EEA CIW - Crosschain Security Guidelines Version 1.0

Crosschain Interoperability WG - Working Draft

24 September 2021

Editors:

- Weijia Zhang, Wanchain
- Peter Robinson, ConsenSys
- Aiman Baharna, Clearmatics

Related work:

[eeaciw-crosschainidentification-v1.0] *EEA CIW - Crosschain Identification Specification Version 1.0*. Edited by Weijia Zhang and Peter Robinson. 14 December 2020. EEA CIW. https://entethalliance.github.io/crosschain-interoperability/crosschainid.html . Latest stage: https://entethalliance.github.io/crosschain-interoperability/crosschainid.html .

Status:

This section describes the status of this document at the time of its publication.

This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.
Please note that Working Drafts published by EEA working groups are not endorsed by the EEA.

At the time of publication, no additional substantive issues or modifications are expected but readers are advised that proposed changes might result in alterations to this document.
The Crosschain Interoperability Working Group intends to request publication of the final version of this document in Q4 2021.

Citation format:

When referencing this document, the following citation format should be used:

[eeaciw-crosschainsecurity-v1.0] *EEA CIW - Crosschain Security Guidelines Version 1.0.* Edited by Weijia Zhang, Peter Robinson and Aiman Baharna. 24 September 2021. EEA CIW. https://entethalliance.github.io/crosschain-interoperability/crosschainsecurityguidelines.html . Latest stage: https://entethalliance.github.io/crosschain-interoperability/crosschainsecurityguidelines.html .

Notices

Table of Contents

Introduction

Security is always the most critical part of software systems and platforms. This is even more the case in blockchain for the following reasons:

- Decentralized nature of blockchain: Any code written and deployed to a blockchain will be run on many blockchain nodes. Anyone can access and run your blockchain code.
- Constraints of patches and upgradability: Due to the immutability of blockchains, smart contracts deployed to blockchains cannot be modified. This increases the difficulty for upgrading decentralized applications. When security flaws are detected in blockchain applications, the cost of patching the applications is high. Poorly designed contracts may be impossible to upgrade.
- Trustless and permissionless environment: For public blockchains, both the blockchain client nodes and decentralized applications are open to global participants.

There is no centralized authority to check the qualifications of participants. There is no security perimeter to block bad actors from participating.

- Privacy and anonymous nature of blockchain: Blockchain users can remain pseudo anonymous. Smart contract functions do not have a way to check the profile of the users. Hackers can carry out an attack, stealing the assets, and remain unidentified.
- High value impact on business: Smart contracts normally have a small footprint. Bigger projects might have in the order of thousands of lines of code while other project may only have hundreds lines of codes. These small pieces of code manage high value crypto assets. A single attack can bring catastrophic results to a decentralized application. Some decentralized applications have suffered huge losses due to simple errors in smart contracts.

Recently, there has been a huge increase in attacks on crosschain bridges. Hackers exploit vulnerabilities in smart contracts, Remote Procedure Calls (RPC), wallets, private key managers, and source code repositories. Since a crosschain bridge is a holistic system that connects a source chain with a target chain, the attacks can target the source chain, the target chain, or the bridge connecting them. While there are many security guides and best practices for securing blockchains and decentralized applications in a single blockchain system, crosschain security is not as well-served. This guideline fills that gap by describing the areas and factors to consider for crosschain bridges.

Crosschain Security Considerations

To better explain crosschain security factors, we use the Ethereum architecture diagram that was developed by Enterprise Ethereum Alliance as shown in Figure 1.

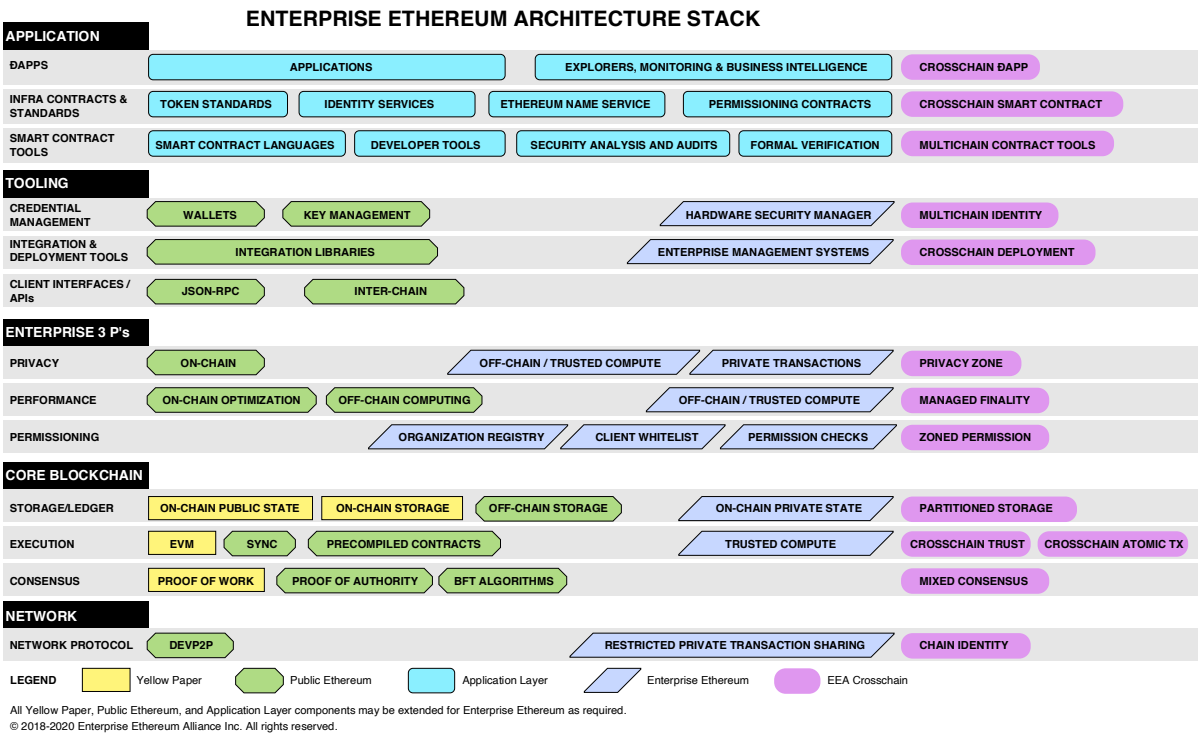

Figure 1: Ethereum architecture with crosschain interoperability components

Blockchain Layer

The crosschain operation should correctly discover and identify source and target blockchains. Today's blockchain identification of using ChainID is not adequate as any blockchain can assign a blockchain id to mimic another blockchain. There should be a mechanism to authenticate a blockchain through blockhash or blockchain data that cannot be altered. One way to do this is to use genesis blockchain hash as an identifier of a blockchain. Once a user specifies the source blockchain and target blockchain in the transaction, the relayers and dapps can verify blockchain id with the genesis blockchain hash. In the case of forked blockchain, the main chain and forked chain should be reviewed, and the forked chain can take the blockchain hash of the block when the forked chain happened. To ensure blockchain layer security for the crosschain operation, there should be a unique blockchain id defined. The blockchain id should be verifiable within the blockchain, the blockchain id can also be registered with a trusted source to include more metadata associated with the blockchain. Besides using a native blockhash to identify a blockchain, the blockchain id can also be expanded to include a checksum and other critical metadata to describe a blockchain. The other additional metadata can then be registered with a name service such as an Ethereum Registration Authority. Organizations such as Enterprise Ethereum Alliance can take the role of helping to ensure that blockchain identification, description, discovery, and registration are consistent with the specification for crosschain operations.

Today, there are blockchains like Ethereum that have an open process for protocol development, which involves multiple stakeholders in consultation and reaching consensus about upcoming protocol changes. In contrast, other blockchain protocols may be controlled by a single entity, which gives this party a great deal of control over user funds, their tokens, and the smart contracts they can deploy and execute. This creates a single point of failure and increases the risk of adverse outcomes in crosschain operations. Also, attention needs to be directed towards the node software that powers the blockchain network. If there are multiple implementations of the blockchain node client, which are developed and released by different groups of programmers, using different programming languages and system libraries, and their usage is evenly distributed among miners and validators all across the world, then a failure in any one node client is less likely to affect the other clients. As a result, this makes the entire network more resilient to attacks and serious failures in any one client. Crosschain bridges can make use of multiple clients to ensure that operations are not disrupted in the event of a problem with the node software.

Consensus layer

When a crosschain bridge is built for a source chain and a target chain, the security of the consensus in its respective native chain needs to be considered. These security factors include who runs as miners for the native chains, what consensus algorithm is used for the blockchains, what finality does the consensus algorithm supply given the configuration, and what are the risks of blockchain rollbacks and forks.

For some blockchains such as private or consortium blockchains, the validator nodes that mine blocks are validated and trusted, and therefore there is less chance for malicious actions from the validators. For public blockchain, the miner nodes are permissionless and security is guaranteed via proof of work (PoW) or proof of stake (PoS). It is important to note that for PoW consensus, malicious nodes that have large hashing power can mine blocks faster than honest nodes, and hence can overtake a blockchain. This is extremely unlikely to occur for

Ethereum MainNet but can occur for PoW blockchains that have a lower hash rate. This introduces a risk for crosschain operation, as an asset locked in a source chain might be on a wrong fork and needs to be rolled back. And when a transaction is rolled back, its corresponding asset that was transferred to the target blockchain might be spent already, and cannot be rolled back. It is reasonable to assume that once a security problem is found in a source chain or target chain, the crosschain transactions will most likely be unrecoverable and the procedure of compensating the crosschain asset loss will need to be triggered.

Another risk factor is finality of the source blockchain. A transaction is final when the block containing the transaction can not be changed. For many crosschain protocols, when a transaction is executed in a source blockchain, an event is recorded and used to trigger a corresponding transaction on a target chain. For a relayer that transfers the event on the source chain, it is essential that the source chain transaction is finalized. For most public blockchains, the finality of a blockchain is probabilistic. That is, it is not a fixed number of blocks. It is important to determine a reasonable number of block confirmations to ensure a balance of security and performance. Ideally, a consensus algorithm will support one block finality (also known as Instant Finality). This is the best feature to have in order for supporting crosschain operation. However, the existing blockchains such as Ethereum MainNet and the Bitcoin blockchain require multiple confirmed blocks for a transaction to be finalized. This is a factor to consider when analyzing crosschain security.

Crosschain Relayers

Crosschain relayers carry assets, messages, events, and commands from a source blockchain to a target blockchain. This is an offchain operation and therefore it is very difficult to ensure security. The relayers can be permissioned or permissionless. For permissioned relayers, there is a governance smart contract to administer the registration of the relayers. The relayer registration smart contracts are managed by multiple administrators with a voting mechanism provided to approve or reject relayers. For permissionless relayers (also known as Open Relayers) any crosschain node can be registered as a relayer by the open registration smart contract as long as certain criteria such as minimum system requirements and asset staking criteria are met. Permissionless relayers are required to stake assets to a governing smart contract as a security deposit to prevent wrong-doing. The smart contracts are implemented with a slashing mechanism to punish relayers who manipulate crosschain transactions and proofs.

The security of permissioned relayers are governed by the integrity and truthfulness of the relayer administrators. When a relayer registration smart contract is first deployed, there is a single owner of the smart contract. The owner then adds additional administrators based on crosschain operation governance policy, evolving from single ownership to a group of owners. The group administrators can then adopt a voting mechanism coded in the smart contract to add relayers and additional administrators. This relayer model best fits a consortium where crosschain operations are managed collectively, an administrator group serves as the board of directors, and relayers are verified and approved to relay crosschain transactions, events, and commands.

The security of permissionless relayers are guarded by asset staking, randomness, and multi-party computing. Staking allows a permissionless relayer node to be held accountable for any fund lost due to malicious manipulation of crosschain transactions. The relayer nodes can also be randomly selected from a pool of relayers candidates to form a multi-party computing

(MPC) group that does not rely on a single visible private key. Instead, it can require a threshold of relayers to sign the message together in order to authorize a crosschain transaction. The higher the threshold of a MPC group, the less of a chance there is chance for a relayer group to collude. One thing to note is that although higher signing thresholds harden security, there is a side effect that there are more restrictions for MPC members, and hence this decreases the availability of the system.

Smart Contract Layer

Similar to decentralization, in order to safeguard the crosschain smart contract, the owner of the smart contract needs to safeguard its private key through a Hardware Security Module (HSM), Key Management System (KMS), hardware wallet, offline wallet, or secure vault technology. Alternatively, shared ownership can be established by having the contract owned by a multi-signature wallet. The smart contract owner can also denounce the ownership of the smart contract, so that the smart contract cannot be altered. However, this model has the drawback that without additional mechanisms and safeguards in place, the system cannot handle emergency situations such as coding errors or external security breaches, and needs to be redeployed to recover from such problems.

Oracle Layer Security

The Oracle layer is external to source chain, target chain and relayers, and is not administered by the crosschain operation group. It is important to choose a trusted oracle service. At this time, the crosschain interoperability specification does not cover the Oracle service and we recommend that best practices and security procedures with respect to using third party services are followed.

Web Services

Although blockchain has higher degree of decentralization and security when compared with legacy IT systems, most of the dapps have a web service layer that aggregate user actions and transform them into blockchain transaction raw data. This web service is centralized, and all security considerations for the web should be followed. It would be good to separate private key storage and transaction signing from any web services.

Administrator Account

When deploying a smart contract to a blockchain, there needs to be an administrator who sends the deployment transaction to the blockchain. To sign a transaction, the administrator needs to unlock the account with the private key. If an account is unlocked on a blockchain node, its private key is open, and can be stolen by an attacker. This kind of hack has taken place multiple times in the crypto world. There are more secure ways to deploy smart contracts. For example, administrators can use a hardware wallet, or use an offline wallet for the deployment. In both cases, the private keys are kept in separate devices and only signed transactions are copied to the online system to be sent out to the blockchain. The private key is then secure against network attacks as it never leaves the dedicated device. However, physical attacks and tampering with devices should still be considered.

Securing Management Accounts with MPC

There are two kinds of accounts that can be used to manage smart contracts, asset tokens, and crosschain bridges: Externally Owned Account and Contract Account. The owner account of a smart contract is the most important account, as this is the party that deploys smart contracts and has the privilege to update, halt, transfer, or disable a smart contract. The owner account can also give up ownership and hence make the smart contract stand on its own. Using dedicated administration accounts can provide an additional account to manage the operations of the smart contract. The administration account can set the parameters for a smart contract and also change the state of a smart contract. To increase the security of management accounts, it would be good to adopt MPC (multi-party computing) to safeguard the private key of such a joint account. The way an MPC works is through sharding a private key into multiple segments and each person has a portion of the private key. When signing a transaction, a certain portion of MPC nodes will need to sign the transactions individually and send out the signature to the MPC group. The MPC method provides a way to sign transactions collectively and hence dramatically improve crosschain security.

Staking and Slashing

For crosschain bridges that are truly decentralized and permissionless, the security of the crypto assets will need to be safeguarded by assets staked by the bridge nodes, in order to ensure that there is no collusion among them and any wrongdoing by the bridge operators will be slashed using the stake deposited for the bridges. Similar to the PoS (proof of stake) blockchain consensus model, stake can help the crosschain registration service to rank and select operators to carry out crosschain transactions. Bridge stake can also help secure the crosschain network to prevent wrongdoing in crosschain service operations. Normally the slashing of staked assets can be performed in two scenarios: one is in the case of inactivity, and another one is signing fraudulent transactions. For the inactivity case, the bridge node may be in an idle state, or not configured correctly, or it is simply shut down. For the case of signing fraudulent transactions, the bridge groups and smart contract have a way to verify the transactions from different bridge nodes and report malsigned transactions to the administrative smart contract for slashing.

Crosschain staking can be done by implementing smart contracts on a blockchain with a staking function such as `stake_deposit()` taking function inputs of: `blockchain_id`, `account_address`, `amount`, `bridge_id`, `duration`, and `signatures` to fund a bridge. Here, `blockchain_id` represents a blockchain, `account_address` is an address of the EOA (Externally Owned Account) from which funds will be withdrawn or locked to support a bridge to process cross-chain transactions, the `amount` parameter specifies the amount of the asset that will be deposited, `bridge_id` represents the bridge for which the fund will be deposited, `duration` is the length of time when the asset will be locked in the account, and `signature` is a signed message from the account owner to enhance the authenticity of the staking request.

To ensure that the staking asset can be used to compensate for the loss due to crosschain collusion, there should be a limit, i.e. bridge capacity, on the maximum amount of assets that can be transferred across blockchains. Bridge capacity can be the sum of the total staking assets deposited to a bridge. If the total value of crosschain assets exceeds the bridge capacity, the bridge is no longer fully secured, because the incentives of bridge nodes to profit from collusion increases.

Acknowledgements

The EEA acknowledges and thanks Chaals Nevile (EEA), Tas Dienes (Ethereum Foundation), Coenie Beyers (Adhara), Brittany Mauck (EEA), James Harsh (EEA), Anais Ofranc (EEA) for their contribution to the development of this version of guideline.