0.1 Introduction

This chapter establishes the theoretical framework and technological infrastructure utilized to construct the Multi-chain Stablecoin Protocol. The methodology is grounded in a rigorous analysis of distributed ledger technologies, cryptographic primitives, and financial stability mechanisms. By synthesizing the high-performance capabilities of the Solana blockchain with the deep liquidity of EVM-compatible networks, the proposed solution adopts a Hub-and-Spoke architecture. This chapter provides a detailed exposition of the selected blockchain platforms, the software engineering principles behind the smart contracts, the mathematical foundations of the cross-chain verification process, and the economic theory underpinning the CDP mechanism.

0.2 Blockchain Platforms and Architecture

0.2.1 Blockchain Fundamentals and Distributed Ledger Technology

The technological cornerstone of this thesis is the blockchain, a concept fundamentally introduced as a purely peer-to-peer version of electronic cash. As delineated by Satoshi Nakamoto in the seminal Bitcoin whitepaper, a blockchain is a distributed system that solves the double-spending problem without relying on a trusted central authority or a financial institution [1]. The core innovation lies in the use of a timestamp server, which works by taking a hash of a block of items to be timestamped and widely publishing the hash. This process proves that the data must have existed at the time, creating a chronological chain of hashed proof-of-work.

This structure forms an immutable record of events. Each timestamp includes the previous timestamp in its hash, forming a chain where each additional timestamp reinforces the ones before it. In the context of a financial protocol, this mechanism ensures integrity and transparency. The network operates on a consensus mechanism where the longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. This guarantees that as long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they will generate the longest chain and outpace attackers.

While Bitcoin established these principles for decentralized currency, the underlying technology, the distributed ledger provides the essential substrate for the Multi-chain Stablecoin Protocol. The guarantee that a transaction (such as depositing collateral) is irreversible once sufficiently buried under proof-of-work (or proof-of-stake in modern iterations) is critical for maintaining the solvency of a

CDP system. This thesis leverages these foundational properties of trustlessness and cryptographic proof to construct a secure financial architecture across disparate networks.

0.2.2 The Hub: Solana Blockchain Architecture

To address the latency and throughput limitations inherent in earlier distributed ledger technologies, this project selects Solana as the central "Hub" for orchestrating the global financial state. The decision is grounded in the architectural innovations proposed by Yakovenko, which introduce a novel method for high-performance blockchain execution that does not rely on wall clock timestamps or local time for synchronization [2].

The defining feature of this architecture is Proof of History (PoH). As defined in the whitepaper, PoH is a sequence of computation that provides a way to cryptographically verify the passage of time between two events. It essentially functions as a high-frequency, verifiable delay function. By encoding the passage of time directly into the ledger, Solana allows the network to create a historical record that proves that a specific event occurred at a specific moment in time, before or after other events. This cryptographic clock eliminates the need for massive messaging overhead typically required for nodes to agree on a timestamp, which is a primary bottleneck in traditional consensus mechanisms.

This architectural shift allows the system to implement a deterministic leader schedule and streamline the consensus process. Because the order of events is verifiable via the PoH data structure, the network can process transactions as they arrive rather than waiting to batch them into blocks based on uncertain network latencies. For the Multi-chain Stablecoin Protocol, this architecture is critical. The ability to verify time and order with sub-second latency ensures that the "Universal Wallet" state remains synchronized and that liquidation triggers based on Health Factor calculations are executed with the speed and determinism necessary to maintain protocol solvency.

0.2.3 The Spokes: Ethereum Virtual Machine (EVM) Networks

While the Solana Hub manages the execution state, the Ethereum Virtual Machine (EVM) ecosystem serves as the critical Asset Custody Layer, or the "Spokes," of the architecture. The selection of EVM, compatible networks is rooted in the foundational design of Ethereum as a "next-generation smart contract and decentralized application platform." As articulated in the Ethereum whitepaper, the core innovation of this system is the introduction of a Turing-complete programming language, allowing anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats, and state transition functions [3].

The EVM operates as a quasi-Turing complete machine that maintains a singleton state computer shared by all participants in the network. Every transaction in this model is a cryptographically signed instruction that transitions the global state from one valid state to another. This state is maintained in a specialized data structure known as the Merkle Patricia Tree, which ensures a high degree of efficiency and cryptographic security for light client verification. The protocol leverages this robust state architecture to deploy "Vault" contracts. These contracts utilize the standardized ERC-20 interface, which a direct derivative of Ethereum's programmable token capability to securely lock collateral assets.

By integrating the EVM as the spoke layer, the protocol taps into the concept of "programmable money" envisioned in the whitepaper. While the EVM's design prioritizes security and generality over high-frequency throughput, its deterministic execution environment provides the ideal substrate for holding significant value. The architecture thus relies on the EVM not for rapid calculation, but for its established consensus and the immense liquidity of assets like Ether, which are native to this "World Computer" paradigm.

0.3 Smart Contract Development Frameworks

0.3.1 Solana Program Development: Rust and Anchor

The development of the protocol's core logic on the Solana Hub is architected utilizing the Anchor Framework, a sophisticated development tool designed to abstract the complexities of the Sealevel runtime. While Rust provides the underlying memory safety through its ownership and borrowing models, raw Solana program development requires developers to manually implement verbose serialization logic and low-level byte manipulation. Anchor addresses these challenges by introducing a framework that enforces security constraints and streamlines the development lifecycle through an opinionated design pattern [4].

A primary motivation for adopting Anchor is its rigorous approach to account validation and security, specifically through the implementation of the "Discriminator." In the native Solana programming model, any account passes as a byte array, making the system vulnerable to type-confusion attacks where a malicious actor might pass a data account of one type into an instruction expecting another. Anchor mitigates this by automatically prepending a unique 8-byte discriminator derived from the SHA-256 hash of the account structure's name to the account data. During execution, the framework verifies this discriminator before deserializing the data.

Furthermore, Anchor standardizes the interaction between the on-chain program and off-chain clients through the IDL. Similar to an ABI in the EVM ecosystem, the IDL provides a structured JSON representation of the program's instructions and account structures. This allows the Guardian infrastructure to deterministically serialize inputs and deserialize outputs without manual offset calculations. By handling the "boilerplate" code of account serialization and constraint checks (such as ensuring an account is mutable or is a signer) via macros, Anchor allows the development focus to remain on the high-level financial logic of the CDP mechanism, ensuring that the state transitions on the Solana Hub are both secure and predictable.

0.3.2 EVM Smart Contracts: Solidity and Hardhat

For the Asset Custody Layer residing on the EVM Spokes, the development methodology prioritizes security and standard compliance over raw execution speed. Accordingly, the Solidity programming language is utilized to construct the Controller and Vault contracts. As a statically-typed, contract-oriented language designed specifically for the Ethereum Virtual Machine, Solidity allows for the precise definition of state variables and the implementation of complex access control logic required to secure collateral assets. The choice of Solidity ensures maximum compatibility with the ERC-20 token standard, which is the fundamental data structure for the assets (such as USDC or WETH) that the protocol accepts as collateral.

To ensure a robust development lifecycle, the project leverages the Hardhat development environment. Hardhat is instrumental not merely as a compiler, but for its advanced testing capabilities, particularly "Mainnet Forking." This feature allows the development team to simulate the protocol's interaction with the real Ethereum state and existing token contracts in a deterministic local environment, validating the Vault's logic against actual network conditions before deployment.

Crucially, to mitigate the inherent risks of smart contract vulnerabilities, the system integrates the OpenZeppelin library. Given that the Controller contract functions as a decentralized vault holding user funds, it is a high-value target for exploits. Rather than relying on custom security implementations, the contracts inherit from battle-tested modules. Specifically, the "ReentrancyGuard" is applied to all asset withdrawal functions to mathematically prevent recursive call attacks, while the "Ownable" pattern is utilized to enforce strict access control, ensuring that only the authenticated Guardian infrastructure can trigger state-changing callbacks [5].

0.4 Cryptography and Cross-chain Verification

0.4.1 Elliptic Curve Mismatch and Solution

A fundamental technical challenge in implementing a "Universal Wallet" lies in the cryptographic incompatibility between the disparate blockchain networks. The Ethereum ecosystem relies on the Elliptic Curve Digital Signature Algorithm (ECDSA) utilizing the secp256k1 curve. In contrast, the Solana blockchain is built upon the Ed25519 curve, which offers faster signature verification times but different mathematical properties. This discrepancy implies that a standard Solana smart contract cannot natively verify a signature generated by an Ethereum private key without significant computational overhead.

0.4.2 Mathematical Formulation of ECDSA Verification

To bridge this gap, the protocol requires a mechanism to mathematically prove that a transaction request m originating from an EVM address A_{evm} was indeed authorized by the holder of the private key d. The domain parameters of the secp256k1 curve are defined over a finite field $\mathbb{F}_p$ by the Weierstrass equation:

$$y^2 \equiv x^3 + 7 \pmod{p} \tag{1}$$

When a user signs a cross-chain request, they produce a signature pair (r, s). The verification process, which must be executed on the Solana Hub, involves recovering the public key point Q from the signature. This is achieved using the inverse operation of point multiplication:

$$Q = r^{-1}(sR - zG) \tag{2}$$

Where z is the hash of the message m, G is the generator point of the curve, and R is the point with x-coordinate r. The derived Ethereum address is then the last 20 bytes of the Keccak-256 hash of Q:

$$A_{evm} = \text{Keccak256}(Q)[12..32] \tag{3}$$

0.4.3 Implementation via Solana Native Program

Implementing the arithmetic operations of Equation 2 directly in a high-level language like Rust would consume an excessive amount of Compute Units, potentially exceeding the block limit. To solve this, the methodology employs Solana's "Native Secp256k1 Program". This is a precompiled BPF program embedded in the validator software that executes signature recovery at a fraction of the computational cost. By invoking this program via CPI, the Main Contract can trustlessly verify EVM signatures, thereby allowing users to control their Solana state using their existing MetaMask wallets without compromising security.

0.5 Theoretical Foundation: The CDP Mechanism

0.5.1 Economic Model of Stability via Over-collateralization

The financial architecture of the proposed protocol is grounded in the CDP model, a primitive that enables the creation of decentralized stablecoins without reliance on fiat reserves or centralized custodians. Unlike algorithmic stablecoins which often depend on endogenous collateral and reflexive arbitrage cycles, mechanisms historically prone to "death spirals" during market downturns. In this protocol, the CDP model enforces stability through strict Over-collateralization. This economic theory posits that the solvency of the system and the peg of the stablecoin are guaranteed as long as the aggregate value of the locked collateral assets significantly exceeds the value of the issued debt. In this system, the stablecoin is not created out of thin air but is backed by a diversified basket of crypto-assets (such as ETH, WBTC) locked across various EVM chains, serving as a secure liability against the protocol's assets.

0.5.2 Borrowing Capacity and LTV Formulation

A critical component of the CDP mechanism is the determination of a user's borrowing power. This is governed by the LTV ratio, a risk parameter assigned to each asset type based on its market volatility and liquidity profile. The LTV ratio, denoted as α, represents the maximum percentage of the collateral's value that can be minted as debt. For instance, a stable asset like USDC might have a high α (e.g., 0.90), while a volatile asset like ETH might have a lower α (e.g., 0.75).

Mathematically, the Maximum Borrowable Debt (D_{max}) for a user holding a portfolio of assets across multiple chains is the sum of the risk-adjusted value of each asset. For a user depositing N different asset types distributed across M blockchain networks, the borrowing capacity is formalized by the following equation:

$$D_{max} = \sum_{k=1}^{M} \sum_{i=1}^{N} \big(Q_{k,i} \cdot P_i \cdot \alpha_i\big) \tag{4}$$

In this equation, $Q_{k,i}$ represents the quantity of asset i locked on chain k, and P_i is the real-time market price of asset i denominated in the reference currency

(USD). The term α_i acts as a dampening factor, ensuring that the protocol accounts for potential price drops before they occur. Consequently, the actual debt D_{actual} minted by the user must always satisfy the condition $D_{actual} \leq D_{max}$ at the time of issuance.

0.5.3 The Health Factor Function

The system solvency is mathematically modeled using the Health Factor (H_f). This metric is dynamic and is updated in real-time based on oracle price feeds. For a multi-chain protocol, the Health Factor is an aggregate function of all collateral assets i across all connected chains k:

$$H_f = \frac{\sum_{k=1}^{M} \sum_{i=1}^{N} (Q_{k,i} \times P_i \times \lambda_i)}{D_{total} \times P_{peg}} \tag{5}$$

In this equation:

- $Q_{k,i}$ represents the quantity of asset i locked on chain k.
- P_i is the current market price of asset i denominated in USD.
- λ_i is the Liquidation Threshold specific to the asset's risk profile (e.g., $\lambda_{ETH} = 0.80$, $\lambda_{USDC} = 0.95$).
- D_{total} is the total outstanding stablecoin debt.
- P_{peg} is the target peg price of the stablecoin (e.g., 1 USD).

The methodology enforces a strict constraint where $H_f \geq 1$. If market volatility causes $H_f < 1$, the protocol triggers a liquidation event. This deterministic mathematical model ensures that the protocol remains solvent and the stablecoin maintains its peg, fulfilling the core objective of the thesis.

0.6 Off-chain Infrastructure

The final component of the methodology is the Guardian Network, which serves as the interoperability layer connecting the deterministic worlds of the blockchain networks.

0.7 Off-chain Infrastructure and Guardian Network

0.7.1 Architectural Overview

The Off-chain Infrastructure, referred to as the Guardian Network, functions as the cryptographic bridge and synchronization engine between the deterministic environments of the EVM Spokes and the Solana Hub. This component is engineered using Python. Unlike passive indexers, the Guardian plays an active role in state transition, specifically in the secure initialization of user identities and the cross-chain relaying of assets.

0.7.2 Universal Wallet Initialization via Signature Verification

A critical security function of the Guardian is the verification of user intent during the creation of the Universal Wallet. Since the Universal Wallet acts as the central storage for a user's multi-chain positions, it is imperative to establish a cryptographically proven link between the user's request and their destination wallet on Solana before any on-chain state is modified.

The process is designed as a "Verify-then-Execute" workflow to prevent Denial of Service attacks and ensure that only authenticated users can allocate storage on the Solana blockchain. The workflow proceeds as follows:

First, the user constructs a request message m, which includes the intent to create a wallet, a unique nonce to prevent replay attacks, and the public key of the destination Solana wallet (PK_{dest}). To prove ownership of this destination wallet, the user signs the message m using their corresponding private key, generating a digital signature σ. This signature typically utilizes the Ed25519 algorithm, which is the native standard for Solana addresses.

Second, the Guardian receives this payload (m, σ, PK_{dest}) via a secure API endpoint. Instead of immediately submitting a transaction to the blockchain, the Guardian performs an off-chain verification using cryptographic libraries. The verification function can be formalized as:

$$V(m, \sigma, PK_{dest}) \to \{\text{True, False}\} \tag{6}$$

If the function returns False, the Guardian rejects the request immediately, ensuring that no gas fees are wasted on invalid transactions.

If the function returns True, the Guardian confirms that the requestor possesses the private key controlling the destination wallet. Subsequently, the Guardian constructs a transaction instruction invoking the Initialize Wallet method on the Solana Main Contract. The Guardian then signs this transaction with its own keyer key (to pay for transaction fees) and submits it to the Solana cluster. Upon successful execution, the Solana state is updated: a new PDA is initialized, serving as the Universal Wallet for that specific user. This mechanism ensures that the heavy lifting of cryptographic validation is shared between the off-chain infrastructure and the on-chain logic, optimizing both security and cost-efficiency.

0.8 Conclusion

This chapter has detailed the methodological approach for the Multi-chain CDP Protocol. By combining the safety of Rust/Anchor for state management, the standardization of Solidity for asset custody, and the mathematical rigor of the Secp256k1 verification for cross-chain identity, the proposed architecture provides a robust solution to the problem of liquidity fragmentation. The economic stability is underpinned by the Over-collateralized CDP model, while the Guardian infrastructure ensures seamless connectivity. These foundational technologies directly enable the implementation of the system design proposed in the subsequent chapters.