Bridges

Martinet Lee

/Outline

- Atomic Swap
- Classic Bridge Intro
- Attacks on Classic Bridges
- More kinds of bridges

/Atomic Swap

Allows two parties to swap assets on two different chains directly, in a trustless manner.

Suppose Alice has 1 BTC and Bob has 10 ETH and they want to trade. Alice wants to get 10 ETH and Bob wants to get 1 BTC.

Wat do?

/HTLC - Hash Time Lock Contract

Party A can lock funds in a contract, indicating a “hash”, a “time”, and a recipient “B”.

If recipient “B” reveals the secret that is the “pre-image” of the “hash”, then “B” can withdraw the funds away.

If “time” has passed, then Party A can withdraw the funds away.

/Atomic Swap

1. Alice: generate a random secret, use secret as pre-image to compute hashA
2. Alice: setup a HTLC (HTLC-A) with arguments (hashA, timeA, Bob), locks up funds to send to Bob
3. Bob: setup a HTLC (HTLC-B) with arguments (hashA, timeB, Alice), where timeB < timeA, locks up funds to send to Alice
4. Alice: interacts with HTLC-B, reveals secret within timeB to withdraw the funds Bob stored for Alice. Bob sees the secret.
5. Bob: interacts with HTLC-A with secret within TimeA to withdraw the funds Alice stored for Bob.

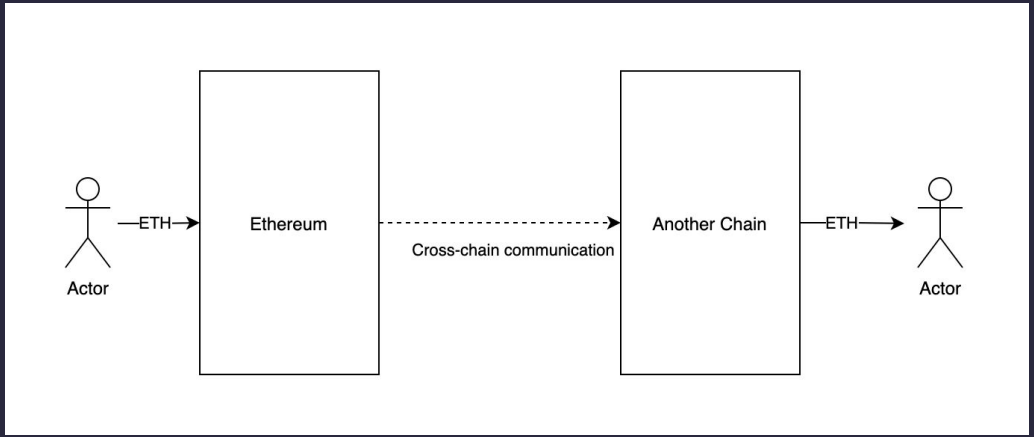
/Atomic Swap

- Discuss the cases where parties deviates.

/Bridge Intro (1)

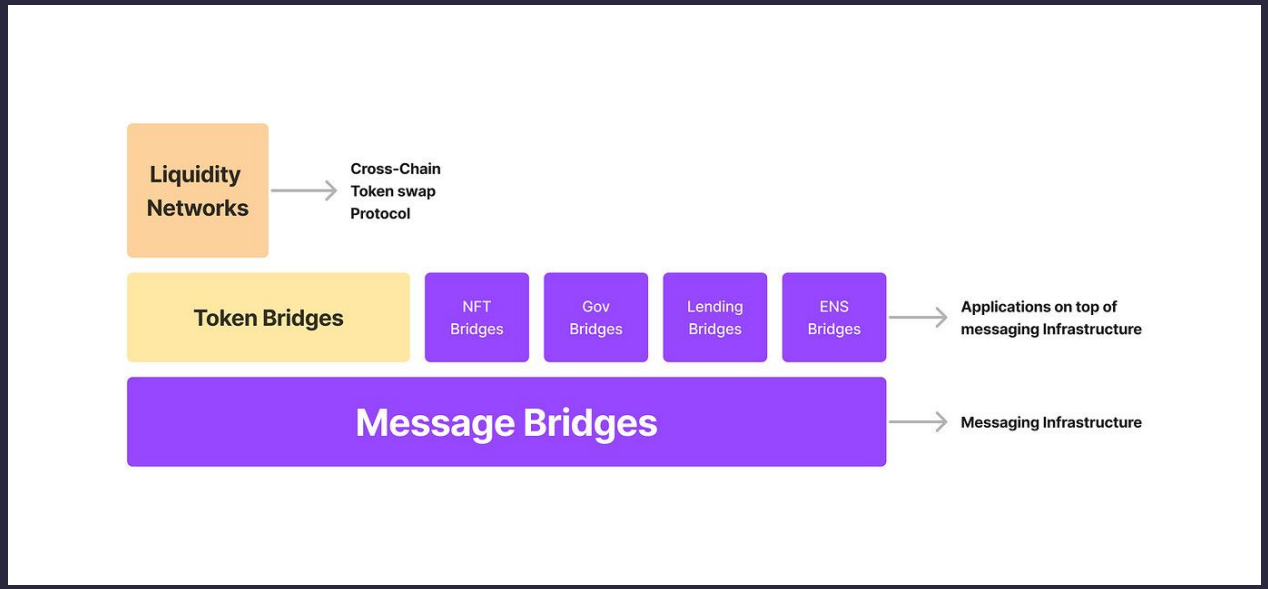
For users, bridges are used to move assets from one chain to another

However, assets cannot really "move" to another chain



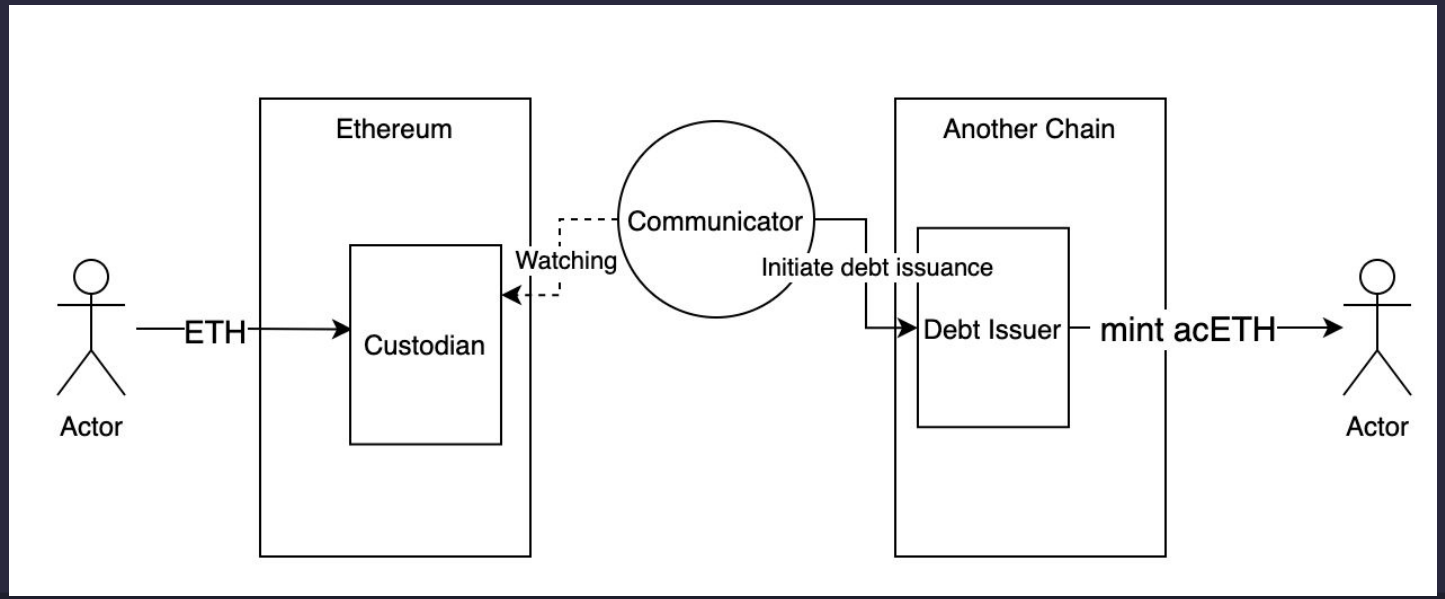
/Bridge Intro (2)

From: L2Bridge Risk Framework



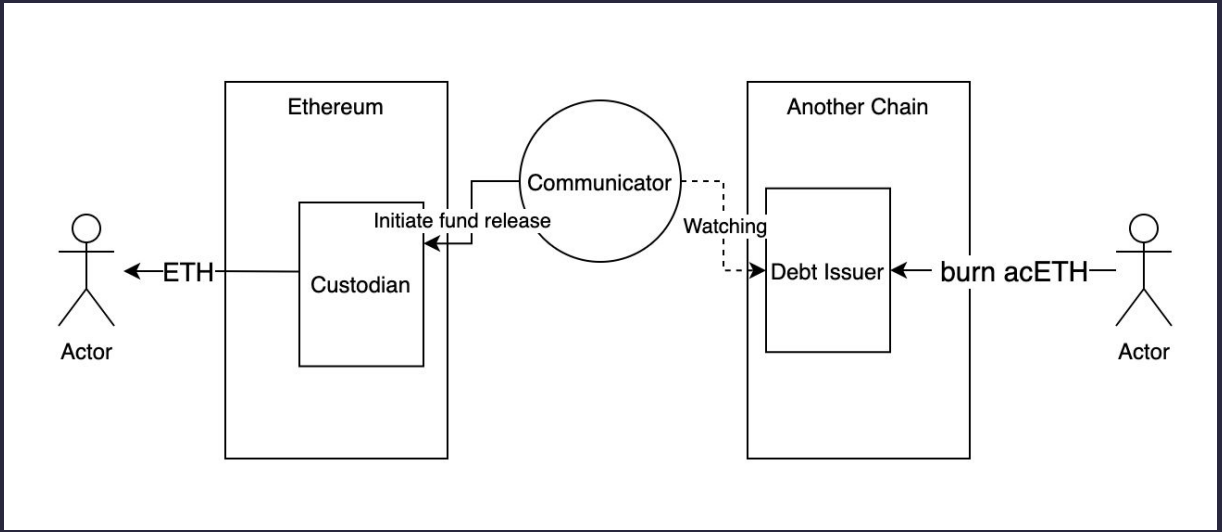
/Bridge Intro: On Deposit

Assets are being custodied on the main chain, and a form of debt token is issued to the user on the target chain



/Bridge Intro: On Redemption

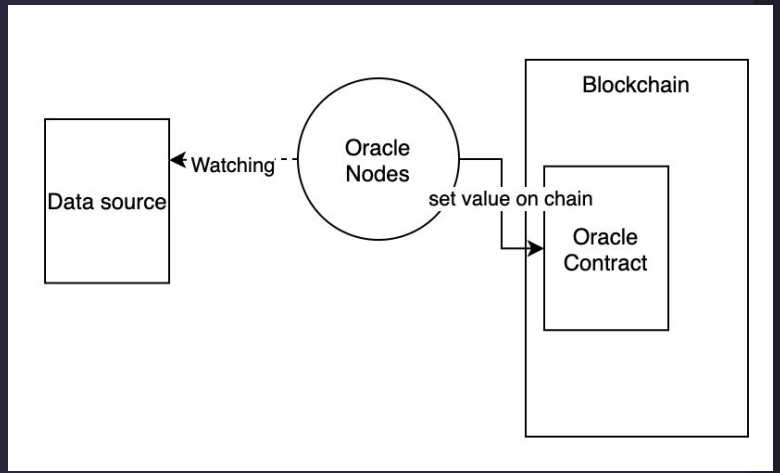
The user burns the debt token on the other chain and the communicator tells the custodian that funds can now be released



/Bridge: (Custodian + Debt issuer) + (Oracle)

If you squint hard enough, bridges are a type of project that combine two commonly seen structure in the blockchain space:

1. Asset custodian + debt issuer
2. Oracle



/The Decentralization of Bridges

Light Client on chain?

Oracle: the offchain communicator that determines whether a bridge protocol is decentralized or not.

Common ways to construct the off-chain communicator:

- Centralized entity - think USDC; WBTC; USDT; BUSD
- External Validator Set:
  - Multi-Sig
  - Consensus algorithm
  - Multi Party Computation

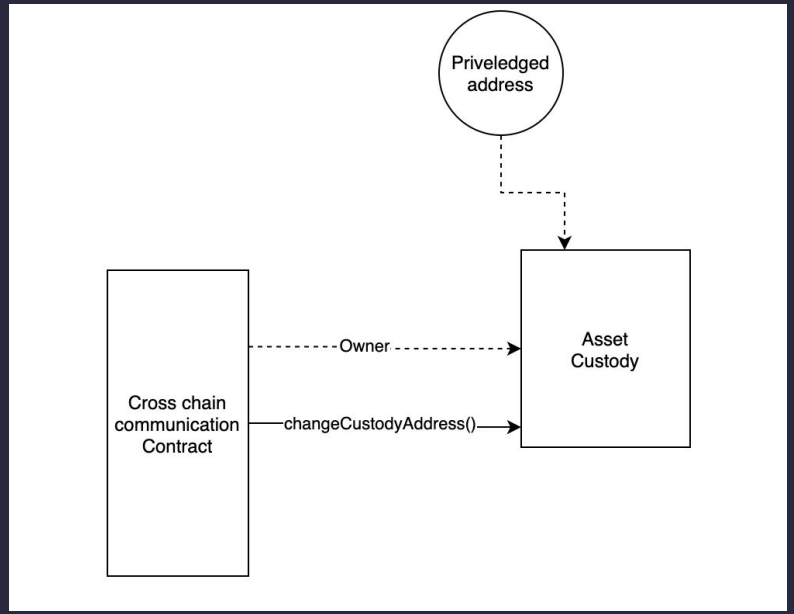
/Attacks on Custodian (1)

Depending on the structure of custodian, privileged addresses may have access to the assets

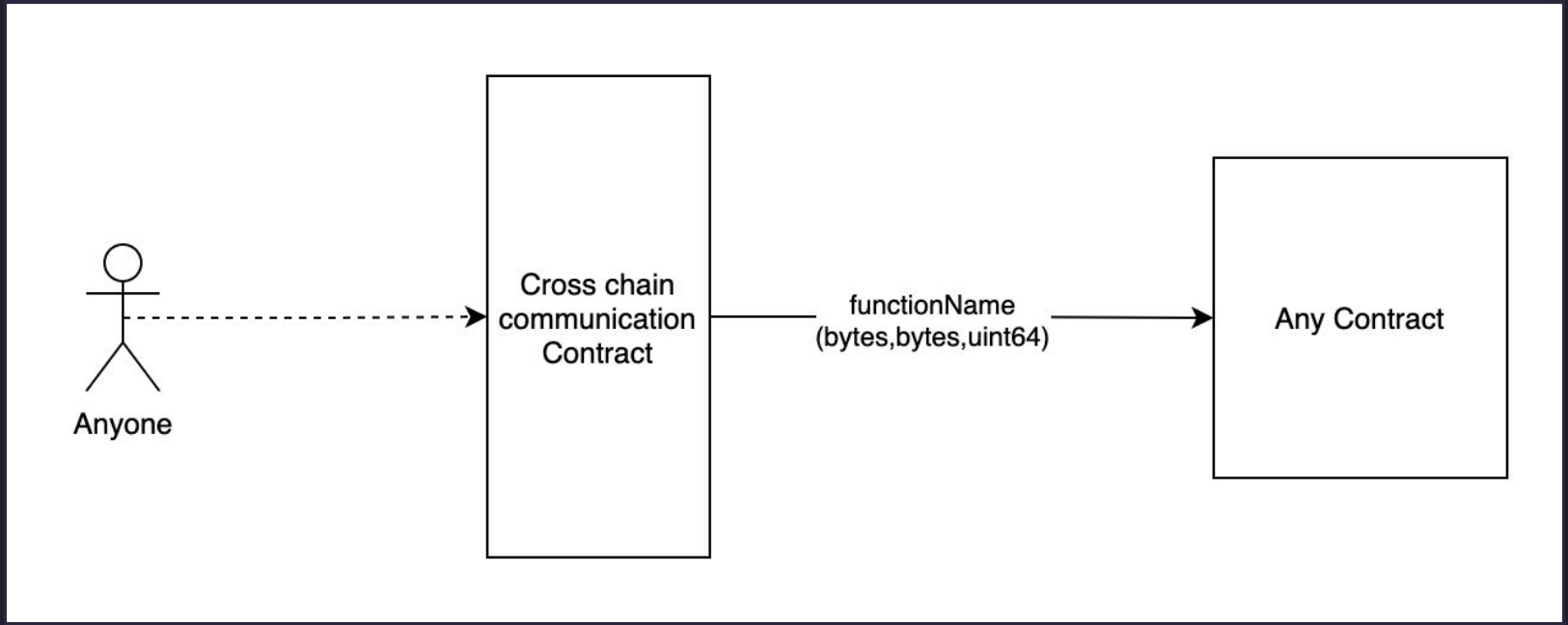
Attacker's goal:

- Take over privileged addresses
- Change the address of privileged addresses

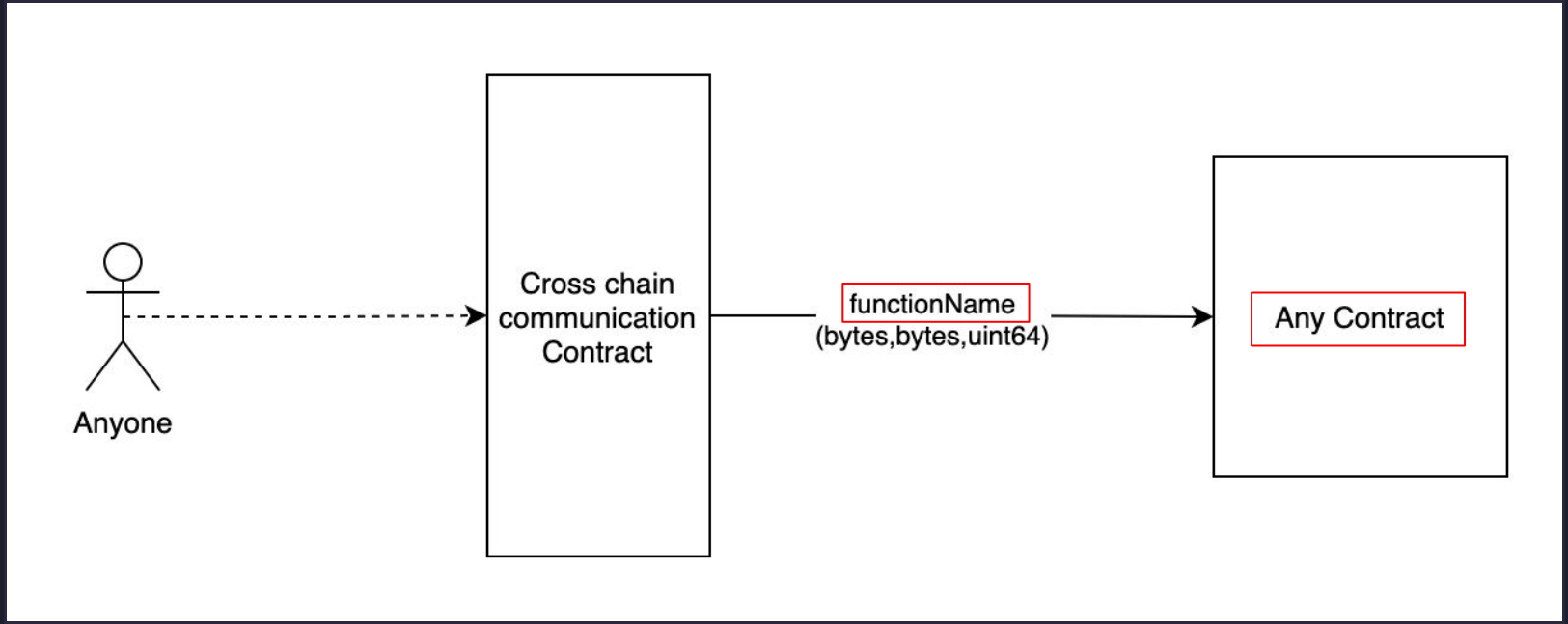
/Attacks on Custodian (1)
Example: Sensitive Structure



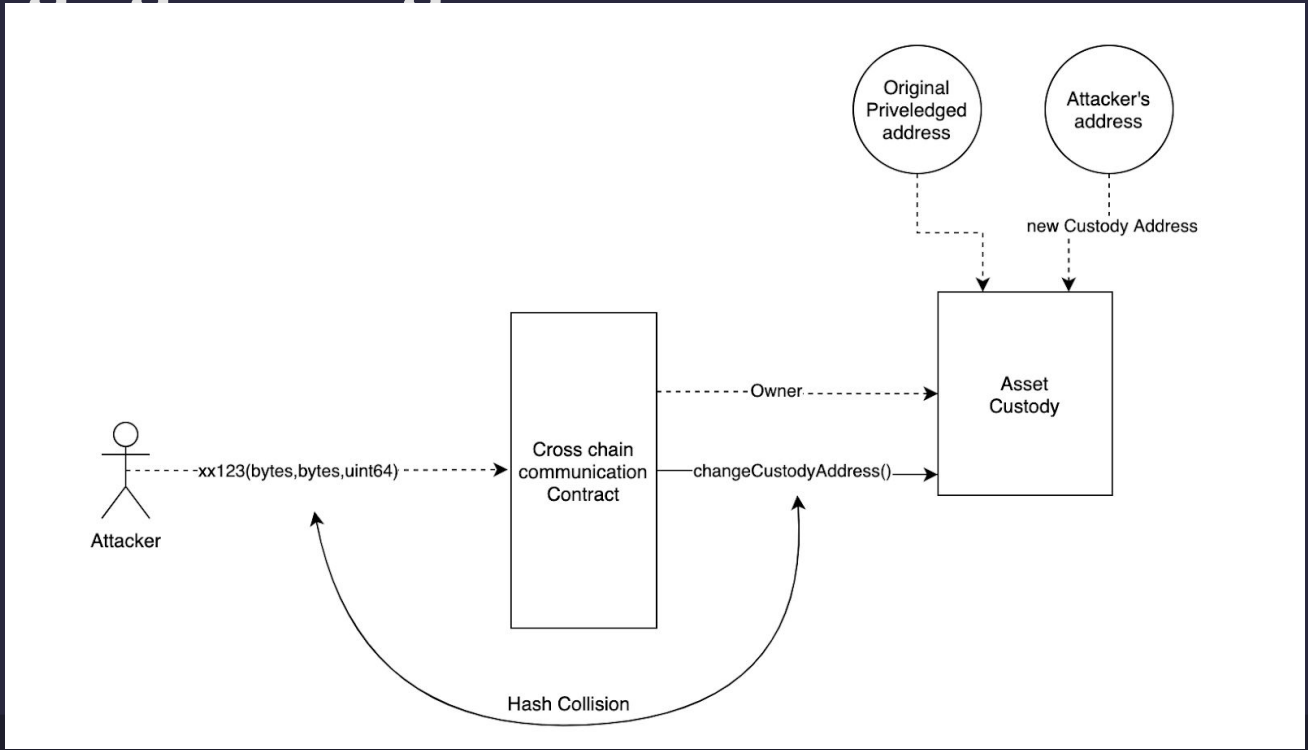
/Attacks on Custodian (1)
Example: Cross Chain Communication



/Attacks on Custodian (1)
Example: Cross Chain Communication



/Attacks on Custodian (1) Changing Privileged Address



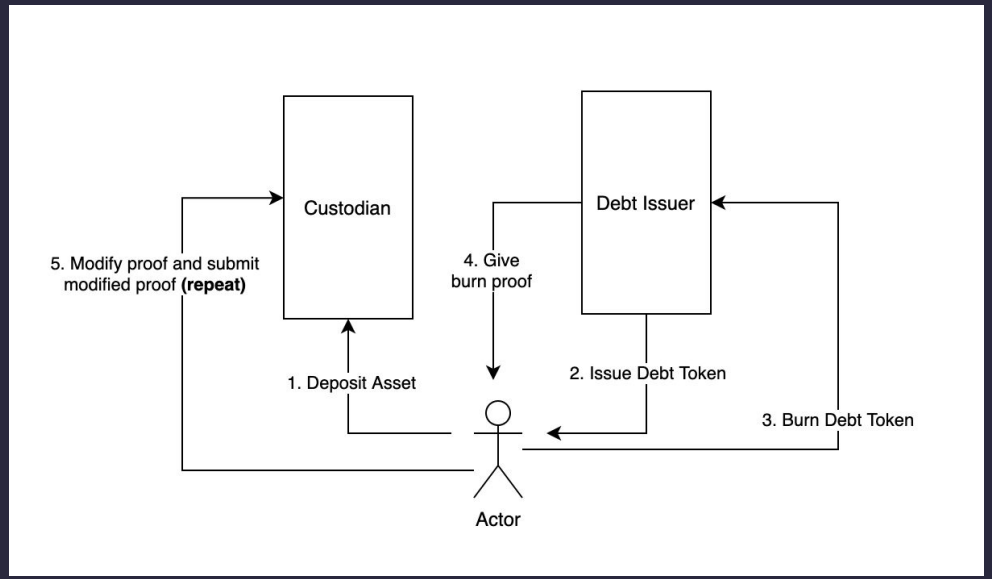
/Attacks on Custodian (2)

Depending on the structure of custodian, proofs are to be presented during redemption/withdraw

Attacker's goal:
Craft proofs that would be valid for the verification process

/Attacks on Custodian (2) Example

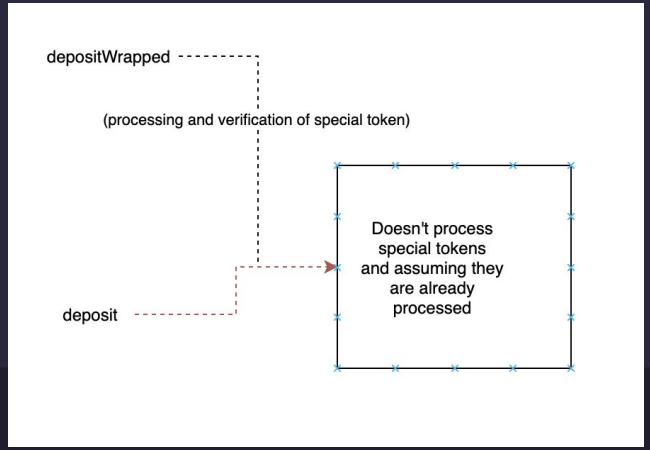
Because of flaw in the verification process, an attacker can use one valid proof to craft different proofs that would pass the verification



/Attacks on Custodian (3) Example

Attacker's goal:
Have the custodian emit deposit events when it didn't really deposit

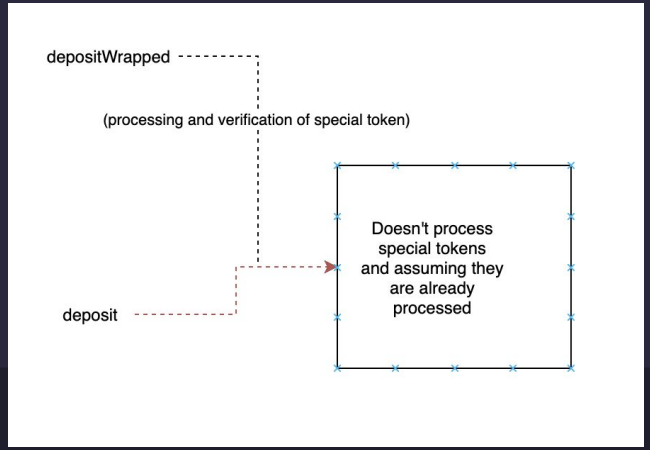
Case example:
A bridge made assumption to skip "locking" and "transferring" for a special token



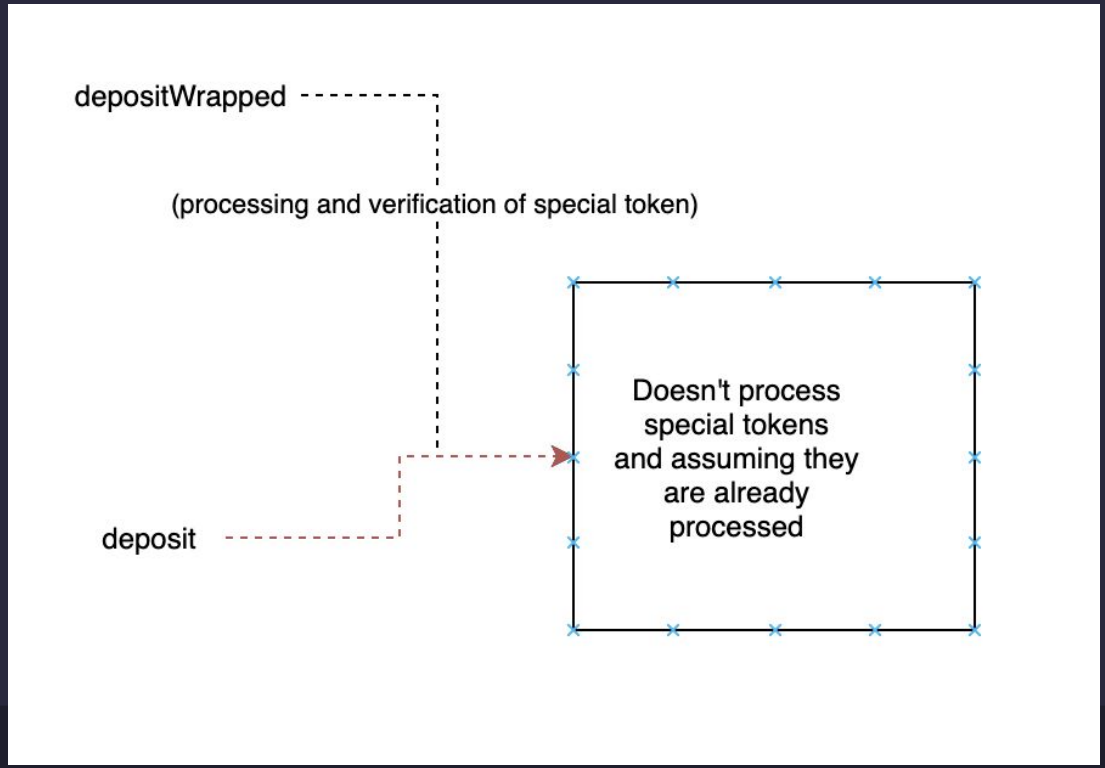
/Attacks on Custodian (3) Example

Attacker's goal:
Have the custodian emit deposit events when it didn't really deposit

Case example:
A bridge made assumption to skip "locking" and "transferring" for a special token



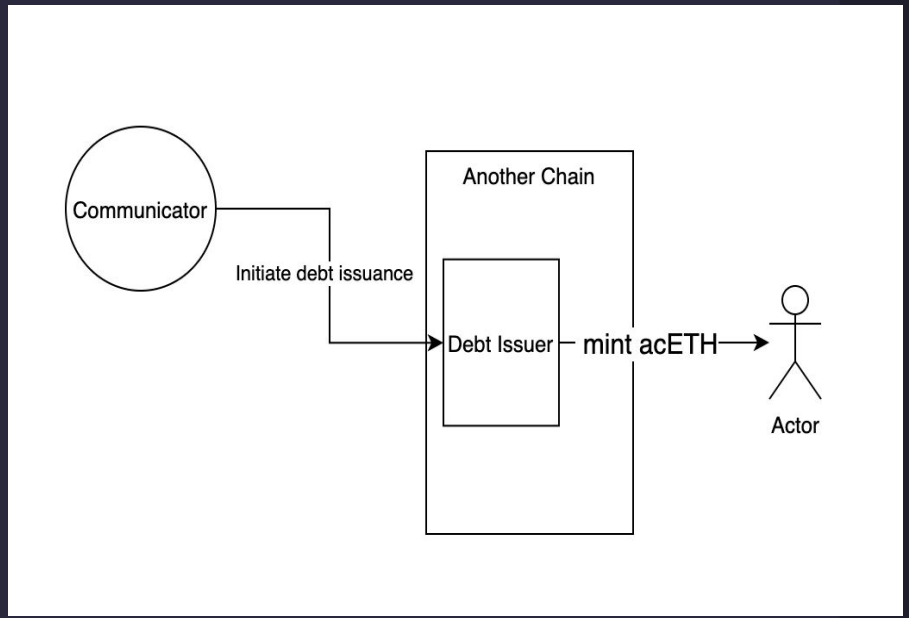
/Attacks on Custodian (3) Example



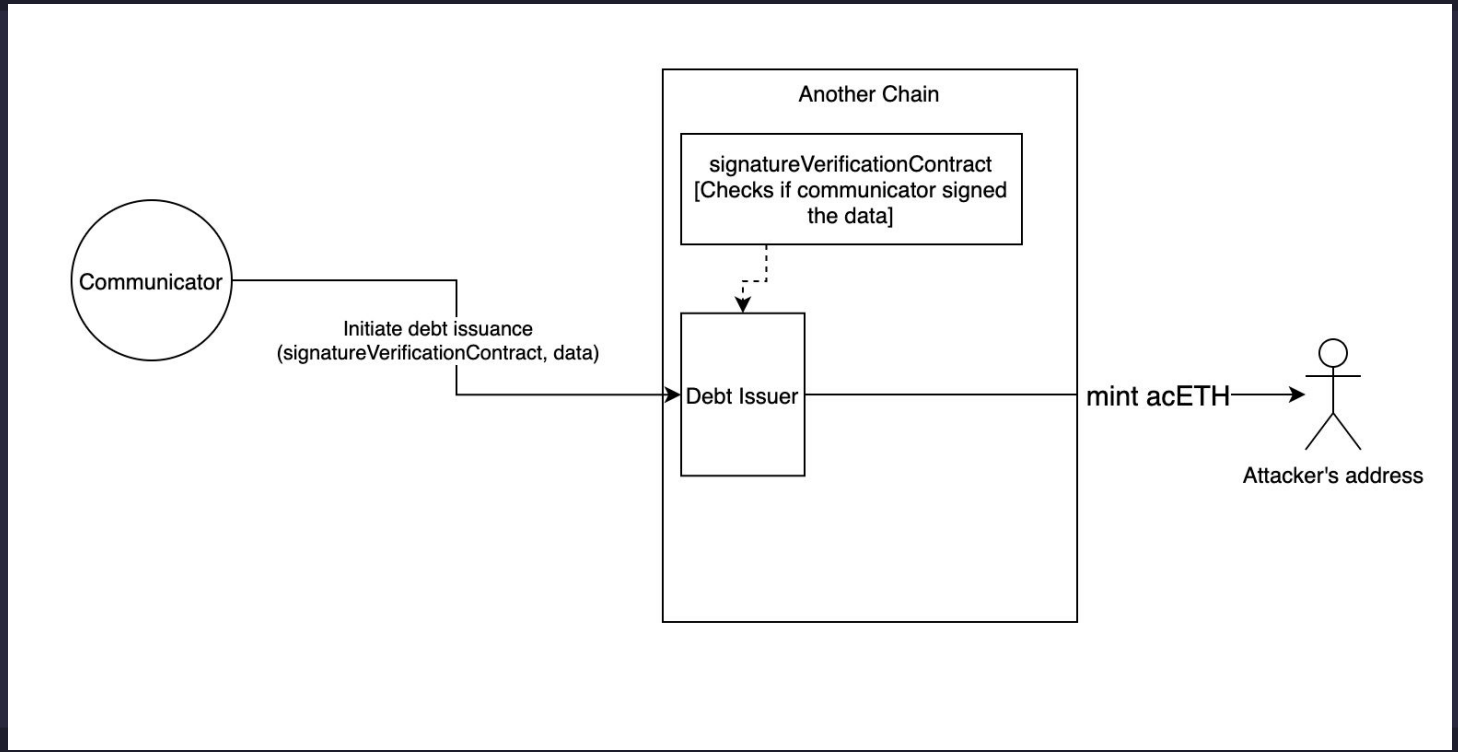
/Attacks on Debt Issuer

Attacker's goal:
Arbitrarily mint debt token

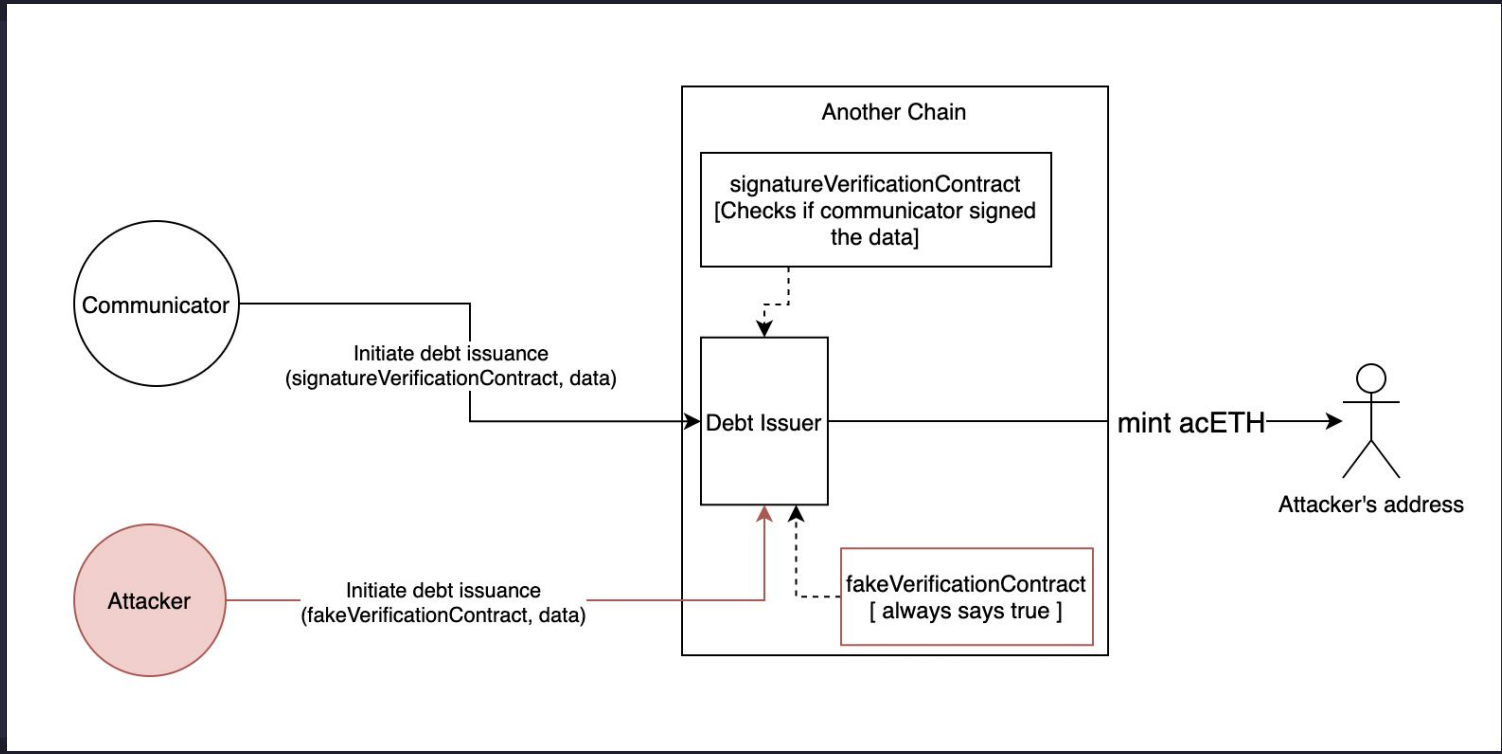
Possible route:
Bypass signature verification



/Attacks on Debt Issuer: Example



/Attacks on Debt Issuer: Example



/Attacks on Communicator (1)

Attacker's goal:
Trick the communicator into forwarding invalid messages specifically, minting debt tokens or redemption

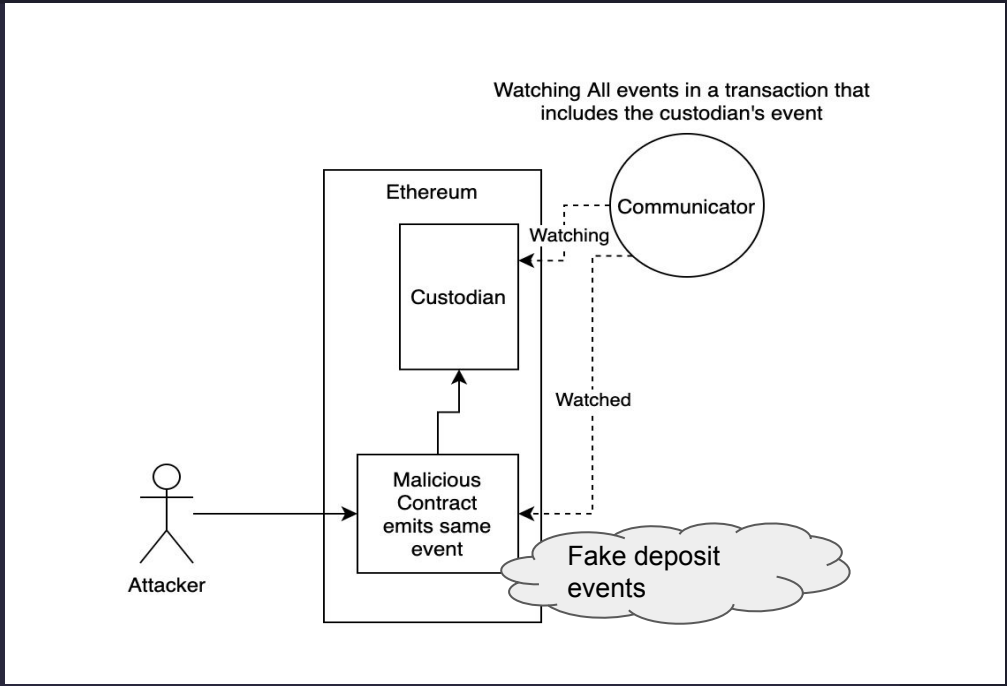
This could be thought of as polluting the data source of an oracle

/Attacks on Communicator (1) Example

Communicator has a bug where it only validates the first contract address of events

Deposit events are faked by crafting a malicious contract that emitted the same deposit event

This creates excessive debt tokens in the bridge



/Attacks on Communicator (2)

Viewing the communicator as an oracle, another way of polluting the data source of an oracle is to completely change the truth of the data source

Changing the source of truth is typically not doable, because history cannot be changed in the real world and most oracles reflect real-world data

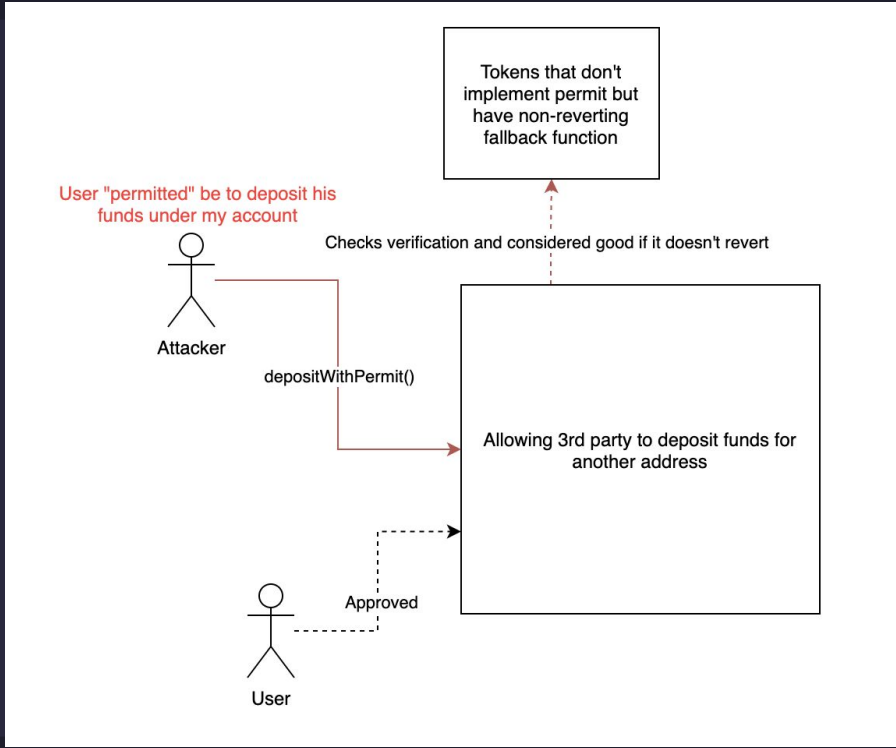
But when the source of truth is another blockchain, one can change the history by 51% attack

/Attacks on Interface (1)

Some issues on the bridge are not specific to the bridge, but pertaining to whether there's an interface for users to deposit

Example:
Contracts that allow depositing funds from other addresses with permit / allowance

"revoke approval"

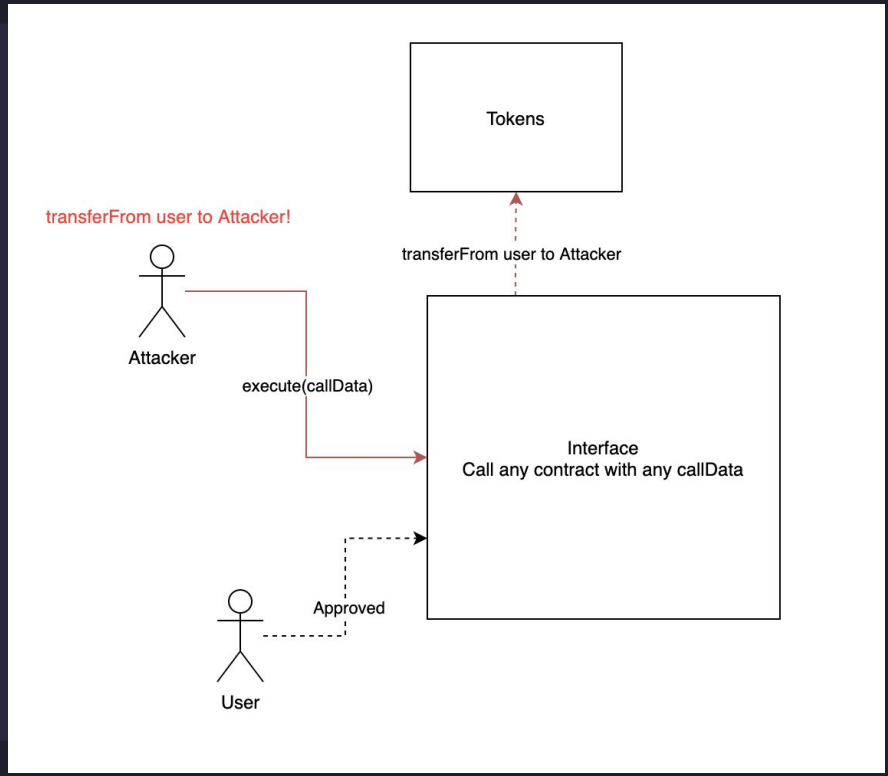


/Attacks on Interface (2)

Example:
Interface doesn't sanitize input and could execute function calls to any contract with any data

Attacker crafts transferFrom

"revoke approval"



/More kinds of bridges

- Message passing bridge
- Burn and Mint Assets (Bridge-native asset)
- Intent based bridges

/THAT'S A WRAP!  (mic drop)