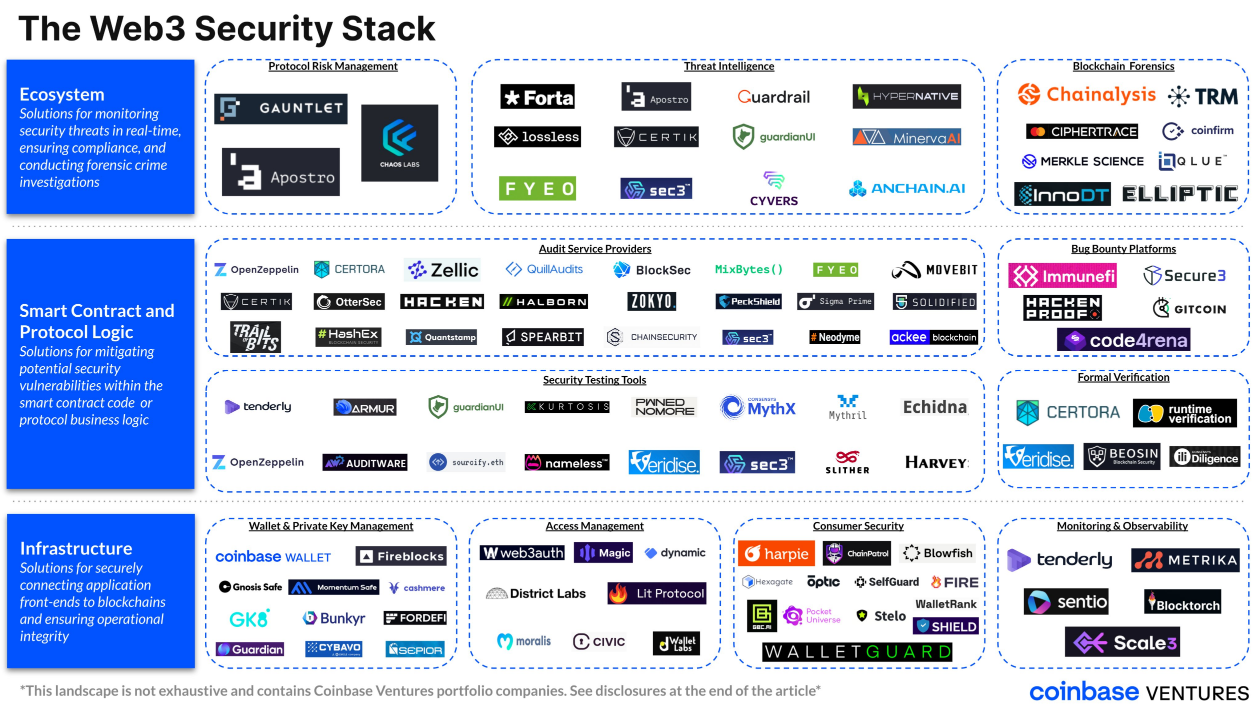
Lesson 7

Security Best Practices

Ecosystem


The Web3 Security Stack
Ecosystem
Solutions for monitoring security threats in real-time, ensuring compliance, and conducting forensic crime investigations
Protocol Risk Management
Threat Intelligence
Blockchain Forensics
Smart Contract and Protocol Logic
Solutions for mitigating potential security vulnerabilities within the smart contract code or protocol business logic
Audit Service Providers
Bug Bounty Platforms
Security Testing Tools
Formal Verification
Infrastructure
Solutions for securely connecting application front-ends to blockchains and ensuring operational integrity
Wallet & Private Key Management
Access Management
Consumer Security
Monitoring & Observability
coinbase VENTURES

This landscape is not exhaustive and contains Coinbase Ventures portfolio companies. See disclosures at the end of the article

Consensys Best Practices

General

- Prepare for Failure
- Stay up to Date
- Keep it Simple
- Rolling out
- Blockchain Properties
- Simplicity vs. Complexity

Precautions

- General
- Upgradeability
- Circuit Breakers

- Speed Bumps
- Rate Limiting
- Deployment
- Safe Haven

Solidity Specific

- Assert, Require, Revert
- Modifiers as Guards
- Integer Division
- Abstract vs Interfaces
- Fallback Functions
- Payability
- Visibility
- Locking Pragmas
- Event Monitoring
- Shadowing
- tx.origin
- Timestamp Dependence
- Complex Inheritance
- Interface Types
- EXTCODESIZE Checks

Token Specific

- Standardization
- Frontrunning
- Zero Address
- Contract Address

Documentation

- General
- Specification
- Status
- Procedures
- Known Issues

- History
- Contact

Attacks

- Reentrancy
- Oracle Manipulation
- Frontrunning
- Timestamp Dependence
- Insecure Arithmetic
- Denial of Service
- Griefing
- Force Feeding

Development recommendations

Token Checklist

Solidity Bugs by version

PolyNetwork Hack

Encoding the function signatures and parameters

Example

```
pragma solidity ^0.8.0;

contract MyContract {

  Foo otherContract;


  function callOtherContract() public view returns (bool){
      bool answer = otherContract.baz(69,true);
      return answer;
  }
}



contract Foo {
    function bar(bytes3[2] memory) public pure {}
    function baz(uint32 x, bool y) public pure returns (bool r)
{
    r = x > 32 || y;
    }
    function sam(bytes memory, bool, uint[] memory) public pure
{}
}
```

The way the call is actually made involves encoding the function selector and parameters

If we wanted to call ***baz*** with the parameters **69** and **true** , we would pass 68 bytes total, which can be broken down into:

1. the Method ID. This is derived as the first 4 bytes of the Keccak hash of the ASCII form of the signature baz(uint32,bool).

```
***0xcdcd77c0:***
```

2. the first parameter, a uint32 value 69 padded to 32 bytes

   0×0000000000000000000000000000000000000000000000000 0000000000000000000045:

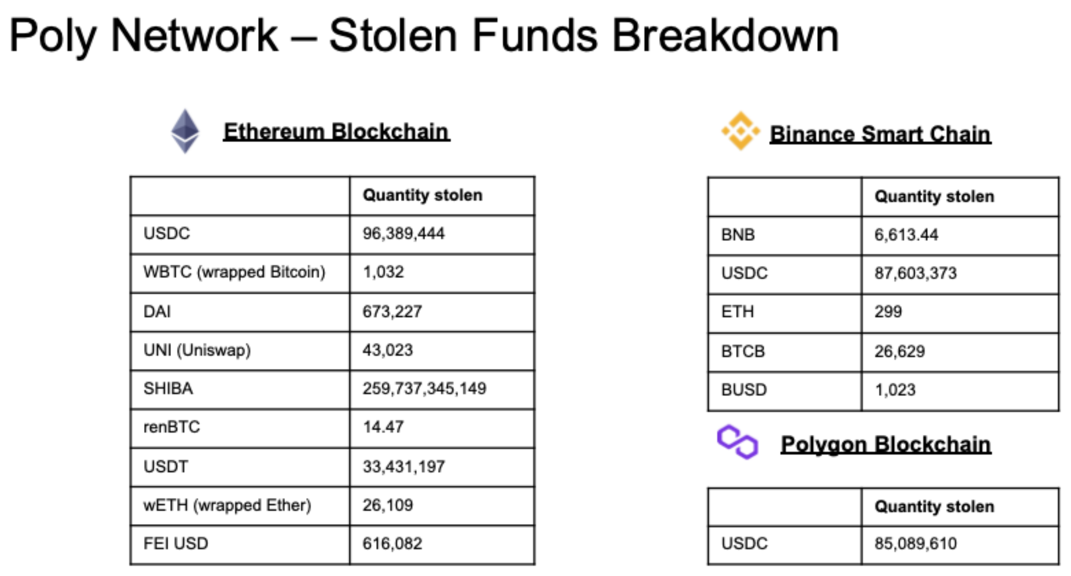
3. the second parameter - boolean true, padded to 32 bytes

   0×0000000000000000000000000000000000000000000000000 0000000000000000000001:

In total

***0xcdcd77c0000000000000000000000000000000000000000000000
0
000000000000000000000045000000000000000000000000000000
00
0000000000000000000000000000000000000001***

The Polynetwork Hack - $600M stolen



Poly Network – Stolen Funds Breakdown

Ethereum Blockchain

	Quantity stolen
USDC	96,389,444
WBTC (wrapped Bitcoin)	1,032
DAI	673,227
UNI (Uniswap)	43,023
SHIBA	259,737,345,149
renBTC	14.47
USDT	33,431,197
wETH (wrapped Ether)	26,109
FEI USD	616,082

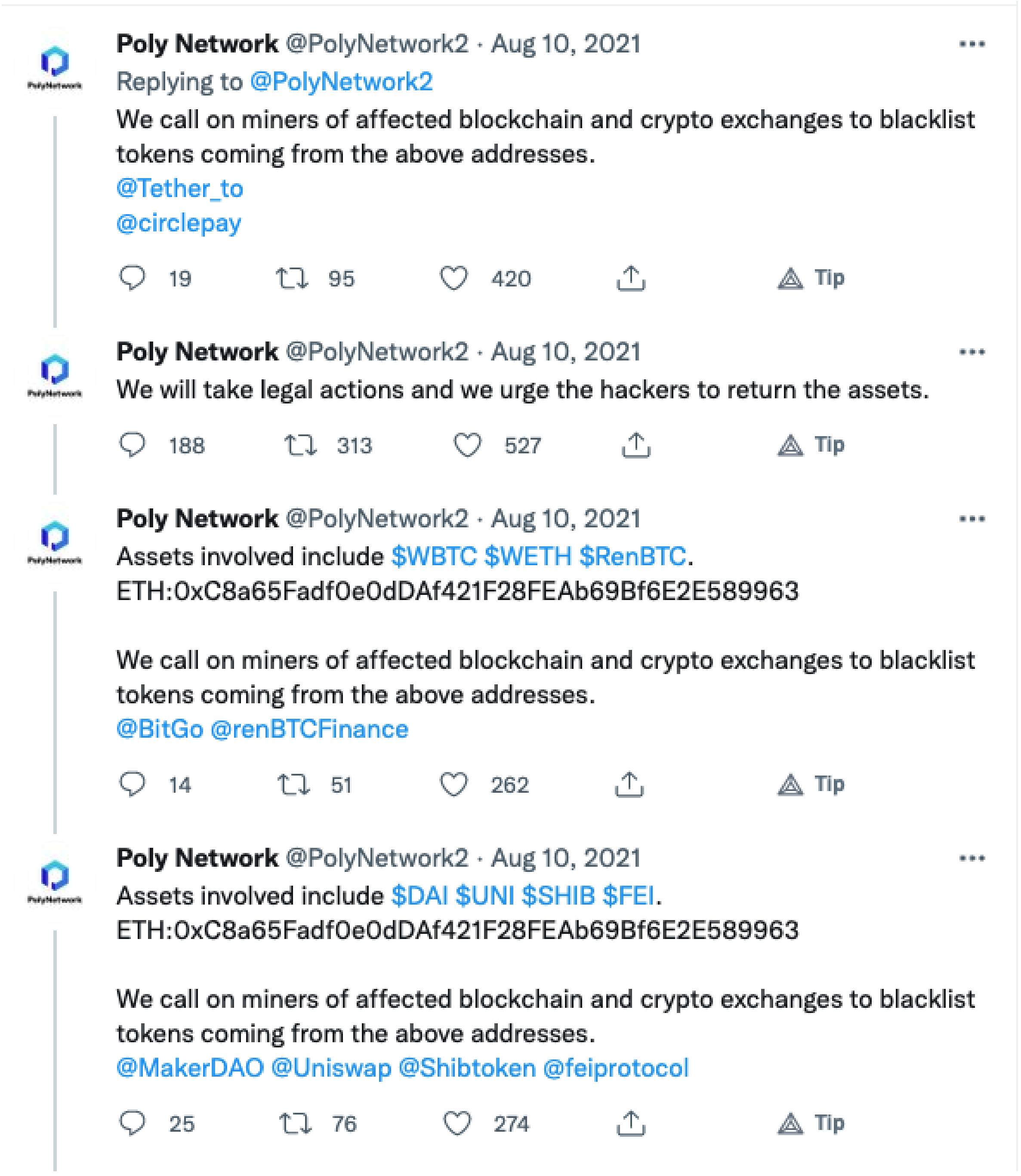
Binance Smart Chain

	Quantity stolen
BNB	6,613.44
USDC	87,603,373
ETH	299
BTCB	26,629
BUSD	1,023

Polygon Blockchain

	Quantity stolen
USDC	85,089,610

Polynetwork react to attack


Poly Network @PolyNetwork2 · Aug 10, 2021
Replying to @PolyNetwork2
We call on miners of affected blockchain and crypto exchanges to blacklist tokens coming from the above addresses.
@Tether_to
@circlepay

19 95 420 Tip

Poly Network @PolyNetwork2 · Aug 10, 2021
We will take legal actions and we urge the hackers to return the assets.

188 313 527 Tip

Poly Network @PolyNetwork2 · Aug 10, 2021
Assets involved include $WBTC $WETH $RenBTC.
ETH:0xC8a65Fadf0e0dDAf421F28FEAb69Bf6E2E589963

We call on miners of affected blockchain and crypto exchanges to blacklist tokens coming from the above addresses.
@BitGo @renBTCFinance

14 51 262 Tip

Poly Network @PolyNetwork2 · Aug 10, 2021
Assets involved include $DAI $UNI $SHIB $FEI.
ETH:0xC8a65Fadf0e0dDAf421F28FEAb69Bf6E2E589963

We call on miners of affected blockchain and crypto exchanges to blacklist tokens coming from the above addresses.
@MakerDAO @Uniswap @Shibtoken @feiprotocol

25 76 274 Tip

Example messages to the attacker

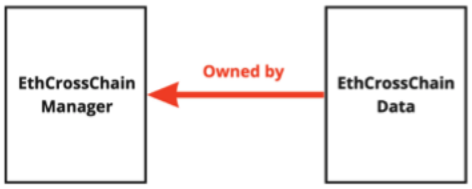
- Can I have some ETH? i been losing a lot of money Thank you
- Consider donating to public goods after all the pleasure you've got from using public infra 🌱
- ngmi
- Welcome to the crypto world. Fee Tips: Use tornado.cash to laundry your money ASAP.
- Forget your USDT. Forget your USDC.
- Swap tokens to ETH then deposit to tornado.cash . Good luck.
- Hi. Boos. Can. You. Give me eth thanks

- You can use Tornado for currency mixing
- DONT USE YOUR USDT TOKEN YOU VE GOT BLACKLISTED
- You can buy every pudgy penguin on opensea :)
- Dad, this is my only asset. Please accept it
- JOINING FOR EPIC SCREENCAP
- Please gice me some eth
- DONT USE YOUR USDT TOKEN
  YOU VE GOT BLACKLISTED,god bless you
- Remove liquidity from curve pool in form of DAI,
  and exchange it for Eth and launder with Tornado

Exploit Details

Mismanagement of access rights between two contracts

EthCrossChainManager is an owner of EthCrossChainData,
= EthCrossChainManager can execute privileged functions!



_method is user defined
= can be set at will.

```
bytes4(keccak256(abi.encodePacked(_method, "(bytes,bytes,uint64)"))),
```

5

Vulnerable Contract 2: `EthCrossChainData`

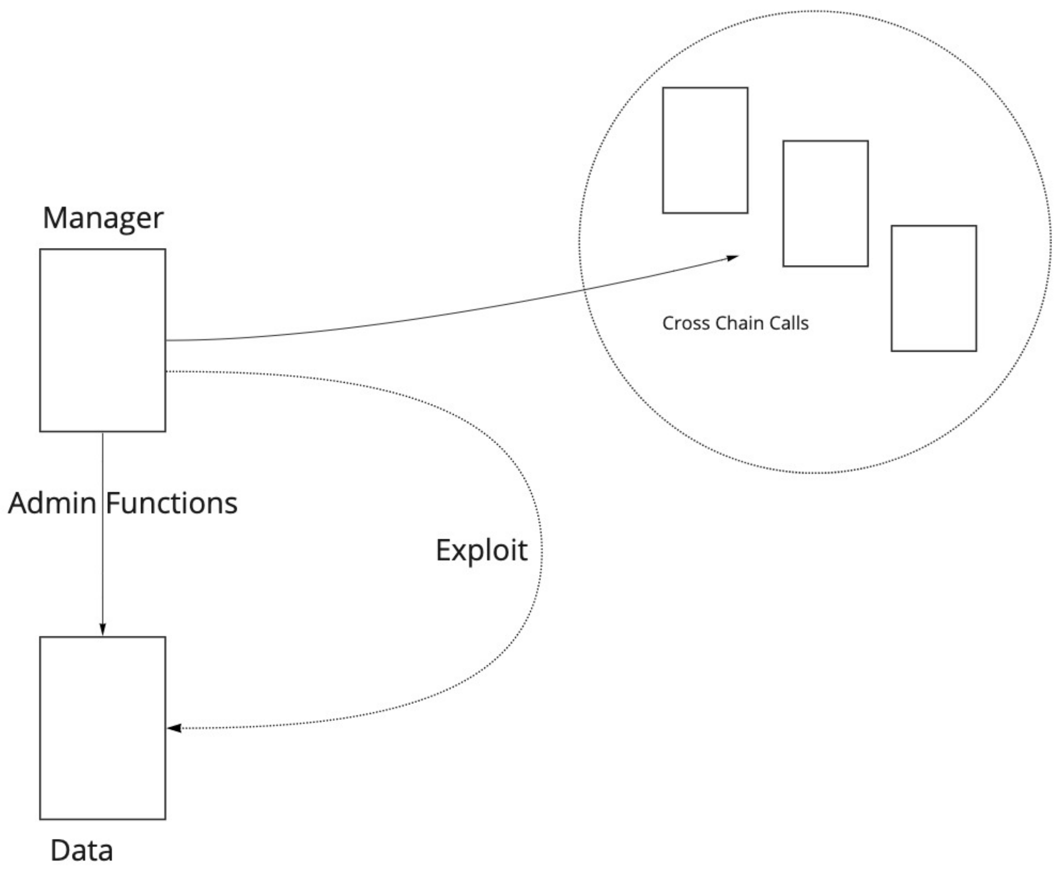
Very High Privileged contract ! ➔ Can only be called by its owners.

Set + manage list of **"Keepers"**
= list of public keys that manage the wallets in the underlying liquidity chain

➔ **Keepers have the right to execute large transactions,** transfer large amounts to other wallets.

Vulnerable function: *putCurEpochConPubKeyBytes*
= become a **"Keeper"**
Set the public key (passed as parameter) as a Keeper

10


Manager
Cross Chain Calls
Admin Functions
Exploit
Data

The attacker could change the contract address, the tricky part was to make the method ID in the call match the target

The target is
Hash(putCurEpochConPubKeyBytes,bytes) = 0×41973cd9

The manager contract will create a call using a method ID calculated as Hash(X,bytes,bytes,uint64)

where X can be supplied by the user calling the contract.

So

The attacker was aiming to find X such that

Hash(X,bytes,bytes,uint64)

=

Hash(putCurEpochConPubKeyBytes,bytes) = 0×41973cd9

The Brute Force solution was

X = f1121318093

The Attacker reconsiders

The attacker received a rather cryptic message
” Dont instant tornado funds, dont instant move blacklistable tokens to DAI/ETH? Insider confirmed!”

The attacker was looking at Tornado Cash and sent themselves a message

”Wonder why Tornado? Will miners stop me? Teach me please”

Then someone found a link between an address used by the attacker and some exchanges and tweeted

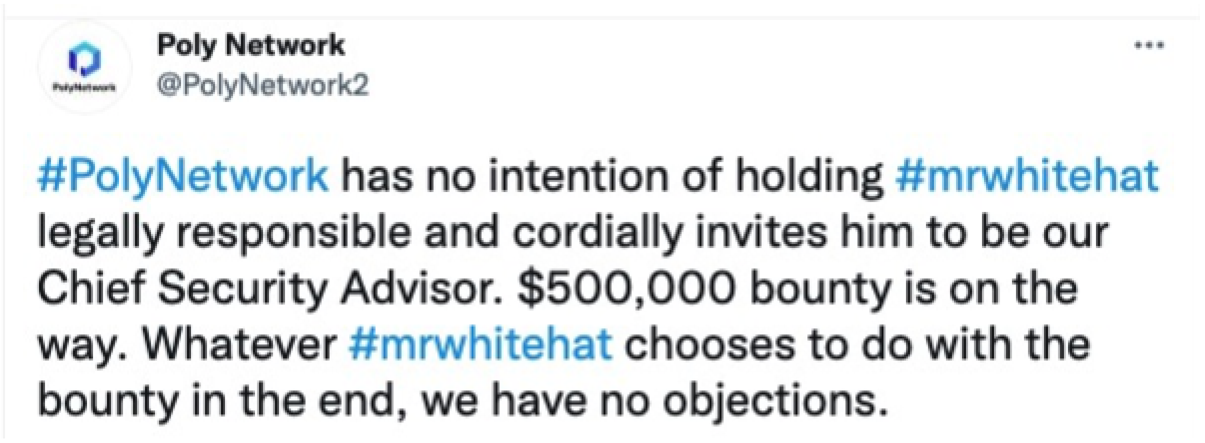
“Did the PolyNetwork Exploiter accidentally use the wrong sender address for this tx 0xb12681d9e? The sender address is tied to FTX, Binance, Okex accounts.”

The hacker’s attitude started to change, he suggested he could return "some tokens" or even abandon them, saying that they were "not so interested in the money".

Followed by a suggestion : “What if I make a new token and let the DAO decide where the tokens go”

Finally the attacker messaged ”Ready to return the fund !”

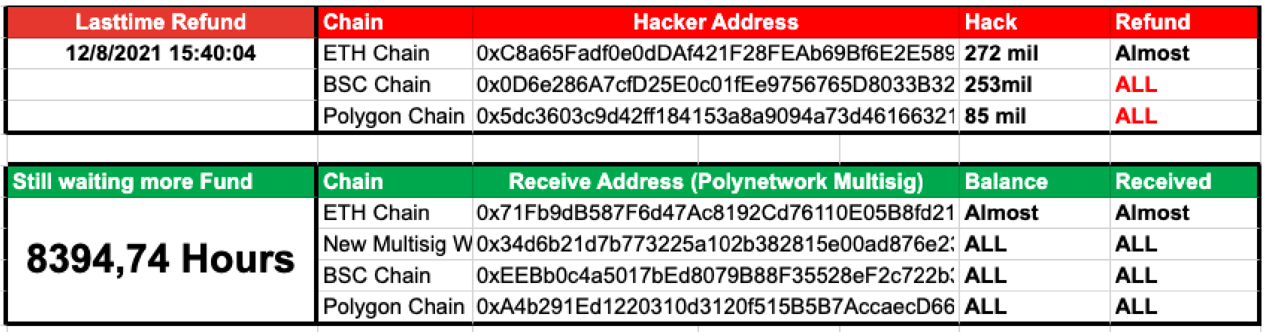
See this spreadsheet for all communications
The attacker starts to return the funds
Polynetwork starts to refer to the attacker as ‘Mr White Hat’ and offer him a job and bounty



Poly Network
@PolyNetwork2

#PolyNetwork has no intention of holding #mrwhitehat legally responsible and cordially invites him to be our Chief Security Advisor. $500,000 bounty is on the way. Whatever #mrwhitehat chooses to do with the bounty in the end, we have no objections.

'We appreciate you sharing your experience and believe your action constitutes white hat behavior. But we can't touch user assets and Poly Network doesn't have its own token. Since , we believe your action is white hat behavior, we plan to offer you a $500,000 bug bounty after you complete the refund fully. Also we assure you that you will not be accountable for this incident.
We hope that you can return all tokens as soon as possible. You can reserve the equivalent value of 500,000 USD in any assets to the current owner address. We will make up this part of the assets to Poly Network users.
Your contribution is very helpful to us. Again, we think this behavior is white hat behavior, therefor this 500,000 USD will be seen as completely legal bounty reward. We will also ensure that you will not be held accountable for this incident, and we will publicly express our gratitude to you.'



Lasttime Refund	Chain	Hacker Address	Hack	Refund
12/8/2021 15:40:04	ETH Chain	0xC8a65Fadf0e0dDAf421F28FEAb69Bf6E2E589	272 mil	Almost
	BSC Chain	0x0D6e286A7cfD25E0c01fEe9756765D8033B32	253mil	ALL
	Polygon Chain	0x5dc3603c9d42ff184153a8a9094a73d46166321	85 mil	ALL

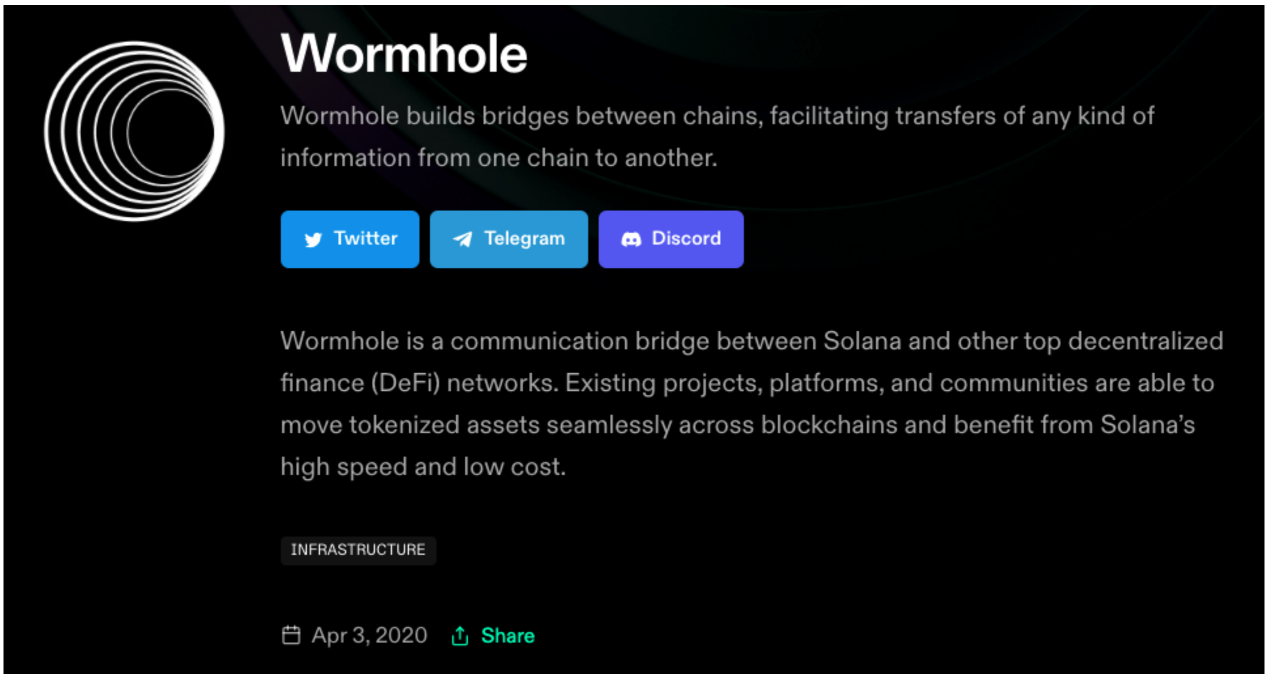
Still waiting more Fund	Chain	Receive Address (Polynetwork Multisig)	Balance	Received
8394,74 Hours	ETH Chain	0x71Fb9dB587F6d47Ac8192Cd76110E05B8fd21	Almost	Almost
	New Multisig W	0x34d6b21d7b773225a102b382815e00ad876e2	ALL	ALL
	BSC Chain	0xEEBb0c4a5017bEd8079B88F35528eF2c722b	ALL	ALL
	Polygon Chain	0xA4b291Ed1220310d3120f515B5B7AccaecD66	ALL	ALL

A number of rather lengthy messages follow the return of the funds

```
Q & A, PART TWO:

Q: WHAT REALLY HAPPENED 30 HOURS AGO?
A: LONG STORY.

BELIEVE IT OR NOT, I WAS _FORCED_ TO PLAY THE GAME.

THE POLY NETWORK IS A SOPHISTICATED SYSTEM, I DIDN'T MANAGE TO BUILD A LOCAL TESTING ENVIRONMENT. I FAILED TO
PRODUCE A POC AT THE BEGINNING. HOWEVER, THE AHA MOMEMNT CAME JUST BEFORE I WAS TO GIVE UP. AFTER DEBUGGING ALL
NIGHT, I CRAFTED A _SINGLE_ MESSAGE TO THE ONTOLOGY NETWORK.

I WAS PLANNING TO LAUNCH A COOL BLITZKRIEG TO TAKE OVER THE FOUR NETWORK: ETH, BSC, POLYGON & HECO. HOWEVER THE HECO
NETWORK GOES WRONG! THE RELAYER DOES NOT BEHAVE LIKE THE OTHERS, A KEEPER JUST RELAYED MY EXPLOIT DIRECTLY, AND THE
KEY WAS UPDATED TO SOME WRONG PARAMETERS. IT RUINED MY PLAN.

I SHOULD HAVE STOPPED AT THAT MOMENT, BUT I DECIDED TO LET THE SHOW GO ON! WHAT IF THEY PATCH THE BUG SECRETLY
WITHOUT ANY NOTIFICATION?

HOWEVER, I DIDN'T WANT TO CAUSE _REAL_ PANIC OF THE CRYPTO WORLD. SO I CHOSE TO IGNORE SHIT COINS, SO PEOPLE DIDN'T
HAVE TO WORRY ABOUT THEM GOING TO ZERO. I TOOK IMPORTANT TOKENS (EXCEPT FOR SHIB) AND DIDN'T SELL ANY OF THEM.

Q: THEN WHY SELLING/SWAPPING THE STABLES?
A: I WAS PISSED BY THE POLY TEAM FOR THEIR INITIAL REPONSE.

THEY URGED OTHERS TO BLAME & HATE ME BEFORE I HAD ANY CHANCE TO REPLY! OF COURSE I KNEW THERE ARE FAKE DEFI COINS,
BUT I DIDN'T TAKE IT SERIOUSLY SINCE I HAD NO PLAN LAUNDERING THEM.

IN THE MEANWHILE, DEPOSITING THE STABLES COULD EARN SOME INTEREST TO COVER POTENTIAL COST SO THAT I HAVE MORE TIME
TO NEGOTIATE WITH THE POLY TEAM.
```

Wormhole attack

See our post mortem article



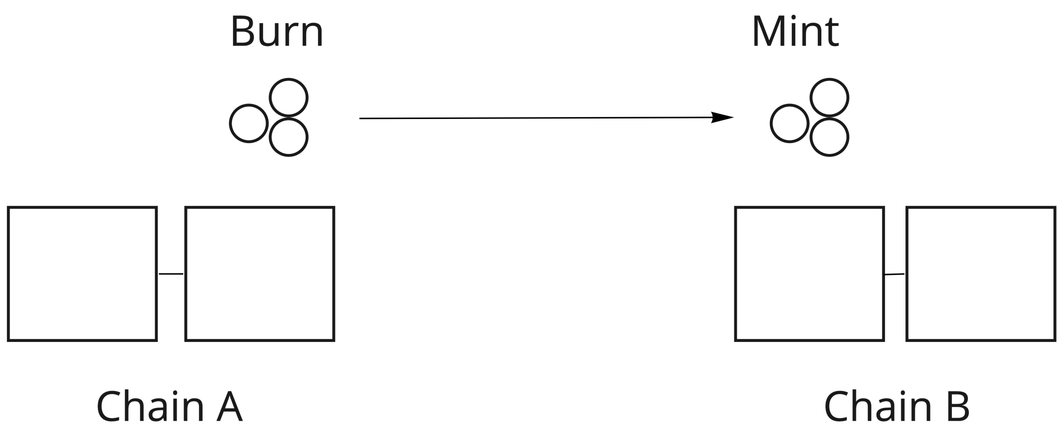
Wormhole

Wormhole builds bridges between chains, facilitating transfers of any kind of information from one chain to another.

Wormhole is a communication bridge between Solana and other top decentralized finance (DeFi) networks. Existing projects, platforms, and communities are able to move tokenized assets seamlessly across blockchains and benefit from Solana's high speed and low cost.

Wormhole is a "bridge", basically a way to move crypto assets between different blockchains.

Bridges between blockchains



Bridges like Wormhole work by having two smart contracts — one on each chain. In this case, there was one smart contract on Solana and one on Ethereum.
A bridge like Wormhole takes an ethereum token, locks it into a contract on one chain, and then on the chain at the other side of the bridge, it issues a parallel token.

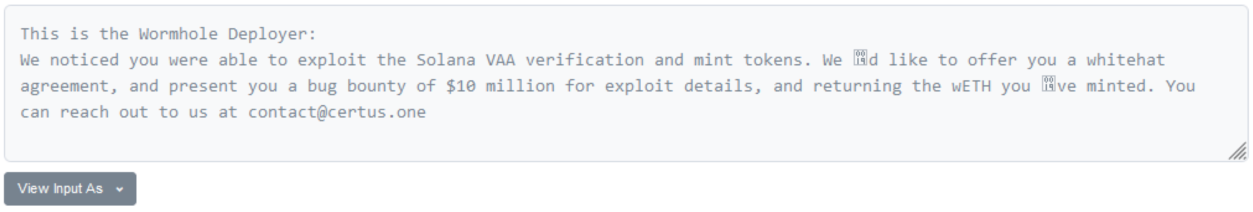
This process needs to be authorised, Wormhole has a set of "guardians" that sign off on transfers between chains.

The affected bridge was between ETH on the Ethereum blockchain and wormhole ETH (whETH) on Solana
It acted as an escrow service holding a 1:1 ration of each token.

Attack - February 2nd 2022

The attacker managed to make it look as if 120k ETH had been deposited on Ethereum, allowing the attacker to mint the equivalent in wrapped whETH (Wormhole ETH) on Solana.

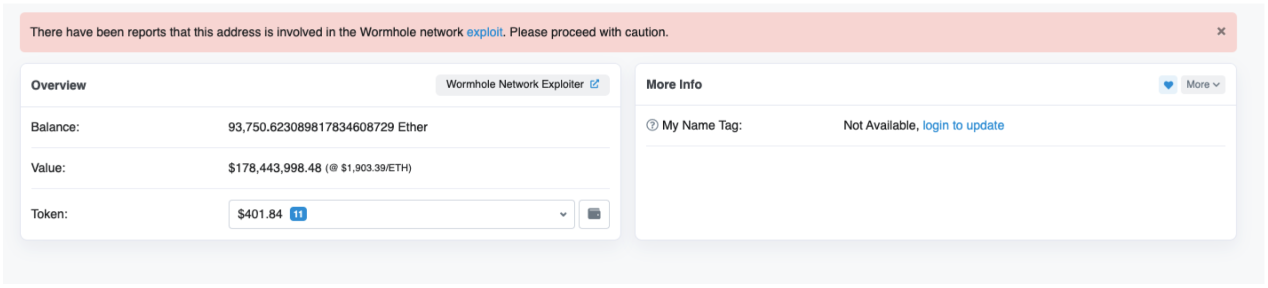
Shortly after the hack, an on-chain message was sent to the hacker from Certus One, the team behind the Wormhole bridge:



```
This is the Wormhole Deployer:
We noticed you were able to exploit the Solana VAA verification and mint tokens. We �d like to offer you a whitehat agreement, and present you a bug bounty of $10 million for exploit details, and returning the wETH you �ve minted. You can reach out to us at contact@certus.one
```

However the hacker didn't make any contact, instead 93,750 ETH was bridged back to Ethereum over the course of 3 transactions, where it still remains in the hacker's wallet.

Ethereum Wallet



There have been reports that this address is involved in the Wormhole network exploit. Please proceed with caution.

Overview

Balance:	93,750.623089817834608729 Ether
Value:	$178,443,998.48 (@ $1,903.39/ETH)
Token:	$401.84 11

More Info

My Name Tag:	Not Available, login to update

The remaining ~36k whETH were liquidated on Solana into USDC and SOL.

This exploit broke the 1:1 peg by stealing 120k ETH from the collateral reserve.
At the time ETH wrapped by Wormhole comprises roughly 19% of the total ETH on the blockchain; if the 1:1 backing of whETH hadn't been replenished swiftly, it could have triggered a situation where DeFi positions may become undercollateralized and potentially fuel a bank run.

Jump Crypto, the cryptocurrency fund which has raised more than $700 million in capital, tweeted that it had "replaced" 120,000 stolen Ethereum-based tokens "to make community members whole" and support Wormhole but provided no further details about the bailout.
Wormhole confirmed on Twitter that funds involved in the hack had indeed been "restored" and wrote "all funds are safe" on Telegram.

Exploit details

The exploit usd the complete_wrapped function, which gets triggered whenever someone mints whETH on Solana.
One of the parameters that this function takes is a `transfer_message`, basically a message signed by the guardians that says which token to mint and how much.
This `transfer_message` comes from a Solana program, and is created by triggering a function called post_vaa, which checks if the message is valid by checking the signatures from the guardians.

> all guardians perform the same computation upon observing an on-chain event, and sign a so-called Validator Action Approval (VAA). If a 2/3+ majority of all guardian nodes have observed and signed the same event using their individual keys, then it is automatically considered valid by all Wormhole contracts on all chains and triggers a mint/burn. — Leo from Certus One

The `verify_signatures` function is called to get the signed signature_set for the function post_vaa.

Basically, the wormhole program obtains the set of signatures via the instruction *sysvar program* (whose address is input by the user).

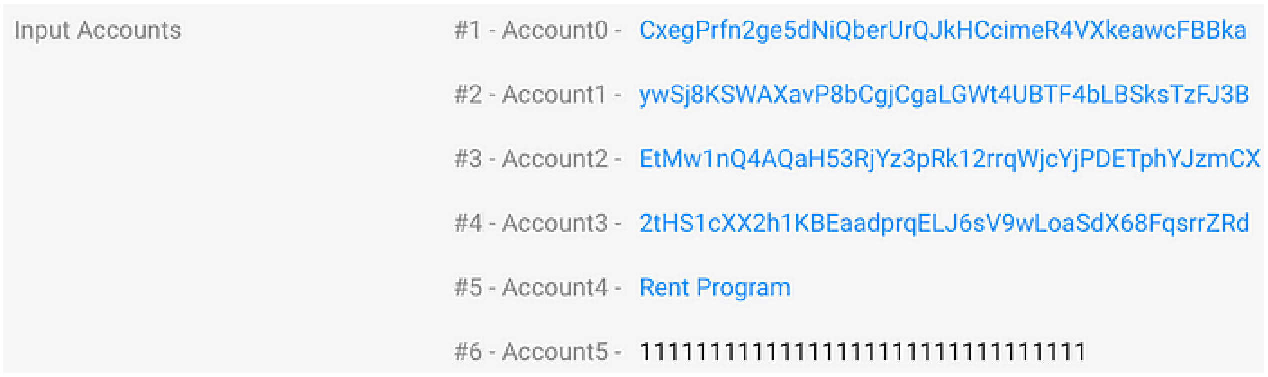
However, the `verify_signatures` function used the `load_instruction_at` function which outputs an instruction that is derived from the input data (which is the data of the instruction sysvar account).

This function does not check if the input sysvar program account is the real sysvar account.
Basically, the instruction sysvar program was never checked.

Thus, the attacker created a fake instruction sysvar account with fake data; therefore the signatures were spoofed with previously valid transferred tokens).

Thus, all signatures in the `signature_set` are marked as true falsely indicating that all the supposed guardian signatures were valid.

Here are the transactions parameters as supplied by the attacker



Input Accounts

#1 - Account0 - CxegPrfn2ge5dNiQberUrQJkHCcimeR4VXkeawcFBBka

#2 - Account1 - ywSj8KSWAXavP8bCgjCgaLGWt4UBTF4bLBSksTzFJ3B

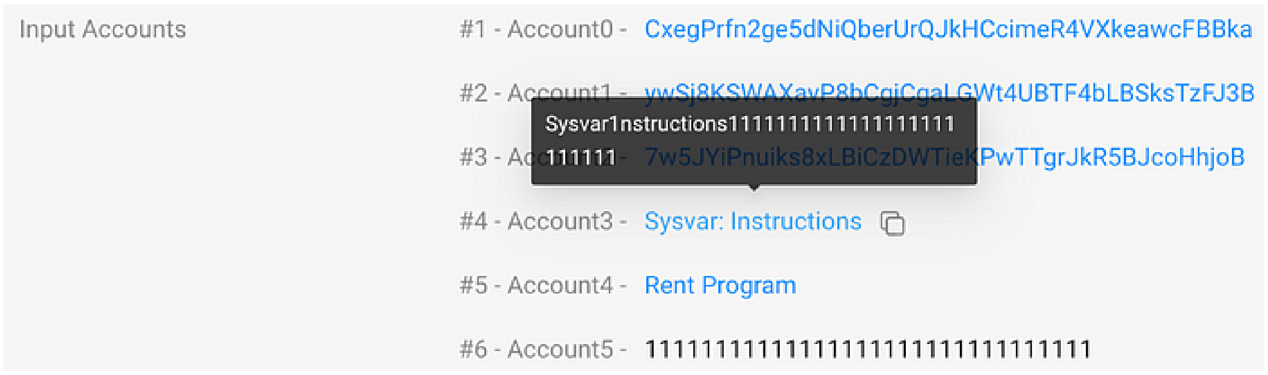
#3 - Account2 - EtMw1nQ4AQaH53RjYz3pRk12rrqWjcYjPDETphYJzmCX

#4 - Account3 - 2tHS1cXX2h1KBEaadprqELJ6sV9wLoaSdX68FqsrrZRd

#5 - Account4 - Rent Program

#6 - Account5 - 11111111111111111111111111111111

This is what they should have been



Input Accounts

#1 - Account0 - CxegPrfn2ge5dNiQberUrQJkHCcimeR4VXkeawcFBBka

#2 - Account1 - ywSj8KSWAXavP8bCgjCgaLGWt4UBTF4bLBSksTzFJ3B

Sysvar1nstructions11111111111111111
111111

#3 - Account2 - 7w5JYiPnuiks8xLBiCzDWTieKPwTTgrJkR5BJcoHhjoB

#4 - Account3 - Sysvar: Instructions

#5 - Account4 - Rent Program

#6 - Account5 - 11111111111111111111111111111111

Using an account created hours earlier with a single serialised instruction corresponding to the Sep256k1 contract, the attacker was able to fake the ‘SignatureSet’, and fraudulently mint 120k whETH on Solana using VAA verification that had been created in a previous transaction.

It is important to verify the validity of unmodified, reference-only accounts .
This is because a malicious user could create accounts with arbitrary data and then pass these accounts to the program in place of valid accounts in Solana .

It is interesting that a commit made on January 13th and pushed to Github on February 2nd replacing usage of `load_instruction_at` with `load_instruction_at_checked`, which actually confirms that the program being executed is the system program.

The exploit came a few hours *after* this commit as team didn’t have time to deploy the new version yet.

It is possible that an attacker was keeping an eye on the repository and looking out for suspicious commits.

Timeline

- **2021.10.20 06:01**: Solana commit to deprecate `load_instruction_at`.
- **2022.01.13 14:29**: Wormhole commit to update to Solana to 1.9.4.
- **2022.02.02 17:31**: Pull request of the Solana update commit.
- **2022.02.02 18:24**: Transaction to mint 120000 wormhole ETH on Solana.
- **2022.02.02 18:28**: Transaction to pull out 80000 wETH from wormhole smart contract.

> This is one of a number of attacks against blockchain bridges.
>
> ## Comments about the security of bridges
>
> Vitalik Buterin
>
> > it’s always safer to hold Ethereum-native assets on Ethereum or Solana-native assets on Solana than it is to hold Ethereum-native assets on Solana or Solana-native assets on Ethereum.
>
> Ronghui Gu, co-founder of CertiK, said in an interview:
>
> > Bridges are an attractive target for hackers: they hold millions of dollars of tokens in what is essentially an escrow contract, and by operating across multiple chains they multiply their potential points of failure.
>
> From Rekt:
>
> > It remains to be seen whether the future of DeFi will be cross-chain or ‘multi-chain’, but either way, the journey there will be long and dangerous. That being said, Solana is not yet the battle-hardened network that Ethereum has grown to be. Every exploit, for all the damage it does, offers a lesson for how to secure an evolving ecosystem.