A Security Survey for Ethereum Based DApps Available for Different Domains and Across the Application Layer Stack

Mukesh Yadav – 19069792

Abstract

DApp is an application build over a decentralised peer to peer network similar to a blockchain platform such as Ethereum. It carries all the benefits which are intrinsic to the Blockchain e.g. decentralised data and processing, peer-to-peer network, immutability, open-source, security and transparency. It is used for development of applications for various domains e.g. financial, game, social and media. It has gained a reputation for its reliability and stability because of its architecture. In this paper, we have surveyed its architecture and security vulnerabilities for released DApps across the application stack including web and smart contracts for several domains.

1. Introduction

Bitcoin (Nakamoto) was the first cryptocurrency which was developed in 2009, since then it has become the synonym to cryptocurrency. It had been recognised as the top rated currency in 2015 and has recorded more than 350,000 confirmed transactions ("N-Transactions,"). Bitcoin is based on a technology called blockchain, it a distributed ledger of blocks that allows secured transactions in an unsecured environment. Many other platforms have been developed based on blockchain, Ethereum (Buterin) is one of them. Ethereum has a feature called 'smart contract' which is a piece of code saved as an immutable transaction on blockchain, it runs on Ethereum Virtual Machine(EVM) after getting converted into bytecode by itself without intervention of any central authority and exchange data among several smart contracts deployed on blockchain. EVM is a Turing complete machine therefore, it is capable to program similar to any of the high-level modern programming languages. With smart contracts, blockchain has revolutionised the development of DApps and we have witnessed many applications for several fields, including finance, healthcare, logistic and entertainment (Al-Jaroodi & Mohamed, 2019).

Ethereum has become one of the dominant platforms for development of a DApp; it has approximately 700,000 transactions on daily basis in Jan, 2020 (etherscan.io, n.d.). On the other hand, approximately 5,650 smart contracts have been created till May, 2020 ("State of the DApps — DApp Statistics,"). However, a severe concern for DApps is its security

especially domains related to finance, game, media, social, healthcare and IoT. There have been several incidents reported on stolen cryptocurrencies ("Overview · Smart Contract Weakness Classification and Test Cases,"), approximately $158 million stolen through seven cryptocurrency heists in 2019 ("Hackers Steal $49 Million in Ethereum From Upbit Exchange,"). A DeFi app has been hacked and $300,000 stolen in April 2020, an attacker stole Ethereum and equivalent token of worth $300,000 by accessing market of the app ("A DeFi App Was Just Hacked for Over $300,000 in Ethereum & Bitcoin," 2020). These attacks mostly occurred due to a lack of attention in designing the client wallet app, exchange system, bugs present in smart contracts, defects present in platform, understanding and limited knowledge of a developer. In this paper I have explored two DApps from each domain(finance, game, media and social) and studied them. I have performed an audit on the web and smart contact layer for each DApp.

In this paper, we have analysed DApps based on Ethereum based blockchain technology, all of them are popular public DApp in their respective domains and performed a security study and audit for each of the DApp. The rest of this paper has been arranged as follows: we introduced the overview of blockchain technology, basic introduction of Ethereum and its architecture and basic introduction of DApp and its architecture in section 2. Subsequently we explain various kinds of vulnerabilities present in DApps in section 3. Next, we performed a case study on several DApps of from finance, game, social and media domain in section 4. Afterward, we have concluded this paper with finding of audit on mentioned DApps in section 5.

2. Blockchain and Ethereum Overview

Blockchain is a collection of nodes connected through each other in a similar pattern as if nodes in linked-list data structure where each node has address of previous node. Each node has data and a hash (a mathematical operation which ensures that no two values result into a same value) of data of a previous node. An authenticity and validity of a transaction is mentioned by using private-public asymmetric keys encryption; a user signs a data of a transaction that they want to send. After receiving that transaction, a node verifies its identity using pubic key (sender's identity) and history (whether sender is eligible to make a transaction or not). After that a transaction is being verified and it is being added to a block, this block will be the part of blockchain after solving a computation complex problem and

but easily verifiable by other nodes, this problem is called a Proof of Work (PoW) (Zhang, Xue, & Liu, 2019).

Ethereum is one of the implementations of blockchain with smart contracts which is a self-executed program run over EVM. With smart contracts DApps could easily be developed and Ethereum is now a de facto standard for DApps (Chen, Pendleton, Njilla, & Xu, 2019).

Ethereum has a four-layers architecture:

1. Application layer: it a layer where user interacts with the Ethereum blockchain. Contract accounts and external Owned Accounts(EOA) are two types of accounts present in Ethereum bloackchain. EOA is used to keep data related to user like fund and it is associated with public key of a user that is used to authenticate a user while performing a transaction. On the other hand, contract address stores the byte code of the smart contract.
2. Data layer: It is a layer where a transaction is stored, every transaction involves a sender and a receive which could be an EOA or another contract account. A sender signs a transaction with its private key and send it to Ethereum peer-to-peer network where one of the nodes pick that transaction and verifies it using sender public key data of a sender present on blockchain. After that, transaction is being sent to next layer of the Ethereum blockchain.
3. Consensus layer: In this layer a block is added to a blockchain after going through consensus procedure (PoW). A new block is added to the Ethereum blockchain in every 14 seconds to be part of that blockchain.
4. Network layer: This layer contains copy of entire blockchain, information about the network of nodes and routing table for the information of other nodes.

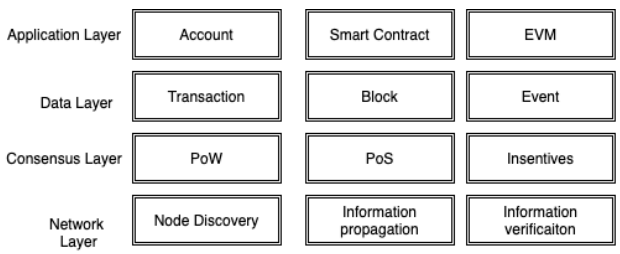

Fig 1. Architecture of an Ethereum blockchain.

DApp architecture:

In a DApp, a website is used to connect a user to an address on Ethereum blockchain using client like MetaMask ("MetaMask,"). MetaMask stores user's private key and balance. After that, a web session is being created and according to the use case of the application some data is saved at application server, cached data for faster retrieval and stored hash of user's data to offload blockchain. A server for a DApp uses web3.js ("Web3.Js - Ethereum JavaScript API — Web3.Js 1.0.0 Documentation,") to interreact with Ethereum blockchain, web3.js is a framework to connect local or remote network of node of a blockchain via HTTP, IPC or WebSocket.

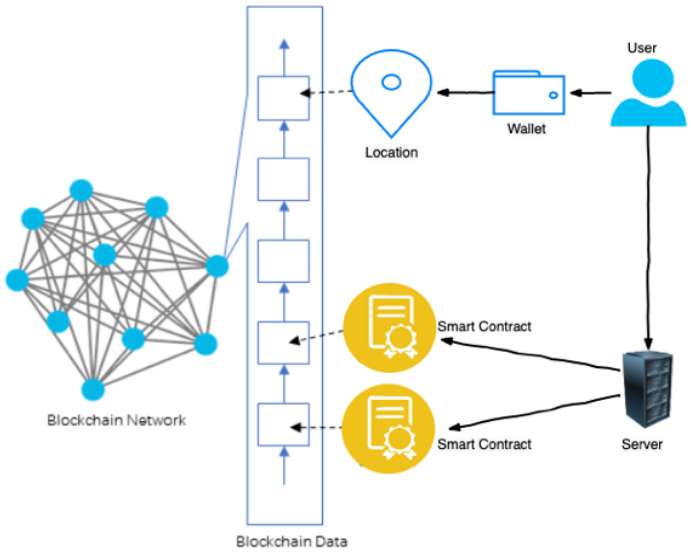

Fig 2. Architecture of a DApp web application and its interaction with a blockchain.

3. DApp Vulnerabilities

Web could be the one of the focal points for the attacker because it has the information of a user's wallet and private key and once an attacker has access to user's key then he can impersonate a user and can get access and perform transactions and after that they might have access to the servers where they can get personal information of a user. It can easily be avoided by adding two-step verification.

On the other hand, smart contracts are easily be available to anyone because it is deployed on an Ethereum blockchain in the form of bytecode which can be converted back to byte code by using several tools available online and can identify potential security risk and could easily be exploited by an attacker.

In this section we have listed most common vulnerabilities present in a DApp which can be found either in the web application side or at smart contract side. These vulnerabilities have been classified majorly into three sections according architecture layer in which it lies (Atzei, Bartoletti, & Cimoli, 2016).

Level	Cause of Vulnerability
Solidity	Re-entrancy, Exception disorders, Call to unknown, Keeping secrets, Gasless send, Type casts
EVM	Stack size in limit, Immutable bugs, Ether lost in transfer
Blockchain	Time constraints, Unpredictable state, Generating randomness

Table 1. Smart Contact Vulnerability

1. Re-Entrancy: Ethereum smart contact is capable to use code of other smart contacts by calling them where it is needed. Since contacts also deal with Ether, it is very likely to send either to external addresses. Every time an external call is being triggered to send an Ether by a caller; an attacker might highjack calling and call fallback function which eventually call itself (re-enter), this iteration will go on and on until all the ethers of the sender is being finished because it does not require any validation. This was the vulnerability with DAO attack.
2. Delegatecall: It is the vulnerability with Parity wallet, it was occurred because of calling an external library and it will become the part of caller's contract bytecode as if it is the part of it and can modify state of caller's state. It can be prevented by declaring a sharable contact as delegatecall.
3. Arithmetic overflow/underflow: This vulnerability was first occurred with BEC tokens. It happens when a result of a calculation fall out of the range of default data type of a solidity. So an attacker might try to change its balance or other state of the variable. It can be avoided using library SafeMath.
4. Predictable Variable: This vulnerability happened because of uses of a predicable function or a dependent variable whose state could be known, for determining the state of a smart contract or randomness state e.g. block's height or timestamp. For example, a smart contact generates a desk of card using timestamp of EVM, this will open a door for an attacker to guess the desk of cards easily.
5. Tx Origin Attack: It is the reason for phishing like attack. It happened because of global variable of smart contract tx.origin. it has an address of that contract which

have called contact in action. It might be used as authentication and if it the case then by introducing a middle contact and using an external call function this vulnerability can be exploited.

6. Weak password: This vulnerability is related to human error, it was first seen in Enigma DApp. It happened because of the low entropy password and reusing previous insecure password. Once an administration password is being compromised then an attacker can change a web page linked to a DApp as per his preferences and steal data linked to normal users of the DApp.

7. Cross-Site Scripting (XSS): This vulnerability is very general in a web application but it was first seen in EtherDelta DApp for the first time. An attacker injects JavaScript into a user's browser and steals data such as private key from user's system and pretend to be an authenticate user to perform an operation. It can be prevented by making signing a transaction offline process so that it would not be loaded into a browser.
8. Cookie Replay: It is used to prevent vulnerabilities like CSRF double cookie. HTTS server marks cookies data as https-only so that no middleman attacks can take place and also it will not allow scripts from other domain can read it. So, attacker can be acts on behalf of authenticated user if they steal the user's cookie till cookies are valid or they might inject JavaScript that can steal cookies and send it to an attacker.
9. Injection: This vulnerability is related to using incorrect HTTP methods like sending sensitive data using GET request like password or sending data that is malicious and can cause SQL injection or cross-site scripting injection. Developer might use http GET verb for deleting some resources on the server which is a security issue. These inputs generally get to server via using input text or input textbox.
10. Broken Authentication: This vulnerability happens because of broken authentication by any means, it includes social engineering, brute force to crack wallet, session id present in URL or leaker on authentication token like JWT in a request, response or in the localStorage of a browser. Attacker might try brute force method to crack an authentication key apart from the other options available for the scanning.

4. Case study

In this section, we have explored DApps related to domains e.g. finance (Nexo, Gimmer), games (0xUniverse**,** Word of Ether), social(2Key , Sapien Network) and media (PixelCons, Documentary). Security issues have been analyzed using audit tools e.g. Mythril

for smart contract and Nikto for Web. Results have been listed in the form of a table with severity level of each of the vulnerability.

Finance:

1. **Nexo** ("Instant Crypto Credit Lines," n.d.): It is a lending platform based on cryptocurrency an user owned, she can add her cryptocurrency to wallet at Nexo and can take loan instantly. It is operated worldwide and with more than 40+ currencies.

Table 2. Smart Contract Mythril Audit

SWC ID	Severity	Contract Name	Function Name
107	Low	Token	transferERC20Token(address,address,uint256)
113	Low	Token	transferERC20Token(address,address,uint256)
110	Medium	Token	transferERC20Token(address,address,uint256)
110	Medium	Token	increaseApproval(address,uint256)

Table 3. Web Audit using Nikto

Issue	Description
Server	cloudflare
X-XSS-Protection	Absent
cf-request-id	Unknown header present
Strict-Transport-Security	Absent
X-Content-Type-Options	Absent

2. **Gimmer** ("Gimmer - Bitcoin Bot," n.d.): It is a decentralized trading bot. It uses user's cryptocurrency exchange account and set few parameters with the help of the user after that with the help of indicator and AI performs trading.

Table 4. Smart Contract Mythril Audit

SWC ID	Severity	Contract Name	Function Name
110	Medium	StandardToken	increaseApproval(address,uint256)
110	Medium	GMRToken	mint(address,uint256)

Table 5. Web Audit using Nikto

Issue	Description
Server	nginx/1.14.0
Cookie session	without the secure flag
Content-Encoding	deflate
x-powered-by	Express

Game:

1. **0xUniverse** ("0xUniverse Is a Blockchain-Based Space Strategy," n.d.): It is a DApp game where players can build their transport mode, visit different planets, capture a planet.

After running analysis on Smart Contract no issues were found.

Table 6. Web Audit using Nikto

Issue	Description
Server	nginx/1.10.3 (Ubuntu)
X-Frame-Options	absent
X-XSS-Protection	absent
Strict-Transport-Security	absent
Expect-CT	absent
X-Content-Type-Options	absent

2. **Word of Ether** ("World of Ether | Decentralized Collectible Dueling Game | WoE," n.d.): It a game related to monsters where you can sell, breed to become more powerful, fight for level up. It's similar to cryptikitties game.

Table 7. Smart Contract Mythril Audit

SWC ID	Severity	Contract Name	Function Name
116	Low	PreSale	bulkPurchageEgg()
105	High	PreSale	Withdrawal()

Table 8. Web Audit using Nikto

Issue	Description
Server	nginx/1.14.0
X-Frame-Options	absent
X-XSS-Protection	deflate
Strict-Transport-Security	Express
Expect-CT	absent
X-Content-Type-Options	absent

Social:

1. **2Key** ("2key Network," n.d.): It is a social platform to generate a trackable link which can be used as to track people who had clicked on that link and that way a list of users or target audience that can used to get desired result and incentives users.

After running analysis on Smart Contract no issues were found.

Table 9. Web Audit using Nikto

Issue	Description
Server	openresty
X-XSS-Protection header	absent

x-lambda-id, x-served-by, x-timer, x-cache, x-cluster-name	unknown header present
Strict-Transport-Security	absent
X-Content-Type-Options	absent

2. **Sapien Network** : It is a social network for users but to keep data with their instead of selling to big giant for their advantages. It is a DApp with keep intension to not sell user's data.

   After running analysis on Smart Contract no issues were found.

Table 10. Web Audit using Nikto

Issue	Description
Server	cloudflare
x-powered-by	Next.js
X-Frame-Options	absent
X-XSS-Protection	absent
X-Content-Type-Options	absent

Media:

1. **PixelCons** ("PixelCons,"): It is a DApp to create, browse and collect minimalist pixel art. Every art is unique and owned by a user which is secured on blockchain platform Ethereum and safe from plagiarism.

   After running analysis on Smart Contract no issues were found.

Table 11. Web Audit using Nikto

Issue	Description
Server	No banner retrieved
x-served-by	cache-akl10323-AKL
X-XSS-Protection	absent
x-cache, x-timer, x-served-by	HIT, S1590924087.453810, cache-akl10323-AKL
Expect-CT	absent
X-Content-Type-Options	absent

2. **Documentary** ("Documentary — Publish Documents,"): It is a DApp for storing documents on a Ethereum blockchain, here documents can be searched and retrieved and it is free to use and only transactions cost incur.

   After running analysis on Smart Contract no issues were found.

Table 12. Web Audit using Nikto

Issue	Description
Server	Google Frontend

X-Frame-Options	absent
X-XSS-Protection	absent
x-cloud-trace-context	54e56f39884e1ebcdf24474096d58fe8
X-Content-Type-Options	absent
X-Content-Type-Options	absent

5. Conclusion

With the evolution of DApps there is new era of data without hold of an central authority. On an Ethereum blockchain platform a user can have virtual assets which they can keep with themselves and have monetary value attached to it. Since there money is involved. therefore security of a DApp is important aspect to be taken into consideration. We have explained how DApps have been attacked and we have explained various kinds of vulnerabilities present along the technology stack of a DApp. We have also discussed eight DApps and scanned them to find various vulnerabilities present with them by using tools Nikto and Mythril for the domains like finance, game, social and media and tabulated all findings.

References

0xUniverse is a blockchain-based space strategy. (n.d.). Retrieved May 31, 2020, from 0xUniverse website: https://0xuniverse.com/

2key Network. (n.d.). Retrieved May 31, 2020, from https://www.2key.network/

A DeFi App Was Just Hacked for Over $300,000 in Ethereum & Bitcoin. (2020, April 19). Retrieved May 28, 2020, from Bitcoinist.com website: https://bitcoinist.com/a-defi-protocol-was-just-hacked-for-over-300000-in-ethereum-bitcoin/

Al-Jaroodi, J., & Mohamed, N. (2019). Blockchain in Industries: A Survey. *IEEE Access*, *7*, 36500–36515. https://doi.org/10.1109/ACCESS.2019.2903554

Atzei, N., Bartoletti, M., & Cimoli, T. (2016). *A survey of attacks on Ethereum smart contracts* (No. 1007). Retrieved from http://eprint.iacr.org/2016/1007

Buterin, V. (n.d.). Ethereum/wiki. Retrieved May 28, 2020, from GitHub website: https://github.com/ethereum/wiki

Chen, H., Pendleton, M., Njilla, L., & Xu, S. (2019). A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses. *ArXiv:1908.04507 [Cs]*. Retrieved from http://arxiv.org/abs/1908.04507

documentary—Publish documents. (n.d.). Retrieved May 31, 2020, from https://www.stateofthedapps.com/dapps/documentary

etherscan.io. (n.d.). Ethereum Daily Transactions Chart | Etherscan. Retrieved May 28, 2020, from Ethereum (ETH) Blockchain Explorer website: http://etherscan.io/chart/tx

Gimmer—Bitcoin bot. (n.d.). Retrieved May 31, 2020, from https://gimmer.com/

Hackers Steal $49 Million in Ethereum From Upbit Exchange. (n.d.). Retrieved May 28, 2020, from https://www.bankinfosecurity.com/blogs/hackers-steal-49-million-in-ethereum-from-upbit-exchange-p-2825

Instant Crypto Credit Lines. (n.d.). Retrieved May 31, 2020, from Nexo website: https://nexo.io

MetaMask. (n.d.). Retrieved May 31, 2020, from https://metamask.io/

Nakamoto, S. (n.d.). *Bitcoin: A Peer-to-Peer Electronic Cash System*. 9.

N-transactions. (n.d.). Retrieved May 28, 2020, from Blockchain.com website: https://www.blockchain.com/charts/n-transactions

Overview · Smart Contract Weakness Classification and Test Cases. (n.d.). Retrieved June 1, 2020, from http://swcregistry.io/

PixelCons. (n.d.). Retrieved May 31, 2020, from https://pixelcons.io/

Sapien. (n.d.). Retrieved May 31, 2020, from Sapien website: https://www.sapien.network/

State of the DApps—DApp Statistics. (n.d.). Retrieved May 28, 2020, from https://www.stateofthedapps.com/stats

web3.js—Ethereum JavaScript API — web3.js 1.0.0 documentation. (n.d.). Retrieved June 1, 2020, from https://web3js.readthedocs.io/en/v1.2.8/

World of Ether | Decentralized Collectible Dueling Game | WoE. (n.d.). Retrieved May 31, 2020, from https://www.worldofether.com

Zhang, R., Xue, R., & Liu, L. (2019). Security and Privacy on Blockchain. *ArXiv:1903.07602 [Cs]*. Retrieved from http://arxiv.org/abs/1903.07602