



How to Earn Millions in Web3 Bug Bounties

Arbaz Hussain & Nemveer

Bsides Ahmedabad, October 12th 2024



Introduction

- Smart contract triagers at @immunefi
- Interested in investigating hacks on blockchain.
- Previously:
 - Nemveer : Independent web3 BH
 - Arbaz : web2 appsec engg
- Create educational content.



 ArbazKiraak



 nem_veer



@immunefi-team/bugfix-reviews-pocs

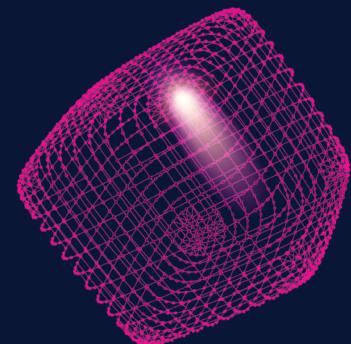


<https://immutenefi.medium.com>



Agenda

- Why are bug bounties in web3 so crucial?
- Key differences between Web2 and Web3
 - Architecture Overview
 - Bug bounties Comparison
- How the top Security Researchers use the Immunefi Platform to make millions
- Anatomy of high impact bug reports on Immunefi
- How to get started with bug bounties in web3
 - Roadmap to bug hunting in web3
 - Resources for you to get started





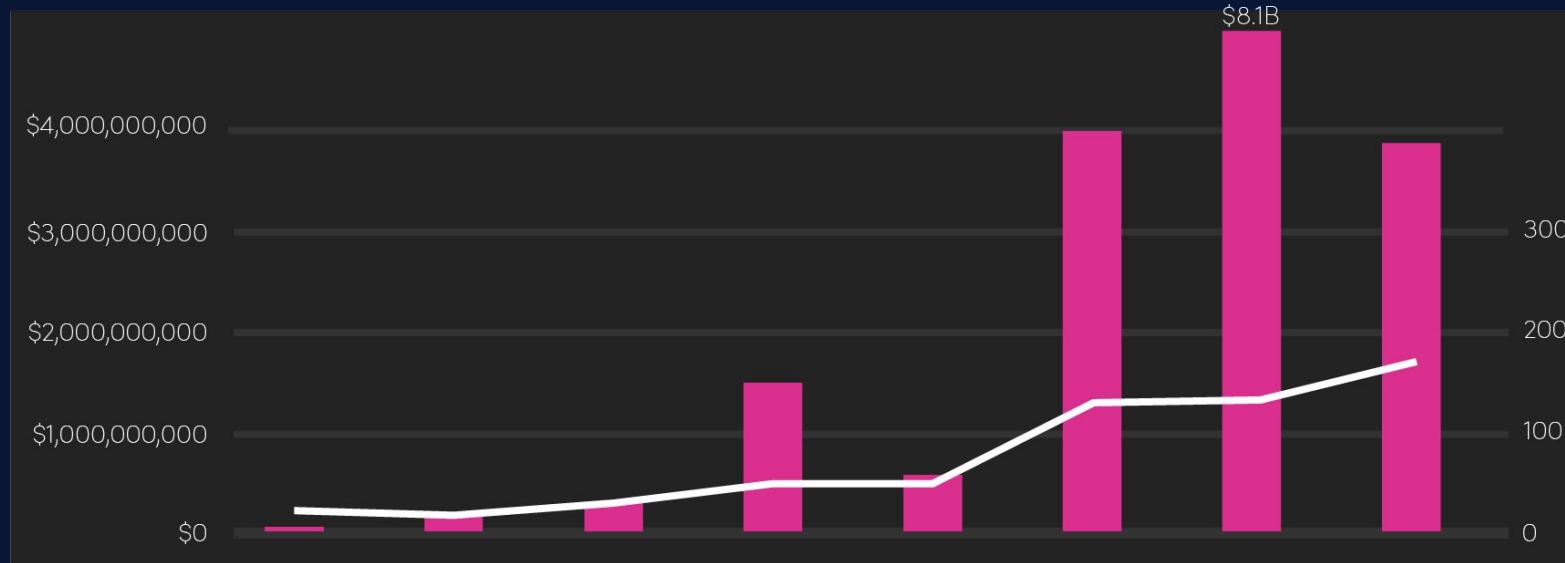
Why Bug Bounties are Crucial in Web3



Hacks are way more common in web3...

Total value hacked, annual

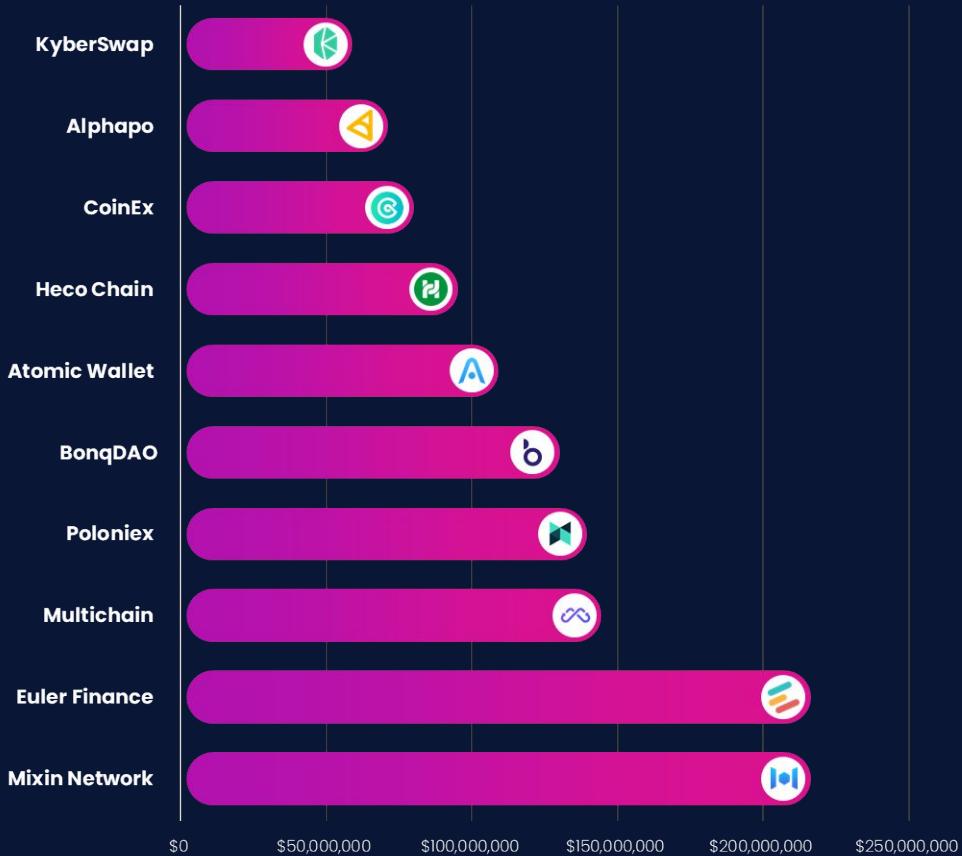
■ Total value hacked ■ Total number of hacks



Source: Immunefi Research



**\$1.8 Bn losses in 2023
in Blockchain**



Source: Immunefi Research



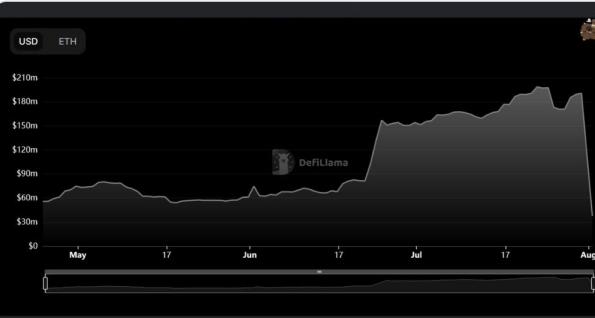
Some of the biggest hacks in web3

Nomad bridge hack ~ \$200M

 samczsun 
@samczsun · Follow

X

1/ Nomad just got drained for over \$150M in one of the most chaotic hacks that Web3 has ever seen. How exactly did this happen, and what was the root cause? Allow me to take you behind the scenes 



USD ETH

\$210m
\$180m
\$150m
\$120m
\$90m
\$60m
\$30m
\$0

May 17 Jun 17 Jul 17 Aug

5:15 AM · Aug 2, 2022

①

Heart 8.7K Reply Copy link

Read 368 replies

Technology

Crypto Bridge Nomad Drained of Nearly \$200M in Exploit

The exploit calls the security of cross-chain token bridges into question once again.

By Sam Kessler, Brandy Betz  Aug 2, 2022 at 9:05 a.m. Updated May 11, 2023 at 9:26 p.m.

- A cross-chain bridge enabling the transfer of tokens and data between blockchains
- A recent update to one of Nomad's smart contracts made it easy for users to spoof transactions.
- This meant users were able to withdraw money from the Nomad bridge that didn't actually belong to them.



Some of the biggest hacks in web3

Polynetwork hack ~ \$600M

Markets

Cross-Chain DeFi Site Poly Network Hacked; Hundreds of Millions Potentially Lost

DeFi platform Poly Network was attacked on Tuesday, with the alleged hacker draining roughly \$600 million in crypto.

By Eliza Gkritsi, Muyao Shen | Aug 10, 2021 at 7:26 p.m. Updated Sep 14, 2021 at 7:07 p.m.



Poly Network

@PolyNetwork2 · Follow



Important Notice:

We are sorry to announce that #PolyNetwork was attacked on @BinanceChain @ethereum and @0xPolygon Assets had been transferred to hacker's following addresses:

ETH:

0xC8a65Fadf0e0dDAf421F28FEAb69Bf6E2E589963

BSC:

0xD6e286A7cfD25E0c01fEe9756765D8033B32C71

6:08 PM · Aug 10, 2021



3.7K

Reply



Copy link to post

Read 755 replies

- Amongst the largest hacks in crypto history
- Hacker managed to steal \$611m taking \$273m in Ethereum, \$253m in Binance Smart Chain and \$85m in Polygon.
- Attacker created malicious parameters to bypass bridge's validation process and withdraw tokens from the bridge to their own address.



Some of the biggest hacks in web3

Eular hack ~ \$197M

② Gas Limit & Usage by Txn: 25,592 | 25,592 (100%)
② Gas Fees: Base: 20.057511121 Gwei | Max: 39.168636822 Gwei | Max Priority: 3 Gwei
② Burnt & Txn Savings Fees: 🔺 Burnt: 0.000513311824608632 ETH (\$0.89) 🔻 Txn Savings: 0.000412315928939992 ETH (\$0.71)

② Other Attributes: Txn Type: 2 (EIP-1559) | Nonce: 60 | Position In Block: 12

② Input Data:
Jacob here. I don't think what I say will help me in any way but I still want to say it. I fucked up. I didn't want to, but I messed with others' money, others' jobs, others' lives. I really fucked up. I'm sorry. I didn't mean all that. I really didn't fucking mean all that. Forgive me.

View Input As ▾

② Ether Price: \$1,773.53 / ETH
② Gas Limit & Usage by Txn: 23,816 | 23,816 (100%)
② Gas Fees: Base: 19.876972635 Gwei | Max: 40.719527422 Gwei | Max Priority: 3 Gwei
② Burnt & Txn Savings Fees: 🔺 Burnt: 0.00047338998027516 ETH (\$0.85) 🔻 Txn Savings: 0.000424938284807192 ETH (\$0.76)

② Other Attributes: Txn Type: 2 (EIP-1559) | Nonce: 59 | Position In Block: 18

② Input Data:
The rest of the money will be returned ASAP. I only look after my safety, and that is the reason for the delay. I'm sorry for any misunderstanding. Please read my next message.

View Input As ▾

- One of the most sophisticated exploit
- Attack was made possible by a vulnerability in the ETOKEN smart contract of Eular, lacking some critical validations.
- At the end, hacker returned stolen funds, marking one of the largest DeFi recoveries



- Not enough security awareness
- Continuous adoption
- Rapidly evolving landscape of Web3 leads to the swift emergence of new vulnerabilities and attack vectors
- Security is an always-on requirement post security audit

A Bug Bounty program is the last line of defence for projects.



Difference in Web2 & Web3 Bug Bounties

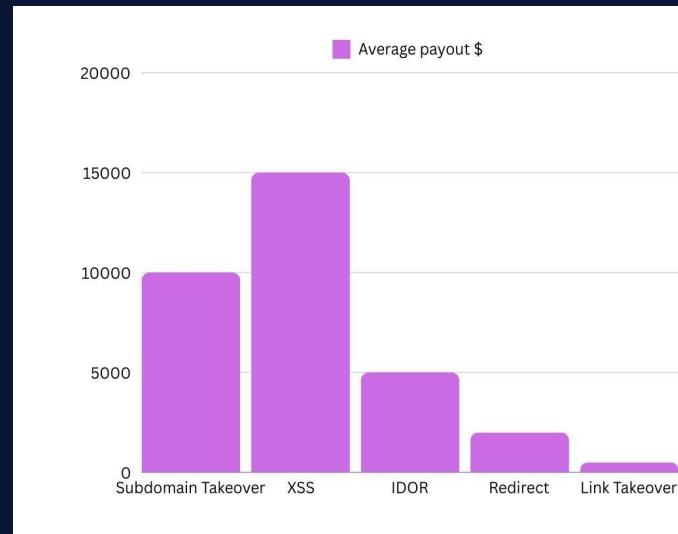
It's quite simple... Let us show you how



Top 5 Most reported Web2 Attack Vectors on Immunefi!

Average bounties for each attack vector

- Subdomain Takeovers : **\$10k**
- Cross Site Scripting (XSS) : **\$15k**
- Insecure Direct Object References (IDOR) : **\$5k**
- Open Redirect : **\$3k**
- Outgoing Broken Link Takeovers : **\$1k**



<https://immunefi.com/immunefi-vulnerability-severity-classification-system-v2-2/>

Aspect	WEB2	WEB3
What's at stake	Personal data (e.g PII)	Tokens or Assets
Impact severity	Higher severity with greater impact of user data	Immediate financial loss due to instant access to assets
Bounty Amount	Generally Lower	Higher, due to direct value at risk
Hacker Traceability	Easier to trace hackers in centralized systems	Challenging due to decentralised system
Vulnerability Emergence	Slower due to established security practices	Rapid leading to new attack vectors introducing vulnerabilities



- Stored Cross Site Scripting (xss)
- Substituting the contract addresses.
- Modifying transaction arguments or parameters.

Severity stands **critical** considering the digital assets at risk.

BadgerDAO Reveals Details of How It Was Hacked for \$120M

The DeFi platform said an application platform that runs on its cloud network was the vector for the attack.

By Nelson Wang ⌚ Dec 11, 2021 at 4:55 a.m. Updated Dec 12, 2021 at 10:09 p.m.

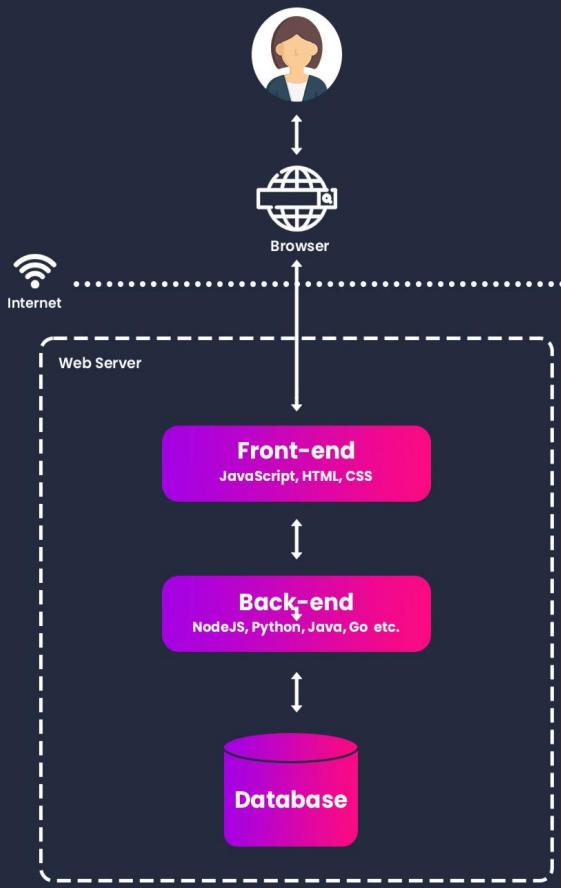
source: [coindesk](#)



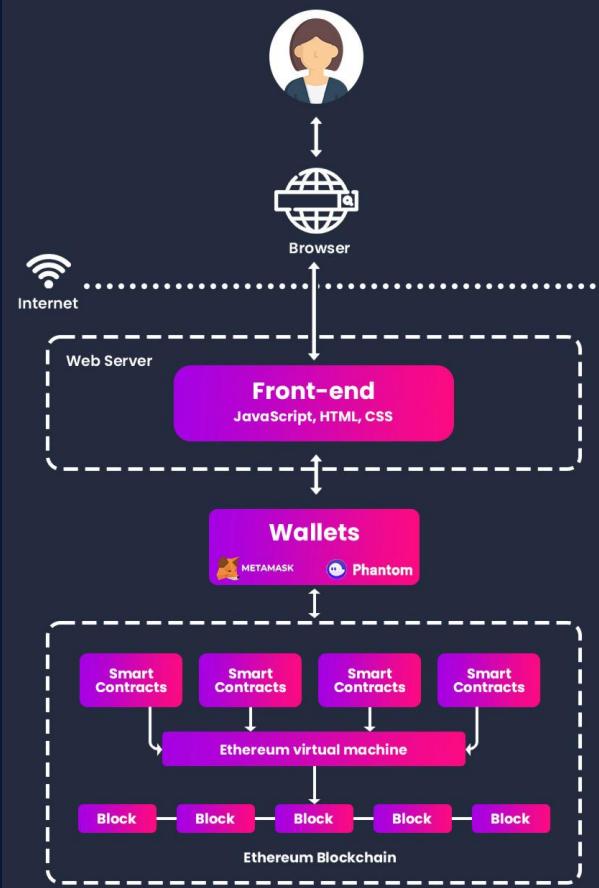
Web2 vs Web3 Architecture



WEB 2.0



WEB 3





Evolution of Blockchain



Early money as ledger



Proto-cuneiform tablet, Uruk, ca 3100–2900 B.C.

Caville 1911		Bettie 1911		Snow 1911		Page	
Feb 11	for gas	Feb 4	for fed	Feb 20	for gas	Feb 13	for gas
Mar 4	for fed	Mar 10	for fed	Feb 21	for gas	Feb 14	for gas
Apr 1	for Milk	May 5	for May	Feb 22	for fed	Feb 15	for gas
May 1	for May	May 1	for May	Mar 4	for fed	Feb 16	for gas
May 25	for May	May 5	for July	Feb 23	for fed	Feb 17	for gas
Jun 3	for June	Aug 5	for Aug	Mar 11	for fed	Feb 18	for gas
July 23	for July	Sept 5	for Sept	Mar 12	for fed	Feb 19	for gas
Aug 5	for July	Sept 5	for Sept	Mar 13	for fed	Feb 20	for gas
Aug 25	for Aug	Sept 5	for Sept	Mar 14	for fed	Feb 21	for gas
Sept 10	for Aug	Sept 5	for Sept	Mar 15	for fed	Feb 22	for gas
Oct 7	for Sept	Oct 5	for Oct	Mar 16	for fed	Feb 23	for gas
Oct 14	for Sept	Oct 5	for Oct	Mar 17	for fed	Feb 24	for gas
Nov 11	for Oct	Nov 5	for Nov	Mar 18	for fed	Feb 25	for gas
Dec 2	for Nov	Dec 5	for Dec	Mar 19	for fed	Feb 26	for gas
Dec 21	for Nov	Jan 5	for Jan	Mar 20	for fed	Feb 27	for gas
Dec 25	for Dec	Feb 5	for Feb	Mar 21	for fed	Feb 28	for gas
Jan 7	for Dec	Feb 20	for Feb	Mar 22	for fed	Mar 1	for Mar
Feb 11	for Jan	Feb 20	for Jan	Mar 23	for fed	Mar 2	for Mar
Mar 4	for Feb	Mar 5	for Mar	Mar 24	for fed	Mar 3	for Mar
Apr 1	for Mar	Apr 5	for April	Mar 25	for fed	Mar 4	for Mar
May 5	for May	May 5	for May	Mar 26	for fed	Mar 5	for May
June 1	for June	June 5	for June	Mar 27	for fed	Mar 6	for June
July 23	for July	July 5	for July	Mar 28	for fed	Mar 7	for July
Aug 5	for July	Aug 5	for Aug	Mar 29	for fed	Mar 8	for Aug
Sept 10	for Aug	Sept 5	for Sept	Mar 30	for fed	Mar 9	for Sept
Oct 7	for Sept	Oct 5	for Oct	Mar 31	for fed	Mar 10	for Oct
Oct 14	for Sept	Oct 5	for Oct	Apr 1	for fed	Mar 11	for Oct
Nov 11	for Oct	Nov 5	for Nov	Apr 2	for fed	Mar 12	for Nov
Dec 2	for Nov	Dec 5	for Dec	Apr 3	for fed	Mar 13	for Dec
Dec 21	for Nov	Jan 5	for Jan	Apr 4	for fed	Mar 14	for Jan
Dec 25	for Dec	Feb 5	for Feb	Apr 5	for fed	Mar 15	for Feb
Jan 7	for Dec	Feb 20	for Feb	Apr 6	for fed	Mar 16	for Feb
Feb 11	for Jan	Feb 20	for Jan	Apr 7	for fed	Mar 17	for Jan
Mar 4	for Feb	Mar 5	for Mar	Apr 8	for fed	Mar 18	for Mar
Apr 1	for Mar	Apr 5	for April	Apr 9	for fed	Mar 19	for April
May 5	for May	May 5	for May	Apr 10	for fed	Mar 20	for May
June 1	for June	June 5	for June	Apr 11	for fed	Mar 21	for June
July 23	for July	July 5	for July	Apr 12	for fed	Mar 22	for July
Aug 5	for July	Aug 5	for Aug	Apr 13	for fed	Mar 23	for Aug
Sept 10	for Aug	Sept 5	for Sept	Apr 14	for fed	Mar 24	for Sept
Oct 7	for Sept	Oct 5	for Oct	Apr 15	for fed	Mar 25	for Oct
Oct 14	for Sept	Oct 5	for Oct	Apr 16	for fed	Mar 26	for Oct
Nov 11	for Oct	Nov 5	for Nov	Apr 17	for fed	Mar 27	for Nov
Dec 2	for Nov	Dec 5	for Dec	Apr 18	for fed	Mar 28	for Dec
Dec 21	for Nov	Jan 5	for Jan	Apr 19	for fed	Mar 29	for Jan
Dec 25	for Dec	Feb 5	for Feb	Apr 20	for fed	Mar 30	for Feb
Jan 7	for Dec	Feb 20	for Feb	Apr 21	for fed	Mar 31	for Feb
Feb 11	for Jan	Feb 20	for Jan	Apr 22	for fed	Mar 31	for Jan
Mar 4	for Feb	Mar 5	for Mar	Apr 23	for fed	Apr 1	for Mar
Apr 1	for Mar	Apr 5	for April	Apr 24	for fed	Apr 2	for April
May 5	for May	May 5	for May	Apr 25	for fed	Apr 3	for May
June 1	for June	June 5	for June	Apr 26	for fed	Apr 4	for June
July 23	for July	July 5	for July	Apr 27	for fed	Apr 5	for July
Aug 5	for July	Aug 5	for Aug	Apr 28	for fed	Apr 6	for Aug
Sept 10	for Aug	Sept 5	for Sept	Apr 29	for fed	Apr 7	for Sept
Oct 7	for Sept	Oct 5	for Oct	Apr 30	for fed	Apr 8	for Oct
Oct 14	for Sept	Oct 5	for Oct	May 1	for fed	Apr 9	for Oct
Nov 11	for Oct	Nov 5	for Nov	May 2	for fed	Apr 10	for Nov
Dec 2	for Nov	Dec 5	for Dec	May 3	for fed	Apr 11	for Dec
Dec 21	for Nov	Jan 5	for Jan	May 4	for fed	Apr 12	for Jan
Dec 25	for Dec	Feb 5	for Feb	May 5	for fed	Apr 13	for Feb
Jan 7	for Dec	Feb 20	for Feb	May 6	for fed	Apr 14	for Feb
Feb 11	for Jan	Feb 20	for Jan	May 7	for fed	Apr 15	for Jan
Mar 4	for Feb	Mar 5	for Mar	May 8	for fed	Apr 16	for Mar
Apr 1	for Mar	Apr 5	for April	May 9	for fed	Apr 17	for April
May 5	for May	May 5	for May	May 10	for fed	Apr 18	for May
June 1	for June	June 5	for June	May 11	for fed	Apr 19	for June
July 23	for July	July 5	for July	May 12	for fed	Apr 20	for July
Aug 5	for July	Aug 5	for Aug	May 13	for fed	Apr 21	for Aug
Sept 10	for Aug	Sept 5	for Sept	May 14	for fed	Apr 22	for Sept
Oct 7	for Sept	Oct 5	for Oct	May 15	for fed	Apr 23	for Oct
Oct 14	for Sept	Oct 5	for Oct	May 16	for fed	Apr 24	for Oct
Nov 11	for Oct	Nov 5	for Nov	May 17	for fed	Apr 25	for Nov
Dec 2	for Nov	Dec 5	for Dec	May 18	for fed	Apr 26	for Dec
Dec 21	for Nov	Jan 5	for Jan	May 19	for fed	Apr 27	for Jan
Dec 25	for Dec	Feb 5	for Feb	May 20	for fed	Apr 28	for Feb
Jan 7	for Dec	Feb 20	for Feb	May 21	for fed	Apr 29	for Feb
Feb 11	for Jan	Feb 20	for Jan	May 22	for fed	Apr 30	for Jan
Mar 4	for Feb	Mar 5	for Mar	May 23	for fed	May 1	for Mar
Apr 1	for Mar	Apr 5	for April	May 24	for fed	May 2	for April
May 5	for May	May 5	for May	May 25	for fed	May 3	for May
June 1	for June	June 5	for June	May 26	for fed	May 4	for June
July 23	for July	July 5	for July	May 27	for fed	May 5	for July
Aug 5	for July	Aug 5	for Aug	May 28	for fed	May 6	for Aug
Sept 10	for Aug	Sept 5	for Sept	May 29	for fed	May 7	for Sept
Oct 7	for Sept	Oct 5	for Oct	May 30	for fed	May 8	for Oct
Oct 14	for Sept	Oct 5	for Oct	May 31	for fed	May 9	for Oct
Nov 11	for Oct	Nov 5	for Nov	June 1	for fed	May 10	for Nov
Dec 2	for Nov	Dec 5	for Dec	June 2	for fed	May 11	for Dec
Dec 21	for Nov	Jan 5	for Jan	June 3	for fed	May 12	for Jan
Dec 25	for Dec	Feb 5	for Feb	June 4	for fed	May 13	for Feb
Jan 7	for Dec	Feb 20	for Feb	June 5	for fed	May 14	for Feb
Feb 11	for Jan	Feb 20	for Jan	June 6	for fed	May 15	for Jan
Mar 4	for Feb	Mar 5	for Mar	June 7	for fed	May 16	for Mar
Apr 1	for Mar	Apr 5	for April	June 8	for fed	May 17	for April
May 5	for May	May 5	for May	June 9	for fed	May 18	for May
June 1	for June	June 5	for June	June 10	for fed	May 19	for June
July 23	for July	July 5	for July	June 11	for fed	May 20	for July
Aug 5	for July	Aug 5	for Aug	June 12	for fed	May 21	for Aug
Sept 10	for Aug	Sept 5	for Sept	June 13	for fed	May 22	for Sept
Oct 7	for Sept	Oct 5	for Oct	June 14	for fed	May 23	for Oct
Oct 14	for Sept	Oct 5	for Oct	June 15	for fed	May 24	for Oct
Nov 11	for Oct	Nov 5	for Nov	June 16	for fed	May 25	for Nov
Dec 2	for Nov	Dec 5	for Dec	June 17	for fed	May 26	for Dec
Dec 21	for Nov	Jan 5	for Jan	June 18	for fed	May 27	for Jan
Dec 25	for Dec	Feb 5	for Feb	June 19	for fed	May 28	for Feb
Jan 7	for Dec	Feb 20	for Feb	June 20	for fed	May 29	for Feb
Feb 11	for Jan	Feb 20	for Jan	June 21	for fed	May 30	for Jan
Mar 4	for Feb	Mar 5	for Mar	June 22	for fed	May 31	for Mar
Apr 1	for Mar	Apr 5	for April	June 23	for fed	May 32	for April
May 5	for May	May 5	for May	June 24	for fed	May 33	for May
June 1	for June	June 5	for June	June 25	for fed	May 34	for June
July 23	for July	July 5	for July	June 26	for fed	May 35	for July
Aug 5	for July	Aug 5	for Aug	June 27	for fed	May 36	for Aug
Sept 10	for Aug	Sept 5	for Sept	June 28	for fed	May 37	for Sept
Oct 7	for Sept	Oct 5	for Oct	June 29	for fed	May 38	for Oct
Oct 14	for Sept	Oct 5	for Oct	June 30	for fed	May 39	for Oct
Nov 11	for Oct	Nov 5	for Nov	July 1	for fed	May 40	for Nov
Dec 2	for Nov	Dec 5	for Dec	July 2	for fed	May 41	for Dec
Dec 21	for Nov	Jan 5	for Jan	July 3	for fed	May 42	for Jan
Dec 25	for Dec	Feb 5	for Feb	July 4	for fed	May 43	for Feb
Jan 7	for Dec	Feb 20	for Feb	July 5	for fed	May 44	for Feb
Feb 11	for Jan	Feb 20	for Jan	July 6	for fed	May 45	for Jan
Mar 4	for Feb	Mar 5	for Mar	July 7	for fed	May 46	for Mar
Apr 1	for Mar	Apr 5	for April	July 8	for fed	May 47	for April
May 5	for May	May 5	for May	July 9	for fed	May 48	for May
June 1	for June	June 5	for June	July 10	for fed	May 49	for June
July 23	for July	July 5	for July	July 11	for fed	May 50	for July
Aug 5	for July	Aug 5	for Aug	July 12	for fed	May 51	for Aug
Sept 10	for Aug	Sept 5	for Sept	July 13	for fed	May 52	for Sept
Oct 7	for Sept	Oct 5	for Oct	July 14	for fed	May 53	for Oct
Oct 14	for Sept	Oct 5	for Oct	July 15	for fed	May 54	for Oct
Nov 11	for Oct	Nov 5	for Nov	July 16	for fed	May 55	for Nov
Dec 2	for Nov	Dec 5	for Dec	July 17	for fed	May 56	for Dec
Dec 21	for Nov	Jan 5	for Jan	July 18	for fed	May 57	for Jan
Dec 25	for Dec	Feb 5	for Feb	July 19	for fed	May 58	for Feb
Jan 7	for Dec	Feb 20	for Feb	July 20	for fed	May 59	for Feb
Feb 11	for Jan	Feb 20	for Jan	July 21	for fed	May 60	for Jan
Mar 4	for Feb	Mar 5	for Mar	July 22	for fed	May 61	for Mar
Apr 1	for Mar	Apr 5	for April	July 23	for fed	May 62	for April
May 5	for May	May 5	for May	July 24	for fed	May 63	for May
June 1	for June	June 5	for June	July 25	for fed	May 64	for June
July 23	for July	July 5	for July	July 26	for fed	May 65	for July
Aug 5	for July	Aug 5	for Aug	July 27	for fed	May 66	for Aug
Sept 10	for Aug	Sept 5	for Sept	July 28	for fed	May 67	for Sept
Oct 7	for Sept	Oct 5	for Oct	July 29	for fed	May 68	for Oct
Oct 14	for Sept	Oct 5	for Oct	July 30	for fed	May 69	for Oct
Nov 11	for Oct	Nov 5	for Nov	July 31	for fed	May 70	for Nov
Dec 2	for Nov	Dec 5	for Dec	Aug 1	for fed	May 71	for Dec
Dec 21	for Nov	Jan 5	for Jan	Aug 2	for fed	May 72	for Jan
Dec 25	for Dec	Feb 5	for Feb	Aug 3	for fed	May 73	for Feb
Jan 7	for Dec	Feb 20	for Feb	Aug 4	for fed	May 74	for Feb
Feb 11	for Jan	Feb 20	for Jan	Aug 5	for fed	May 75	for Jan
Mar 4	for Feb	Mar 5	for Mar	Aug 6	for fed	May 76	for Mar
Apr 1	for Mar	Apr 5	for April	Aug 7	for fed	May 77	for April
May 5	for May	May 5	for May	Aug 8	for fed	May 78	for May
June 1	for June	June 5	for June	Aug 9	for fed	May 79	for June
July 23	for July	July 5	for July	Aug 10	for fed	May 80	for July
Aug 5	for July	Aug 5	for Aug	Aug 11	for fed	May 81	for Aug
Sept 10	for Aug	Sept 5	for Sept	Aug 12	for fed	May 82	for Sept
Oct 7	for Sept	Oct 5	for Oct	Aug 13	for fed	May 83	for Oct
Oct 14	for Sept	Oct 5	for Oct	Aug 14	for fed	May 84	for Oct
Nov 11	for Oct	Nov 5	for Nov	Aug 15	for fed	May 85	for Nov
Dec 2	for Nov	Dec 5	for Dec	Aug 16	for fed	May 86	for Dec
Dec 21	for Nov	Jan 5	for Jan	Aug 17	for fed	May 87	for Jan
Dec 25	for Dec	Feb 5	for Feb	Aug 18	for fed	May 88	for Feb
Jan 7	for Dec	Feb 20	for Feb	Aug 19	for fed	May 89	for Feb
Feb 11	for Jan	Feb 20	for Jan	Aug 20	for fed	May 90	for Jan
Mar 4	for Feb	Mar 5	for Mar	Aug 21	for fed	May 91	for Mar
Apr 1	for Mar	Apr 5	for April	Aug 22	for fed	May 92	for April
May 5	for May	May 5	for May	Aug 23	for fed	May 93	for May
June 1	for June	June 5	for June	Aug 24	for fed	May 94	for June
July 23	for July	July 5	for July	Aug 25	for fed	May 95	for July
Aug 5	for July	Aug 5	for Aug	Aug 26	for fed	May 96	for Aug
Sept 10	for Aug	Sept 5	for Sept	Aug 27	for fed	May 97	for Sept
Oct 7	for Sept	Oct 5	for Oct	Aug 28	for fed	May 98	for Oct
Oct 14	for Sept	Oct 5	for Oct	Aug 29	for fed	May 99	for Oct
Nov 11	for Oct	Nov 5	for Nov	Aug 30	for fed	May 100	for Nov
Dec 2	for Nov	Dec 5	for Dec	Aug 31	for fed	May 101	for Dec
Dec 21	for Nov	Jan 5	for Jan	Sept 1	for fed	May 102	for Jan
Dec 25	for Dec	Feb 5	for Feb	Sept 2	for fed	May 103	for Feb

Old ledger from 1911 stock photo



Alice pays Bob \$20
Bob pays Casey \$40
Casey pays Daniel \$30
Daniel pays Alice \$10

Alice
Bob
Casey
Daniel

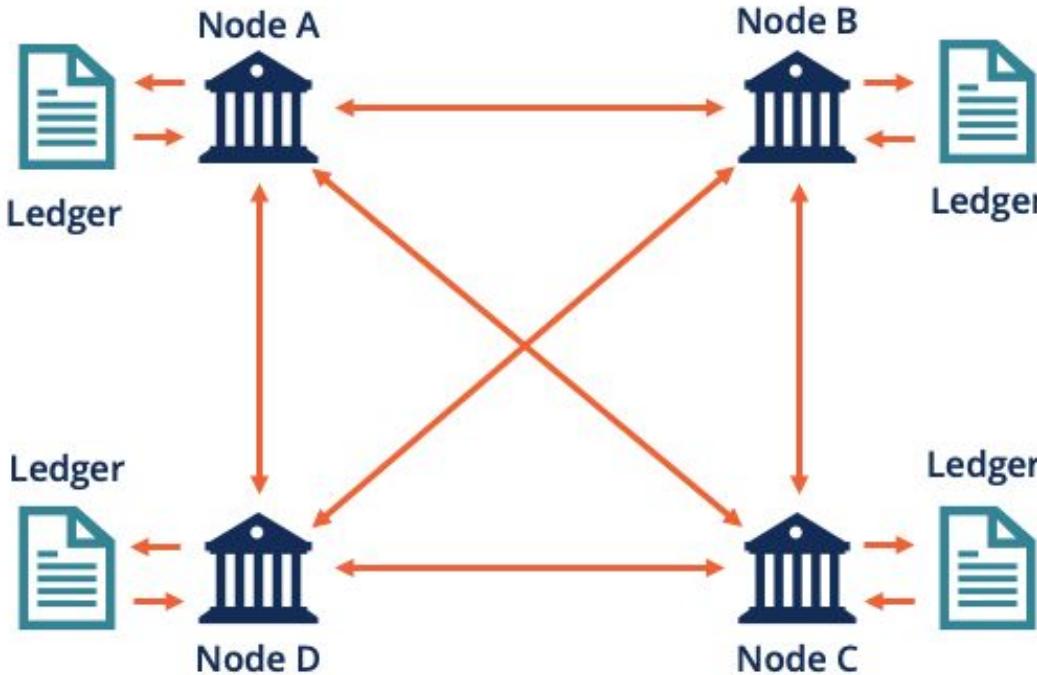
Common Ledger



Everyone has a copy of the same ledger



Distributed Ledgers

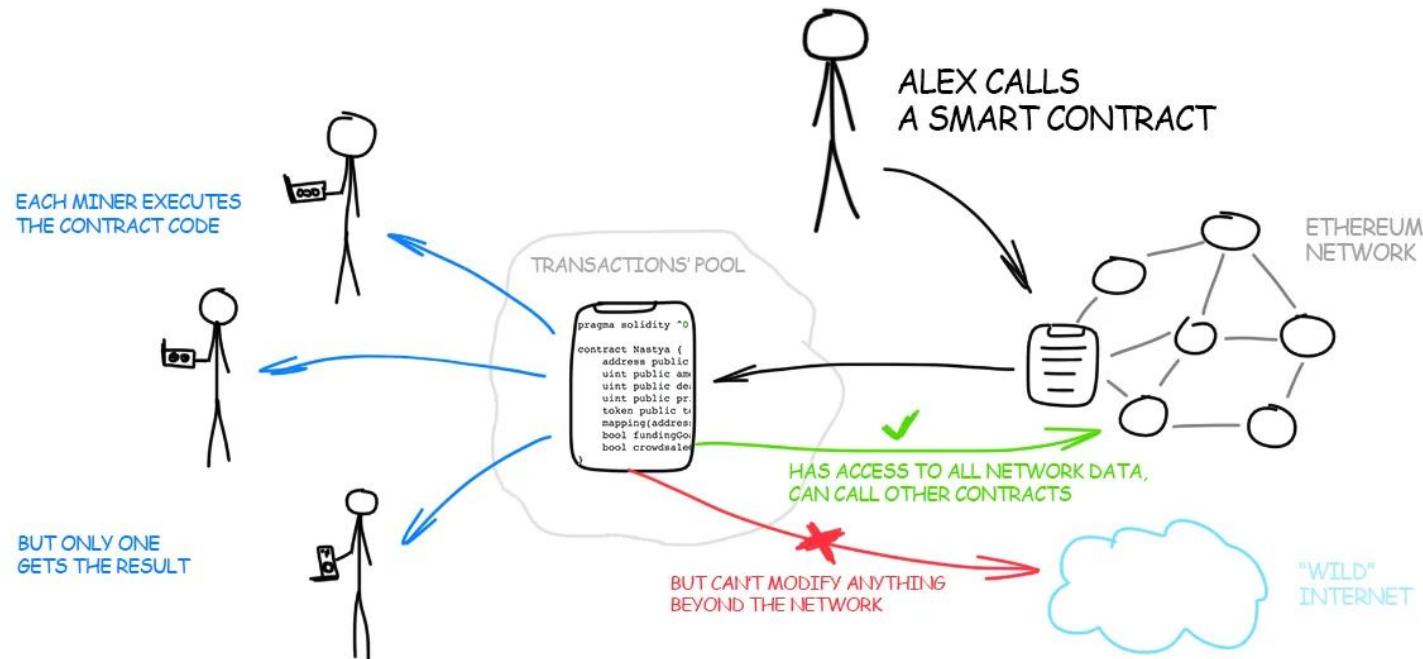


Problems are soluble

- Proof of work
- Cryptographic Hash Functions & Digital Signatures
- Merkle trees
- Blockchain
- Native currency



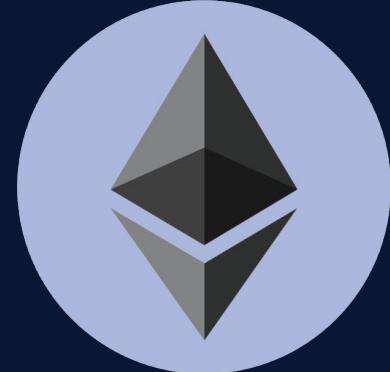
Ethereum - programmable currency





Web3

- Multi-sigs +\$100B
- Autonomous Market Makers
- Decentralized borrowing and lending,
flash loans
- **Decentralized Finance**





Languages used for smart contracts

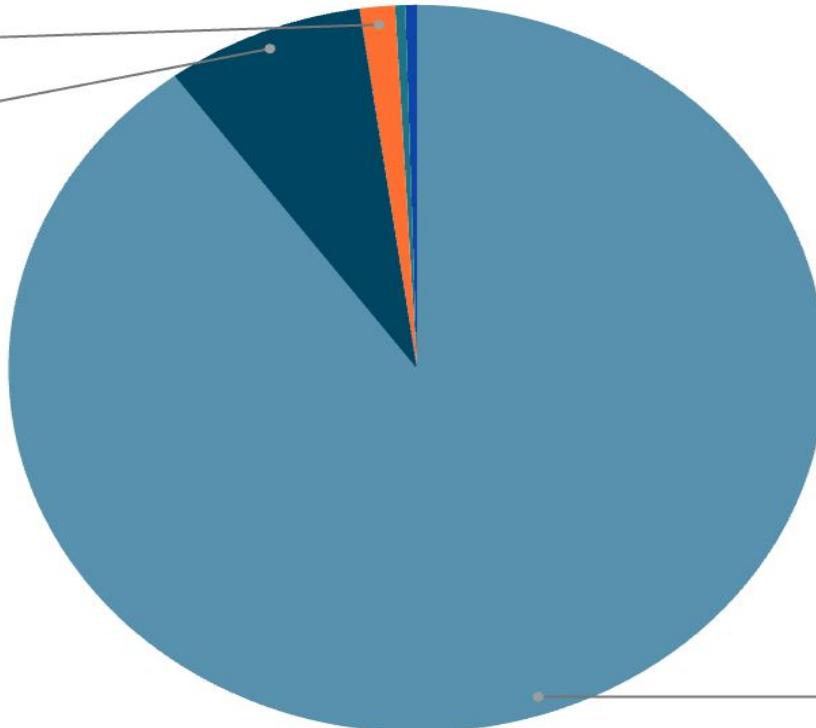
Languages by TVL dominance

Vyper

1.4%

Rust

7.9%



Solidity 89.89 %

Rust 7.86 %

Vyper 1.39 %

Bitcoin Script 0.4 %

Haskell 0.17 %

Cairo 0.12 %

C++ 0.06 %

Clarity 0.03 %

C# 0.02 %

Java 0.02 %

Others 0.03 %

Solidity
89.9%



Differences in Web2 & Web3 Languages

Language	Blockchain	Similar to...
Solidity	Arbitrum, BNB, Ethereum, Polygon, Optimism	Javascript
Rust	Solana, Polkadot, Cosmos	C, C++
Vyper	Same as Solidity	Python

```
contract SSimpleStorage {
    uint256 storedNumber;

    function storeNumber(uint256 newNumber) external {
        storedNumber = newNumber;
    }

    function readNumber() external view returns (uint256) {
        return storedNumber;
    }
}
```

Solidity

```
struct NumberStorage(Option<u32>);

impl NumberStorage {
    fn new() -> Self { Self(None) }
    fn store_number(&mut self, value: u32) { self.0 = Some(value); }
    fn read_number(&self) -> Option<u32> { self.0 }
}

fn main() {
    let mut storage = NumberStorage::new();
    storage.store_number(42);
    println!("Stored number: {:?}", storage.read_number());
}
```

Rust

```
storedNumer: uint256

@external
def storeNumber(newNumber: uint256):
    self.storedNumer = newNumber

@external
@view
def readNumber() -> uint256:
    return self.storedNumer
```

Vyper



How Immunefi bug bounty platform works?



Bug bounties and Immunefi

- **Code is Law, Code is Money**
 - In DeFi and blockchain, exploited bugs translate directly into financial loss.
 - Bug bounties serve as a frontline defense in web3.
- **Immunefi: Leading the Charge**
 - Hosts the largest bug bounty program in DeFi and blockchain.
 - Attracts top security talent with substantial rewards.
 - Aligns incentives to convert potential blackhats into whitehats.



Leading Blockchain security with largest and most proficient community of **elite** security researchers safeguarding over **300+ leading Blockchain projects**

45k

Largest Blockchain Security Community

1000+

SRs reported bugs to avert mainnet theft of funds

\$100M

Bounties Paid

+52%

Report Hit Rate* on Impact Focused Bugs

\$25B

Loss Prevented

\$60B

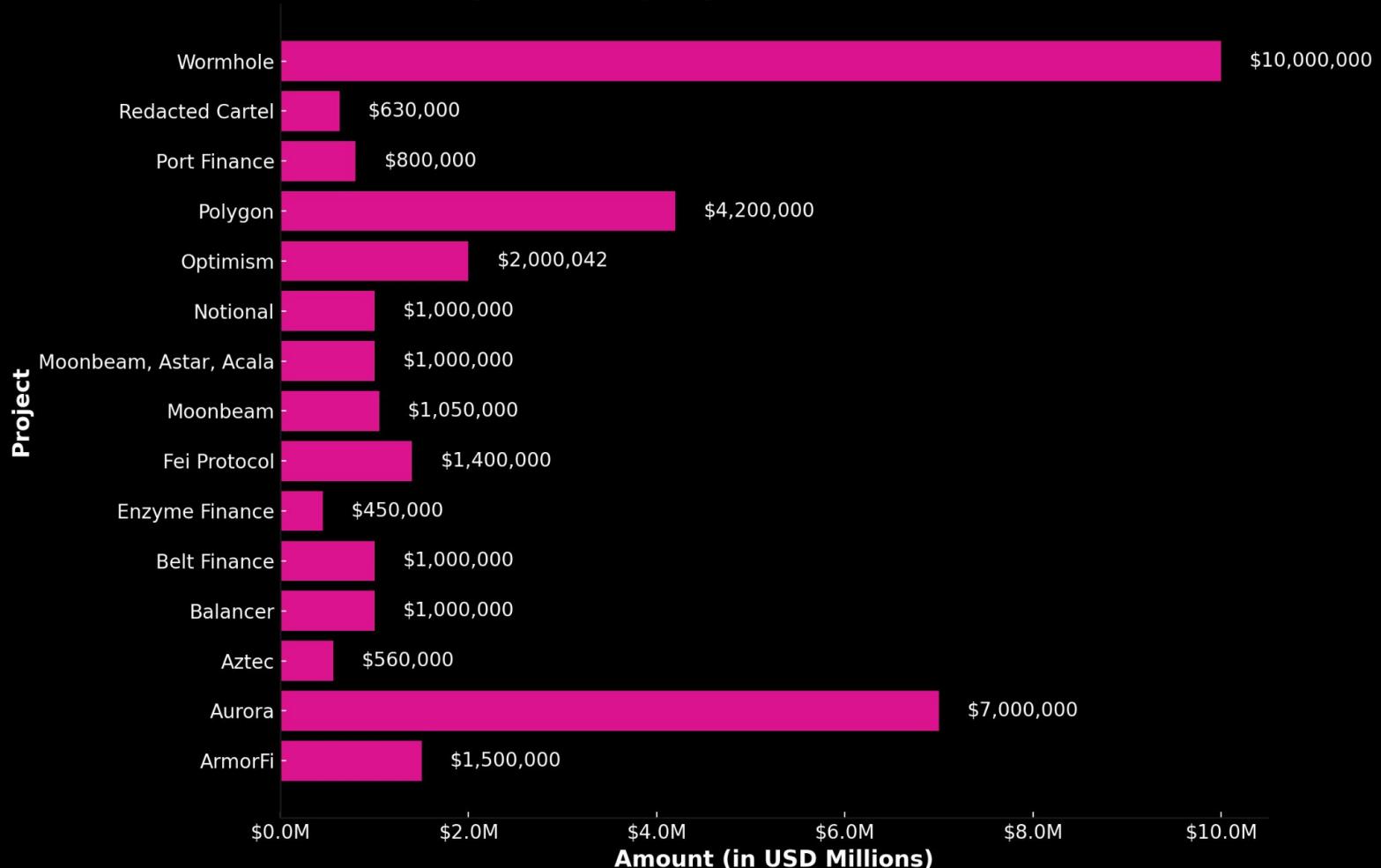
Users Funds Protected

TRUSTED BY





Top Paid \$ Bug Reports on Immunefi Platform





How bug hunting on Immunifi Works?

Our triage team evaluates the report based on factors like severity of the vulnerability, the potential impact on the project, and the likelihood of exploitation.

Project confirms the valid reports and completes pay out to the SR

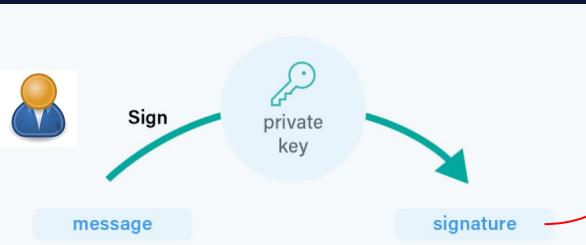




Review of the Top Paid Bug reports on Immunefi



Polygon - \$2M Bounty



Ether on Ethereum = Matic on Polygon

ecrecover = A built-in function which returns address of signer from hashed data and signature

```
function transferWithSig(
    bytes calldata sig,
    uint256 amount,
    bytes32 data,
    uint256 expiration,
    address to
) external returns (address from) {
    require(amount > 0);
    require(
        expiration == 0 || block.number <= expiration,
        "Signature is expired"
    );

    bytes32 dataHash = hashEIP712MessageWithAddress(
        hashTokenTransferOrder(msg.sender, amount, data, expiration),
        address(this)
    );

    require(disabledHashes[dataHash] == false, "Sig deactivated");
    disabledHashes[dataHash] = true;

    from = ecrecovery(dataHash, sig);
    _transferFrom(from, address(uint160(to)), amount);
}
```



Polygon - \$2M Bounty

```
function ecrecovery(bytes32 hash, bytes memory sig)
    public
    pure
    returns (address result)
{
    bytes32 r;
    bytes32 s;
    uint8 v;
    if (sig.length != 65) {
        return address(0x0);
    }
```

If bytes in signatures are less than 65, it will return address(0) as the signer

```
function _transfer(address sender, address recipient, uint256 amount)
    internal
{
    require(recipient != address(this), "can't send to MRC20");
    address(uint160(recipient)).transfer(amount);
    emit Transfer(sender, recipient, amount);
}
```

It doesn't check that the sender has enough balance to transfer.

It allows an exploit where anybody can craft such a signature with less than 65 bytes and mint an arbitrary number of tokens



Wormhole - \$10M Bounty



Normal Workflow.



1. Malicious user deploys Evil contract containing SELFDESTRUCT opcode.

2. Delegate(call) to Evil contract.



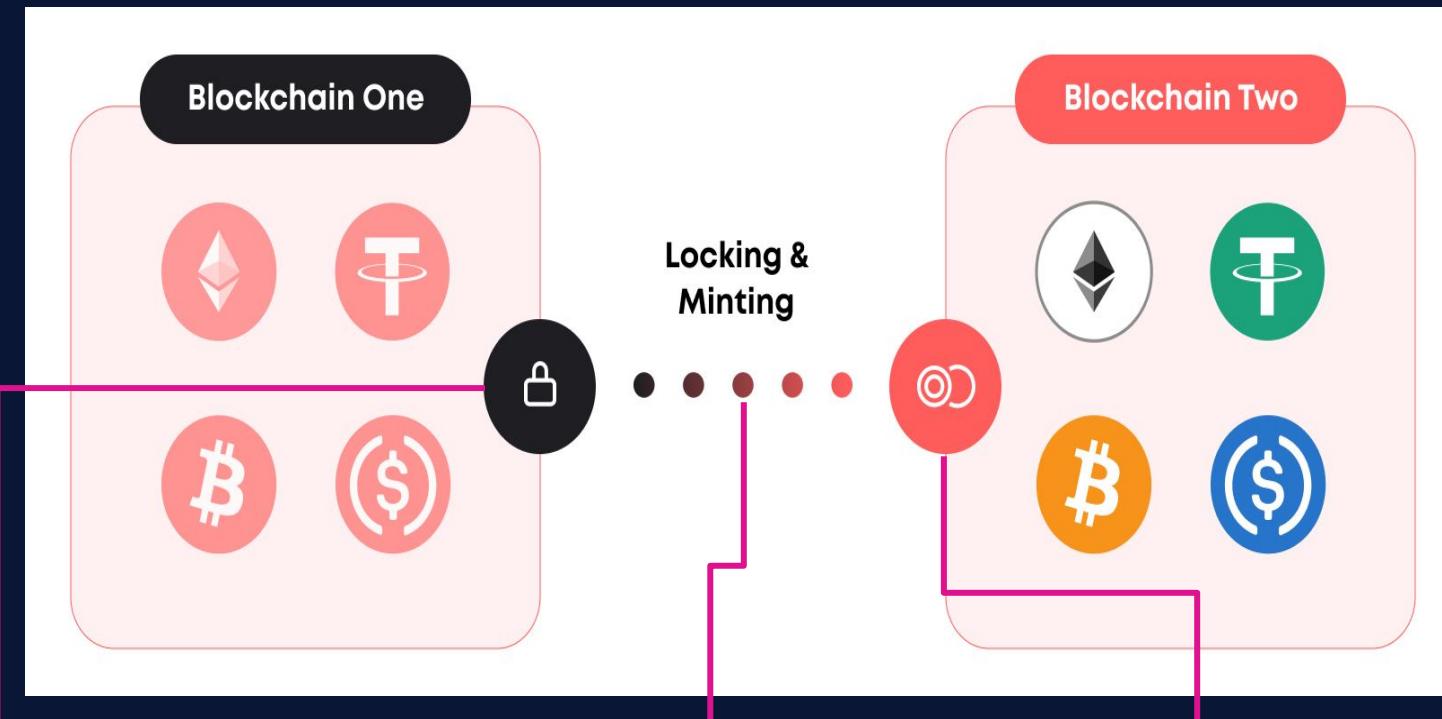
Its storage and code are erased from the blockchain **making funds locked**.

Proxy contract is bricked.



Aurora Finance : \$6M Bounty

Bridges in blockchains to communicate with each other



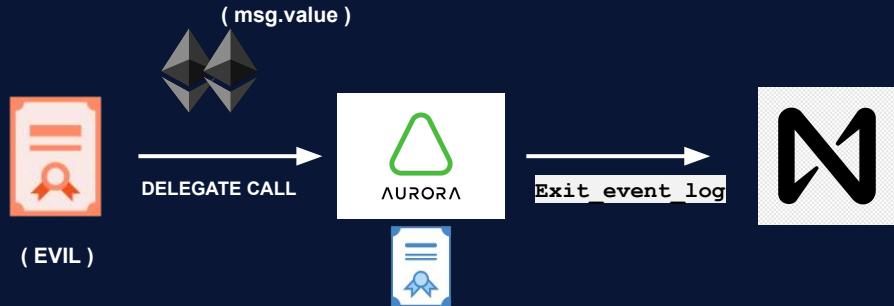
Sends Tokens to special contract to lock

Emits deposit event

Release on others



Aurora Finance : \$6M Bounty



1. Attacker creates a malicious contract.
2. It triggers a `delegateCall` to the bridge contract, simulating an ETH transfer.
3. No actual ETH is sent.
4. Bridge contract runs as if ETH was received (via `msg.value`).
5. Bridge emits an event, and the attacker receives ETH on the NEAR blockchain.

```
impl ExitToNear {  
    // Exit to NEAR precompile address  
    pub const ADDRESS: Address =  
        super::make_address(0xe9217bc7,  
        0x0b7ed1f598ddd3199e80b093fa71124f);  
  
    pub fn new(current_account_id: AccountId) -> Self {  
        Self { current_account_id }  
    }  
  
    impl Precompile for ExitToNear {  
        .....  
        let exit_event_log = Log {  
            address: Self::ADDRESS.raw(),  
            topics: exit_event_log.topics,  
            data: exit_event_log.data,  
        };  
        Ok(PrecompileOutput {  
            logs: vec![promise_log, exit_event_log],  
            ..Default::default()  
        }  
        .into())  
    }  
}
```



Who wants to be a Web3 Hacker?



Roadmap to be a Web3 Hacker?

Learn the architecture of different blockchains and languages specific to that blockchain for eg. Solidity for EVM based chains

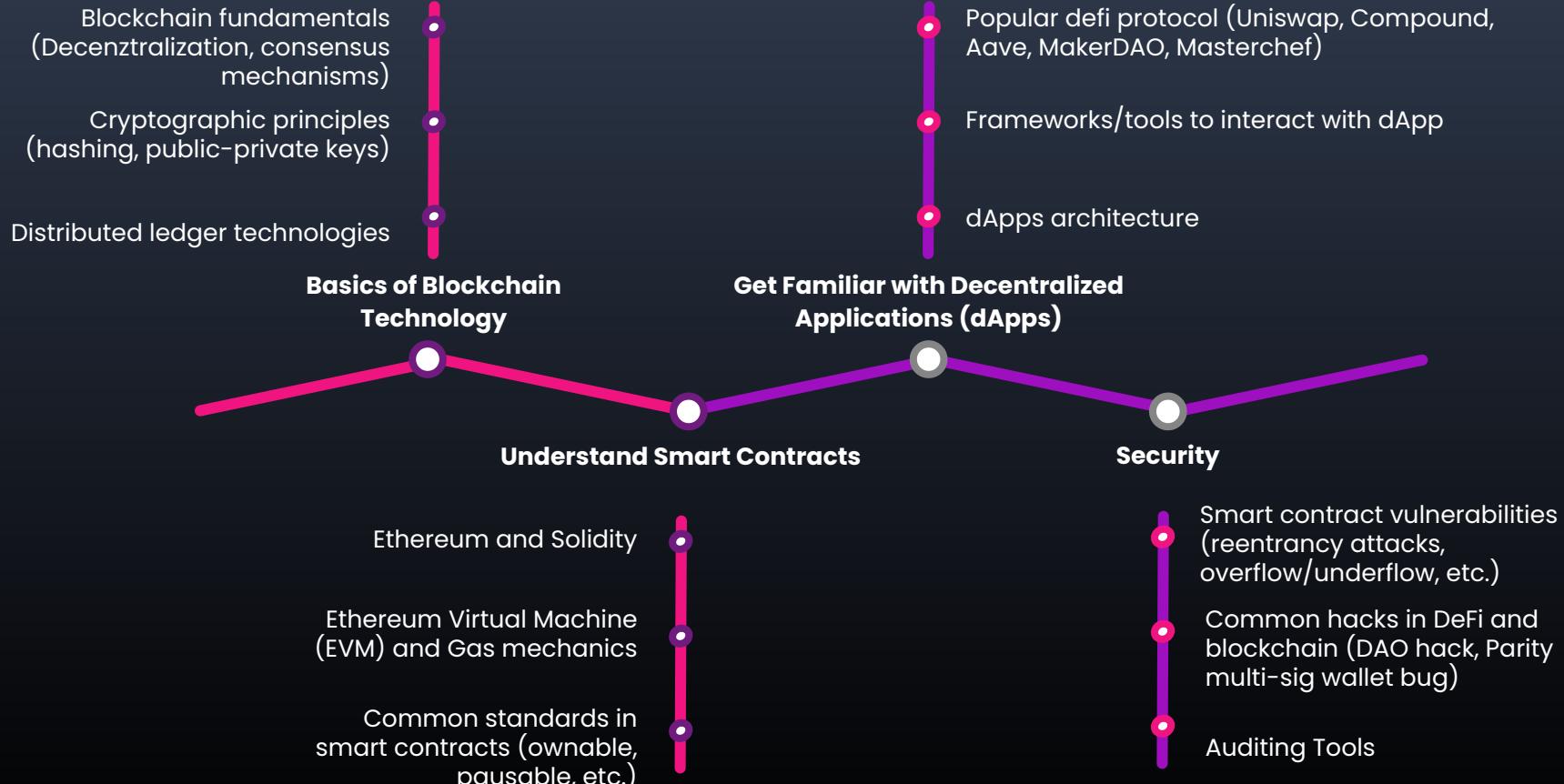


Understand the fundamentals
of blockchain

Familiarize with common security pitfalls
and security best practices



Roadmap to be a Web3 Hacker?





Useful links to get you started

- <https://medium.com/immunefi/hacking-the-blockchain-an-ultimate-guide-4f34b33c6e8b>
- <https://www.youtube.com/@PatrickAlphaC/playlists>
- <https://solidity-by-example.org>
- <https://github.com/ethereumbook/ethereumbook>
- <https://github.com/OffcierCia/DeFi-Developer-Road-Map>
- <https://www.damnvulnerabledefi.xyz>
- <https://cmichel.io/how-to-become-a-smart-contract-auditor/>
- <https://ethernaut.openzeppelin.com/>
- <https://github.com/immunefi-team/community-challenges>
- https://github.com/x676f64/secureum-mind_map
- <https://cryptozombies.io/>



The Only Web3 CrowdSec Platform to Pay
\$100 Million+
to Security Researchers

Scan the QR →

To sign up and we will send you resources to get you started with web3 bug hunting



SIGN UP

Win Cool Immunefi Merch

Visit our Booth

We have some exciting contests lined up for you

