is done to a level where it can allow multiple users to access tokens through frontend application, participate in Radar Hackathon to display the significance of 1CS to the Solana developer community.
3. **One Source Code**: Provide openAPIs to integrate 1CS library in other Solana based DAPPs.
4. **Social Media Marketing**: Create a social media campaign.
5. **Collaborate with Existing Wallet**: Collaborate with an existing wallet so that the wallet allows users to encapsulate assets and use an ACL for maintaining permissions when connected with 1CS application.

Competitive Landscape:

Feature	1CS	Magic Eden	Solanart	Staking Applications	Permanent Delegate Extensions	Solana SPL, Metaplex Library	Solana Wallets
Share Ownership of digital assets	✔️	❌	❌	❌	✔️	✔️**	❌
Multi-owner NFTs	✔️	❌	❌	❌	❌	❌	❌
Delegating Certificates with Issuer Access	✔️	❌	❌	❌	✔️	✔️	❌
Time-Based Data or Asset Access	✔️	❌	❌	❌	✔️*	✔️*	❌
Ability to borrow Assets	✔️	❌	❌	❌	❌	❌	❌
Web3-based Will	✔️	❌	❌	❌	❌	❌	❌
Backup Wallets	✔️	❌	❌	❌	❌	❌	✔️****
Royalty Management	✔️	✔️	✔️	❌	❌	✔️***	❌
Mobile Application	✔️	✔️	✔️	❌	❌	❌	✔️
Browser Extension	✔️	❌	❌	❌	❌	❌	✔️

(
** = it is possible to **extend** or **customize** the delegation functionality to incorporate time-based access.*
*** = Can use functions such as transfer(), approve(), setAuthority().*
**** = Metaplex library can provide royalty management.*
***** = Creating the same wallet account in different devices so that if one device is inaccessible, the wallet still can be used from other devices. But note, it is not a separate wallet.*
)

Tech Stack:

1. Smart Contract
    - Access Control Logic: Define roles and permissions for each account.
    - Data Enveloping: Associate on-chain data or assets with access control rules.
    - Role Management: Add, modify, or revoke roles dynamically.
    - Event Handling: Emit events when permissions are updated or accessed.
    - Libraries: SPL library (for simple token), Metaplex library (for NFTs)
2. SDK (for Scaling; Not in MVP)
    - Encoding and decoding instructions for the smart contract.
    - Abstracting complex smart contract interactions (e.g., role management).
    - Providing utility functions for developers (e.g., querying permissions).
    - Programming Language: Js/ Ts for making it compatible with the frontend.
3. Frontend
    - Enable users to interact with the smart contract through a user-friendly interface.
    - Allow actions like adding permissions, viewing access control lists, and interacting with enveloped data or assets.
    - Display a dashboard for role management, activity logs, or data sharing.
    - Tech Stack: Next Js
4. Backend (Optional)
    - Cache on-chain data for faster querying and displaying it in the frontend.
    - Handle integration with external services, such as email notifications for permission changes or integration with Dialect.
    - Tech Stack: Node Js or FastAPI

Smart Contract Development:

Anchor Solana will be used for 1CS but Solana's stateless structure, where accounts store the data is a crucial part of 1CS. Therefore, Rust will be used. Typescript Specs test will be used for testing.

Conclusion:

1. **Project TimeLine:**
    a. **MVP: Develop the Smart Contract in Anchor (Feb, 2025):**

i. Writing program logic to manage roles, permissions, and data enveloping.
ii. Providing features for enveloping SPL tokens.
iii. Ensuring the contract is scalable to handle additional use cases in the future.

b. **V1: Basic Frontend/CLI & Creating the Will App (Jul, 2025):**
   i. Providing a simple interface or CLI tool to test adding and removing permissions.
   ii. Allowing querying access rules for a given data or asset.
c. **V2: Building the Ecosystem (Dec, 2025)**:
   i. Providing the facility for NFTs
   ii. Creating the feature for borrowed wallets
   iii. Updating the frontend for the new features.
   iv. Providing Blink features.
d. **Future Versions (2026):**
   i. Creating the SDK
   ii. Creating the OpenAPI and backend
   iii. Other future developments

2. **Commitment**: Until the V1 is launched, I am fully prepared to dedicate at least 25 hours per week for the project. After V1, I will create a team of 3 from my friends (also tech developers in different Software based fields) so that we can collectively work on the Radar hackathon. If 1CS gets enough funds in the hackathon, I will start working full time.
3. **Initials:** Zubayr Khalid