Learning Best Practices from Web Applications to Avoid Similar Security Vulnerabilities in Decentralized Applications

Mahmoud Aboualy

Bachelors Thesis

Information Technology

2019

DEGREE THESIS	
Arcada	
Degree Programme:	Information Technology
Identification number:	20495
Author:	
Title:	Learning best practices from web applications to avoid similar security vulnerabilities in decentralized applications
Supervisor (Arcada):	M.Sc. Magnus Westerlund
Commissioned by:	

Abstract:

Our sensitive information is disseminated across a large number of websites throughout the internet. Most business organizations and associations that individuals deal with have accurate and effective security programs to safeguard our confidential information, but this does not prevent cybercriminals from having a try to snatch our data. Vulnerabilities in traditional web-applications are often regarded as one of the reasons that allows cybercriminals to hack into systems and purloin thousands of people's personal information. Despite the proven benefits of using web applications, it is imperative to consider using more secured technologies, associated with Blockchain, and more specifically to switch to decentralized applications. Decentralized Application or DApp for short is an application with source code that anyone can examine, amend, and enhance and connects independently on a decentralized public Blockchain. DApp is stored in a distributed file system, such as IPFS, but transactions among peers are verified through the smart contracts that run on top of the blockchain. This thesis provides a review of common security vulnerabilities in web applications and a comparison how decentralized applications deal with the same vulnerabilities. This provides an insight into how applications may be created in the future.

Keywords:	HTML, CSS, JavaScript, React.js, web3.js, MetaMask, Truffle-framework, IPFS, Ganache, Ethereum, Smart contracts (Solidity)
Number of pages:	46
Language:	English
Date of acceptance:	

Table of Contents

Figures

Tables

1 INTRODUCTION

Over the past few years, there have been influential and noteworthy technological advancements, which in turn have had either direct or indirect impacts on our daily lives. Blockchain technology is one of those technologies that managed to draw attention to its essentiality and novelty by many non-professionals, specialists, and experts in various fields such as IT, business, etc. Since the emergence of some rather controversial cryptocurrencies such as Ethereum Litecoin, Bitcoin, etc., there has been much talk about the technology underlying those cryptocurrencies and especially Bitcoin.

Bitcoin is defined as the first decentralized digital currency, p2p payment network without a central point of control. Bitcoin is considered as an example of a decentralized platform, and thus, the term decentralized platform has started to acquire widespread common usage. Bitcoin has blazed a trail for developing a new kind of application software program that can make use of its exceptional and new-fangled technology, namely, blockchain, the core component of Bitcoin.

This chapter begins with a brief history of JavaScript, Blockchain, Ethereum Blockchain, and distributed ledger technology, followed by the research aim, and after that comes the research problems and questions.

1.1 Front-end development with JS

In May 1995, within a few days of hard work, Brendan Eich, an American technologist and the co-founder of both Mozilla and Firefox, created JavaScript, while working at a company called Netscape Communications Corporation.

The newly created language was initially named Mocha, and in September 1995 it was decided to replace the naming with “LiveScript”, but then again, with JavaScript within the same month of September, as they aimed to piggyback on the great success and fame of Java programming language

In 1996, a number of script technologies such as JScript and VBScript were published by Microsoft and later in the same year, JavaScript was sent to ECMA to get an official

standard for it. The ECMA international trademarks the ECMA script to indicate to the standard for the language.

Scripting language such as Jscript, JavaScript, ActionScript and VBScript are an implementation of those standards in one way or another. However, JavaScript has stayed the renowned implementation of ECMAScript specification.

ECMAScript 1 showed up in 1997, while not very far away from that data, the second version of ECMAScript came out, followed by ECMAScript3 in 1999 (Leprohon, M.A., 2017), whilst there was never ECMAScript 4, the fifth version of ECMAScript was introduced in 2009 and ES6 was eventually released in 2015. (Materzok, M., 2016)

1.2 Blockchain

Blockchain, the technology probably to have the uttermost influence on the next few decades on all types of businesses in lots of different areas has arrived.

Although, it is claimed that blockchain was contrived in 2008 by an anonymous person (or even a group of them) under the pseudonym "Satoshi Nakamoto" (Lemieux, P., 2013.), the actual history of Blackchin technology dates back to the 1990s.

In 1991, two physicists, namely Stuart Haber and W. Scott Stornetta worked together on a feasible solution for time-stamping digital document (Haber, S.A. et el., 1992), in order to avoid them getting manipulated or backdated. They proposed a system that makes use of "a cryptographically secured chain of blocks" (Wikipedia, 2019) to maintain the time stamped documents.

Merkle trees were integrated into the design shortly after the start of the project for the purpose of enhancing its efficiency by letting many documents to be combined into a block. However, the idea did not get adopted and the technology was disregard in 2004.

In 2004 a developer called Hal Finney invented RPOW scheme (stands for Reusable Proof-of-Work) (Karlstrøm, H., 2014) that significantly inspired Bitcoin. PROW system helped soundly in preventing double spending problem by maintaining the possession of tokens registered on a reliable network that is designed to make users across the world able to verify its veracity and probity in real time.

In November 2008, Satoshi Nakamoto published the bitcoin whitepaper on the internet representing Bitcoin for the first time as a Peer-to-Peer Electronic Cash System (Chuen, D.L.K. ed., 2015)

1.3 The Ethereum blockchain

Ethereum was originally innovated by a Canadian programmer called Vitalik Buteri who is very fond of the decentralization concept in principle. While him being a part of the Bitcoin community, he suggested what is so-called the" white paper" (Buterin, V., 2017) which was essentially a kind of virtual machine taped on as an inordinately protocol on top of prime coin. The main idea behind the white paper was to add a scripting programming language to blockchain in order to give individuals a much easier time building things such as Namecoin, etc, but after failing to come an agreement, he recommended evolving a new platform that allows the use of a scripting.

For the first time, between the 22nd of July and the 2nd of September in 2014, Ethereum foundation conducted an online crowdsale for selling Ether at a proximately price of $0.31 per token. The main idea behind the conducted crowdsale was to provide financial support for the further development of Ethereum platform.

Ethereum is a public, distributed, blockchain-based decentralized platform designed to allow applications to run precisely as intended (with no chance for centralized control, TPI (Third-party interference) or swindle).

One of the greatest strengths of Ethereum platform, is that the decentralized association and the escrow agreement are not required to make account of what type of account every party to the contract is. (Buterin, V., 2014)

The Ethereum platform incudes a plentiful number of advantages, for instance, it enables its users to generate their own virtual cryptocurrencies according to their needs or wishes, provides its users with the ability to plan any kind of independent contracts that have no more need for middleman or supervisor, etc.

1.4 Distributed ledger technology (DLT)

Distributed ledger technology (DLT) is an old concept that has been for a while and dates back at least 30 years. Distributed ledger technology is termed a consensus of shared, proliferated and synchronized digitized information distributed all over the world. Distributed ledger technology clarifies the various implementations followed with a view to achieving consensus in distributed ledgers while achieving consensus mainly falls into two main categories based on the parties involved in the consensus process. (Westerlund, M. and Kratzke, N., 2018)

There are mainly three generations of Distributed ledger technology, each of which has important strengths and weaknesses. The first generation is effectively connected with Bitcoin blockchain that provided the ability to safely maintain transactions in an immutable ledger and proof-of-work also known as consensus mechanism. Proof-of-work is a protocol, or a system followed to validate transactions, deter attacks on the chains, add a new block to the chain and agree on the state of the network. In the second generation, Ethereum Virtual Machine (EVM) which is considered to be a runtime(sandboxed) environment for smart contracts based on Ethereum network, was introduced. Ethereum Virtual Machine allows running applications that are locked off from the outside world.

The third generation testifies to the shift from consensus based on Proof of Work (PoW) to one based on Proof of Stake (PoS). Proof of Stake is described as one of the most frequently used algorithms for achieving consensus in blockchain sharing the same purpose with proof of work algorithm and differentiating from proof of work in the way it is achieving that purpose. (Westerlund, M. and Kratzke, N., 2018)

1.5 Research aim

The main purpose of this thesis is to provide an understanding of common security vulnerabilities in Web applications, by reviewing literature and comparing how Web apps and DApps deal with the same vulnerabilities. This provides an insight into how applications may be created in the future. This will be achieved by:

- Building a decentralized application using React.js, web3.js MetaMask, Solidity Truffle framework and Ganache.

- Analyzing the basic components of both centralized and decentralized applications.
- Criticizing the literature concerning the most common security vulnerabilities in Web applications.
- Identifying the potential solutions introduced by decentralized applications to the most common security vulnerabilities in traditional Web application.

1.6 Research problems and questions

The essential questions that the search will aim to answer, are:

- What are the main components of centralized & decentralized applications?
- How does the server-side in traditional Web applications differ from its counterpart in decentralized applications?
- How does the client-side interact with the server in both the traditional Web application and the decentralized application cases and what are the common techniques used for achieving that interaction?
- What are the three most common security vulnerabilities in Web applications and how they are handled in in decentralized applications?

2 LITERATURE REVIEW ON CENTRALIZED & DECENTRALIZED APPLICATION

In this chapter, first, I am going to perform a general analysis of the core components of both web applications and decentralized applications. Secondly explain in detail some of those components. Lastly, present the most common security vulnerabilities in web applications that lead to data breaches and the possible ways to detect and prevent them.

2.1 The core components of web apps and DApps and the interaction between them.

Almost every Web-based database application existing on the internet has at least three major components, namely a client-side, a server-side and a database server. DApps differ

from Web apps in the server-side where smart contracts and Blockchain are used instead Table 1).

Table 1 A comparison between the client-side in both web apps and DApps

	Web-app	**DApp**
Client-side	The client side is simply everything that the end-user or the client sees and interacts with (Jadeja, Y. and Modi, K., 2012). There are a variety of technologies that can be used for Web development. However, CSS, HTML, and JavaScript are the most commonly used ones.	
	JavaScript, the world's most common scripting language, has a multitude of frameworks and libraries including: • AngularJS (Supported by Google) • ReactJS ((Supported by Facebook) • VueJS • Meteor • Aurelia • Backbone.js • Ember.js Most of them are meant for Web development. Each library/framework has advantages, disadvantages and a community standing behind.	A decentralized application or a DApp can have its client side written in any scripting language that makes use of API calls to its server side. However, there are a number of libraries and frameworks that are commonly preferred such as • ReactJS (There are many ReactJS truffle boxes available to DApp developers allowing them to easily build DApps) • Meteor (Meteor is considered to be a complete framework that includes all the tools, necessary for development. In addition, Fabian Vogelsteller, a key member of Ethereum Foundation wrote a book about it and recommends using it for building Ethereum DApps • EmbarkJS (EmbarkJS is a framework that facilitate building and deploying DApps. Embark has several advantages over other libraries and

		frameworks, since it allows DApp developer to smoothly perform a wide variety of tasks such as automatically deploy smart contracts (and redeploy if necessary), deploy the entire app to Swarm or IPFS, integrate Embark with any other technologies, pipelines and tools such as ReactJS, grunt, webpack and gulp, etc. (github/embark, 2019)

One of the major differences between Web applications and DApps is the database that can be used in both of them, and therefore, it is necessary to bring to light some examples of those databases (Table 2).

Table 2 Examples of databases that can be used in Web application and DApps

Database	There are a wide variety of databases that are used for Web applications such as relational databases (the most commonly used one), distributed database, centralized database, distributed database, OOD (stands for object-oriented database) NoSQL database, private database. etc.	Alongside Blockchain which is described as a public digitalized ledger that is distributed and decentralized, there are a group of decentralized databases (decentralized hosting) that can be used including but not limited to • IPFS (stands for InterPlanetary File System) • Swarm which is "a distributed storage platform and content distribution service" (swarm-guide.readthedocs.io, 2019) • BigChainDB (stands for InterPlanetary DataBase),

		• Distributed file storage, etc. (Dmitry Kochin, 2017)

Due to the difference in server-side technology used in both Web applications and DApps, the communication techniques followed for facilitating the interaction between the server-side and the client-side are different. The table below gives information on some of those techniques (Table 3).

Table 3 A comparison between the interaction between the client-side and the server-side in both Web apps and DApps

The interaction between the client side and the server side	**Web API**: APIs stands for application programming interfaces and they are constructs made ready in programming languages with a view to facilitating developers' employment. (developer.mozilla.org, 2019) APIs in client-side JS go under two classes third party APIs and Browser APIs. **HTTP**: HTTP is defined as an application-level protocol with the agility, resiliency and rapidity indispensable for distributed, collaborative, hypermedia data systems. (Berners-Lee, Tim, et el.). HTTP has a set of properties that make it more preferable to other communications protocol, for instance, it complies with the standard client-server model, it is stateless (which means that each transaction carried out via HTTP is separate and not associated with any other transactions, so the transaction is gone the moment that the transaction ends.), etc.	Decentralized application has no central server but decentralized ones which means that every user of the Ethereum dApp either uses his own blockchain node (as a server) or another person's blockchain node from the network. All nodes are generally arranged into P2P network and for the client-side to interact with those nodes an Ethereum JavaScript API known as "web3" is most frequently used.

	REST: REST (stands for Representational State Transfer) is described as an architectural style or a design concept for exchanging information in well-defined formats so as to enhance interoperability. HTTP is the most frequently used communication protocol in RESTful Web services. REST adds decidedly no particular functionality to HTTP. All REST services are considered to be APIs, not vice versa.	

The table below summarizes the differences between the web application server and the decentralized application server (Table 4).

Table 4 A comparison between the server-side in both Web apps and DApps

Server-side	Server-side programming language is a language designed to run on a Web server. For instance, • PHP (PHP is known as the most frequently used scripting language on the server-side) • Node.js • Python • Java • C++ • ASP.NET	There are many (blockchain) programming languages that can be used for writing smart contacts on Ethereum Blockchain that includes a number of the traditional server-side programming languages such as Java, Python and C++. In addition, • Solidity (A high-level and object-oriented programming language) • Simplicity (A low-level language) • Rholang (A fresh programming language for coding smart contracts)

2.2 Web application

A web-application is defined as a conceptually centralized application program which is maintained on a remote server allowing clients all over the world to access through the internet, by using a web browser of their choice.

All Web-applications have three fundamental components, namely a web server, a client-side and a database.

2.2.1 Web server:

In a technological sense, a web server is well described as a computer program, a somewhat distinct computer, in a sense that it is meant to serve other computers. Servers are commonly connected either to a personal network known as an intranet which is limitedly accessible to an association's staff, or the internet; accordingly, client computers have to be connected to the same network as server computers and simplified vision of how the communication between them occurs would be:

A user types a specific URL into a web browser, which in turn, splits the URL into its basic components, namely scheme (either http: or https), domain name, and path, and construes the domain name to an IP Address with the help of DNS (Domain Name System), the IP Address is then used to make a connection to the targeted server by sending an HTTP using the right request method such as GET, POST, DELETE, etc. the serve handle the request and sends back a response to the request with a particular status code, and a response header which provides a more exhaustive context of the response.

Servers interact with other servers every now and then, for instance, the web servers at a company like Google will communicate to the servers at Arcada (University of applied sciences) to get data that lets google searches to find Arcada programmes and admissions, while client computers scarcely interact with one another without passing through a web server.

There are multiple types of connections to servers, for instance,

- FTP means file transfer protocol is well termed as a standard network protocol that allows files to be transmitted between servers and clients or heterogeneous computer systems via the internet. (Gien, M., 1978)
- SSH secure shell), is a protocol used for secure authenticated and encrypted network activities such as secure remote login through a secure network. (Ylonen, T. and Lonvick, C., 2005.)
- SFTP means SSH file transfer protocol a network protocol "that provides file access, file transfer, and file management" through any dependable information stream. (Wikipedia, 2019)

2.2.2 Database

A database is defined as a collection of data, which is chiefly structured for making a number of data-processing operations such as searching, retrieving or inserting further data series, easy and more efficient. The information maintained in databases can be altered, retrieved and erased as necessary. Database can be accessed in diverse ways; However, a more common way is through the internet.

There are many types of databases, each of which has its own perks, and its own advantages.

- **Centralized database:** It defined as a collection of data/information, maintained in computerized form in one location. (Árnason, V., 2004.). Data is stored and altered from that specific location only and commonly. This type of database is usually used by organizations such as universities, banks etc. Carrying out data-processing-related operations, in case of using DPs (decentralized databases) can vastly be done through distributed parallel computations.
- **Distributed database**: It can be described as a system that makes use of sharding and more likely replication to enhance efficacy and augment dependability. Distributed database is made up of a single rational database that is divided into a set of fragments each of which is maintained on one or more physical places on the same network or on completely distinct networks.
  There are two traditional ways of partitioning, namely horizontal partitioning and vertical partitioning.

Horizontal partitioning referred to as “sharding” and it is a type of partitioning where developers work on splitting the different rows into different tables.
Vertical partitioning is the second type of partitioning where developers tend to split the columns instead of rows and they are required to assemble and reassemble the data in order to able to read and write it

- **NoSQL database:** The term 'NoSQL database', suitably, is the substitutional to the classic relational databases where data is set in tables and also data schema is accurately planned beforehand, before even building the database. NoSQL databases are able to maintain structured, semi structured or unstructured data. (Mohamed, M.A., et el, 2014)
- **Relational databas**e: The term ' relational database ', appropriately, is a set of abilities that improve requisite database operations for business purposes. A relational database maintains data in an organized format known as a schema to make sure the integrity of the information. Relational database is an efficacious way of arranging data with respect to storage utilized

2.2.3 Client-side JavaScript framework

Comprehending what precisely JavaScript is and how the web browser deal with it, gives a better eye at how things are going under the hood. Each webpage available on the internet comprises of a combination of CSS, HTML and JavaScript.

JavaScript or the language of the browser has always been considered as one of the most widespread programming or scripting language on the Internet for it being a dynamic, responsive, flexible, lightweight, easy-to-learn language. In addition, JavaScript is also categorized as a high-level, object-oriented object-based, interpreted and weakly typed language.

High-level language indicates that the instructions come in a way that is more flexible, understandable, readable and natural to humans, the object-oriented feature is a practical and beneficial programming methodology which mainly supports modularity in design and software reuse. (Snyder, A., 1986) and object-based means that the language does not have classes but objects (Thiemann, P., 2005) and relies on prototype objects to model

inherence, weekly typed implies that distinct forms of data can be handled inter-changeably with one another, while the feature interpreted comes from the fact that at runtime it is automatically interpreted by the web browser. However, one may argue that any programming language can be classified as either compiled or interpreted depending more on the implementation of the developer itself, meaning that it is not a specification of the language.

Scripting with JavaScript can be either on the server side or on the client side. However, it is most ordinarily utilized on the client side. Client-side scripting occurs in the web browser or at any rate on the client computer

JavaScript was mainly designed to run on an assortment of web browsers. Today, almost all modern web browsers come with a built-in ECMA Script Engine for interpreting JavaScript code line by line or even more accurately byte by byte and executing it.

2.3 Decentralized Application (DApp)

Bitcoin's white paper was introduced by Satoshi Nakamoto describing Bitcoin as “a Peer-to-Peer Electronic Cash System” (Nakamoto, S., 2008) making it possible for online payments to be performed without a central point of control or middleman.

DApp is short for decentralized application and it is an open source application that has its server-side code running on a decentralized P2P network such as Ethereum while all the records of the applications processes are maintained on Ethereum blockchain. (Figure 1)

Being an open source application signifies that the code is accessible to everyone and can be enhanced by anyone. For the end-users, decentralized applications might not look any different than other applications they use today.

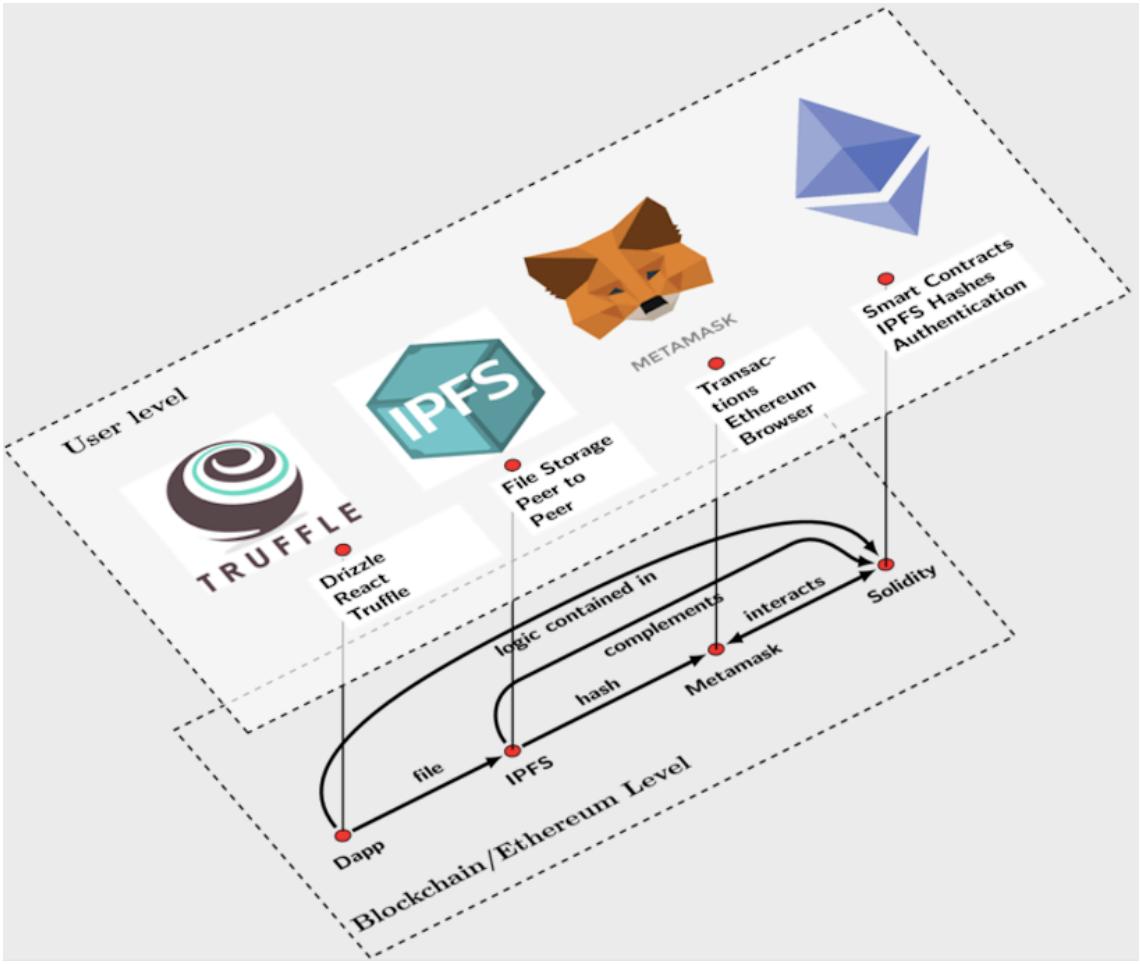


Figure 1 IPFS DApp (friendlyuser.github)

2.3.1 Client-side React

A declarative, dynamic and functional JavaScript library which is managed by Facebook and widely used by web developers to build interactive UIs (stands for user interfaces). (reactjs.org, 2019) React has introduced the concept of Virtual DOM to web application. Virtual DOM is described as a programming concept in which a hypothetical example of the user interface is maintained in memory and synced with the actual DOM. Virtual DOM is implemented in one of the libraries provided by ReactJS, React DOM, which handles the process of updating the DOM in order to be consistent with the React elements. (reactjs.org, 2019)

The prime advantages of ReactJS are the following:

- ReactJS is often described as an easy-to-learn and easy-to-use library. It uses a private syntax named JSX that is described as a kind of unrivalled mixture of both

HTML and JavaScript, and hence a web developer with knowledge in programming will comprehend ReactJS rather straightforwardly

- ReactJS allows developers to smoothly build Single Page Application (SPA). A Single Page Application is a web application or web site that loads an individual HTML page, and then works inside the web browser. Single Page Applications instead of reloading an entire Webpage as a result of user interaction, they dynamically update that Webpage and therefore, they communicate with the server only in a very limited sense
- React supports code reusability and responsive design, making it quite easy to create Android, IOS and web application from a similar code base.

2.3.2 IPFS

IPFS (InterPlanetary File System) is described as a P2P hypermedia protocol, file sharing distributed system. IPFS endeavors to supersede Hypertext Transfer Protocol (HTTP). (ipfs.io, 2019)

InterPlanetary File System mainly aims to subrogate the traditional way of accessing resources using a web browser. The connotation of content-based addressing was introduced for the first time by IPFS which means and contrary to what has been ordinary, a resource existing on the internet can be reached by its content instead of its fixed location ("addressing hosts"). One of the main qualities that is associated with IPFS is "the speed of data transfers" which has been enhanced a lot though IPFS (Cisneros, J.L.B., et al., 2018.)

2.3.3 Smart contract

Smart contracts are ordinary contracts written into lines of code, that is designed to be automatically self-executed precisely as previously programmed and stored across (on top of) a public, distributed, digitized decentralized, blockchain network. (Karafiloski, E. and Mishev, A., 2017)

Smart contracts are "irreversible and immutable" (Luu, L., et el., 2016), which means that after parties concerned come to an agreement on the terms and the contract gets deployed

on blockchain, it is impossible afterwards for the smart cataract to be mutated. Blockchain-related coding is a comparatively considered as a new practice, with limited documentation, guidelines, or security standards to follow.

In August 2014, a new programming language influenced by C++, Python and JavaScript called solidity was initially proposed by a programmer named Gavin James Wood.

Solidity is a "contract-oriented and high-level" programming language that smoothed the way for writing smart contracts. (solidity.readthedocs.io, 2019)

2.3.4 Web3.js

Web3.js is a variety of libraries, holding particular functionalities that facilitate the interaction process with a local or a remote Ethereum node using one of the following communication techniques, HTTP or Inter-process communication (IPC). (Prusty, N., 2017) (official documentation on web3js.readthedocs.io)

2.3.5 Ganache

Ganache is a private blockchain with the objective of facilitating a number of Ethereum development related activities such as deploying contracts, developing decentralized applications, and running tests.

Truffle Suite provides two different versions of Ganache, the first version is a visual interface, commonly known as Ganache GUI, while the second is a command line tool for those who are interested in doing development with the command line. (truffleframework.com, 2019).

My choice went for Ganache GUI for being easy-to-install and easy-to-use. GUI automatically generates 10 accounts, each of which comes laden with 100 (spurious) ethers (With no actual value on the prime Ethereum network). (Figure 2)

Those accounts can readily be exported from Ganache to MetaMask for development purposes by simply clicking on the key icon on the right side, copying the private key and pasting it to MetaMask (Under: My Accounts → Import Account → Paste your private

key string here:). Performing transactions will alter the amount of Ether available and the remaining balance will automatically show up under “Balance”.

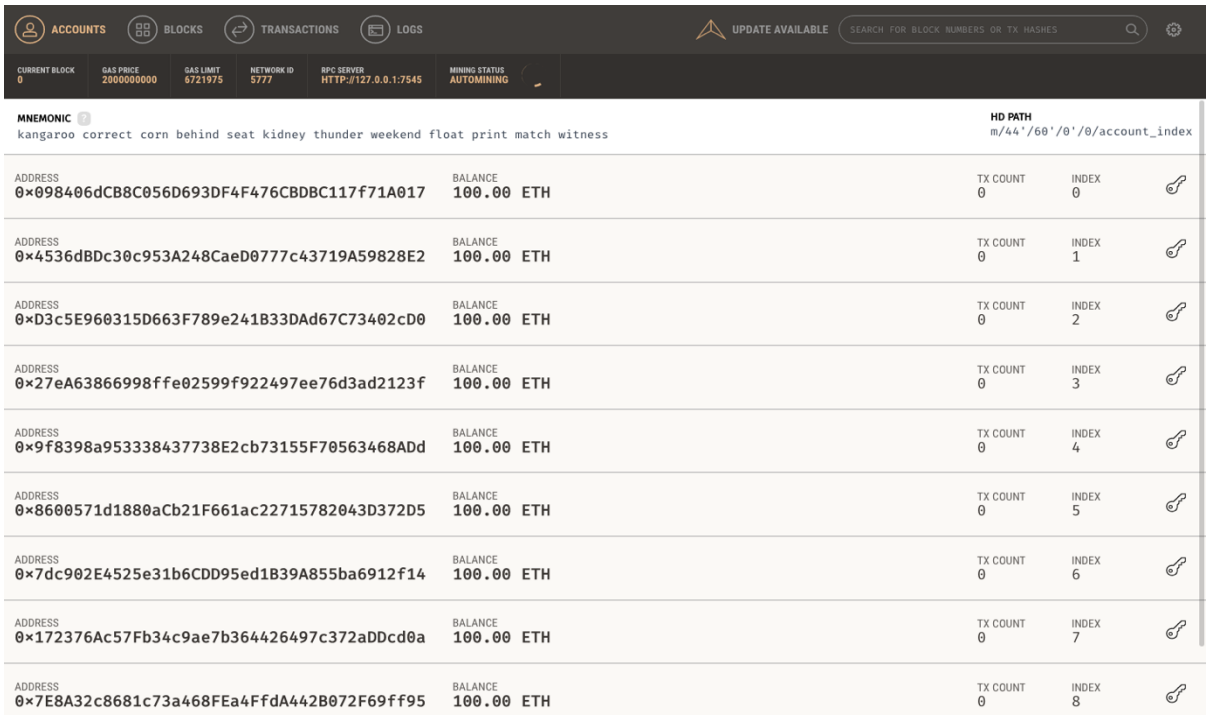
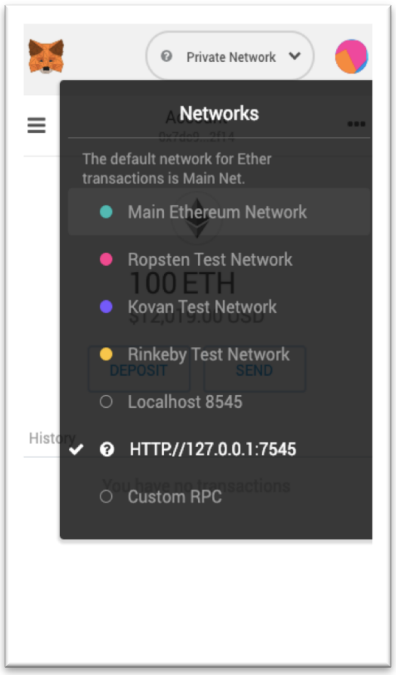


Figure 2 A screenshot of my Ganache GUI

2.3.6 Metamask-extension

MetaMask, which is a web browser extension, built by use of web technology, is destined particularly for use with an overt distributed decentralized blockchain-based network that is known by the name of Ethereum network. MetaMask extension/ add-on is available on a variety of browsers such as Opera, Chrome, Firefox, etc. (metamask.io, 2019)



Installing MetaMask onto one of the supported web browsers, will transform that browser into an Ethereum one. With the help of MetaMask, webpages will able to retrieve in-formation from the blockchain, and users will be able to securely superintend identities and subscribe transactions.

After the first installation of MetaMask, and to get MetaMask connected to Ganache, there is a need to switch from the default network "Test Network" to HTTP://127.0.0.1:7545(http://localhost:7545), the local server where the blockchain runs. The name of network existing at the top will then appear as "private Network".

MetaMask allows creating numerous accounts and to do so, the user needs to press on the round icon, in the top-right corner, and then click on "Create account" or "Import Account".

Ganache provides 10 accounts with fake 100.00 ETH for development purposes, each of which can be used in creating a new MetaMask account. Each account can be named and renamed at any time and will possess its own unique public and private keys. Each of the created account-wide storages is encrypted with the public key associated with the account and internally stowed within the web browser, Metamask is installed on. The data stored in the account vault won't be shared with any third party including the server of Meta-mask. MetaMask makes it possible for its users to send and receive ether as any ordinary wallet application. (Figure 3)

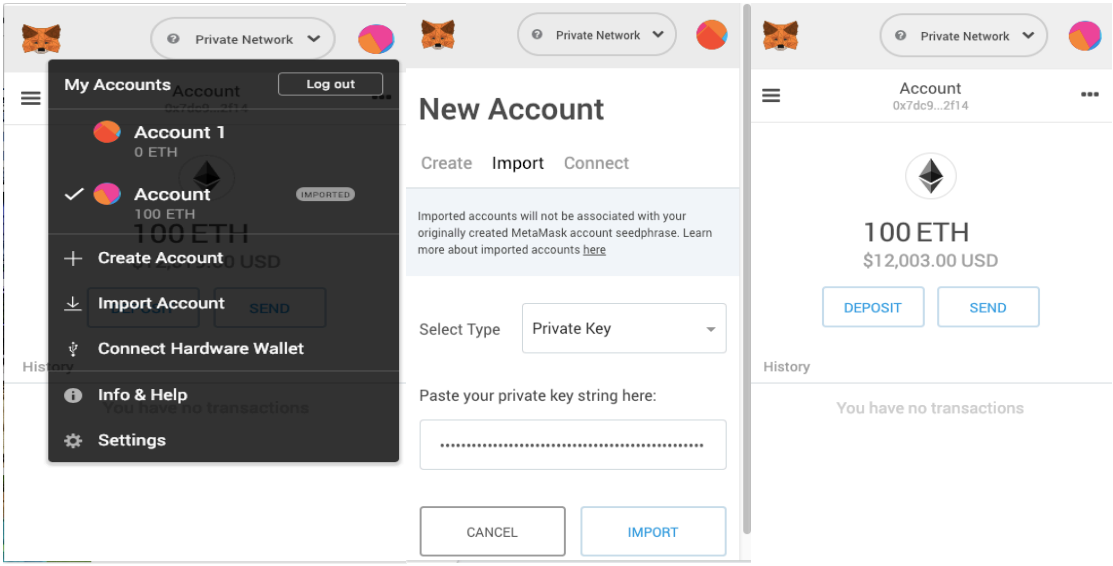


Figure 3 Screenshots of Metamask for Chrome

2.3.7 Truffle framework

Truffle is a potent framework which is commonly described as an integrated environment that enable Ethereum developers to build, develop, test and deploy their decentralized applications on Ethereum network using the EVM (stands for Ethereum Virtual Machine) and smart contracts.

2.4 Applications and security

Data theft and data breaches influence hundreds of millions of people across the world and is on the rise. Numerous companies have undergone a large data breach giving rise to layoffs, lawsuits and deceit once these cyber criminals have our private information. Data breach can also conduce to identity theft permitting someone else to misuse other' credit cards, names and probably ruining the history of the credit card. Data breaches can some-times be conducive to extortion if the targeted individuals are members of a suspicious web-site or have information, they wish to keep private.

Below some of the popular security vulnerabilities in traditional applications are introduced, namely Cross Site Scripting (XSS), SQL injection and insufficient auditing in database.

2.4.1 Cross Site Scripting

XSS attacks (Cross Site Scripting) is defined as a virulent form of injection in which pernicious scripts are injected into reliable and benign website which even could be safeguarded by a trusty firewall and encrypts all the communication between the client and the server.

Cross Site Scripting has turned into one of the most widespread security vulnerability of numerous web sites and web applications. (Galán, E., et el, 2010) and the biggest menace on web applications today. (Bisht, P. and Venkata krishnan, V.N., 2008) (Parvez, M., et el., 2015)

XSS attacks are also used by cybercriminals to imitative actual websites and deceive individuals into providing confidential data. Cross-site Scripting starts when a cybercriminal detects and takes advantage of potential security vulnerabilities in a specific web application and injects and executes pernicious code into it, which in turn sends that malicious code to another other end user. It can simply happen because of erroneous, inaccurate or lack of sanitization of user inputs, which in turn ends up in plentiful issues for web application users and server applications. (Hydara, I., Sultan, A.B.M., et el, 2015)

An efficacious way that helps in preventing XSS attacks on the server side, is to validate or filtere each and every place where user- input fields are shown. (Cook, S. (2003) (Bisht, P. and Venkatakrishnan, V.N., 2008)

Following is a very simple JavaScript snippet that demonstrates an XSS attack, that aims to purloin the cookies maintained on the victim's computer.

```
<script>window.location='https://myownwebsite.net/?cookie='+doc
ment.cookie</script>
```

A cookie is data maintained in either an individual file or multiple small-sized ones depending on the web browser used while visiting a web site.

Reflected XSS, Persistent XSS and DOM-based XSS are the three most common types of cross-site scripting attacks. (Gupta, S. and Gupta, B.B., 2016)

- Reflected XSS is vastly labelled as the most recurrent sort of XSS attack so far detected in the wilderness. Non-persistent XSS, reflected XSS or type 1 XSS all

refer to the same thing which is a type of attack in which malicious script code is reflected off a specific web server to the targeted victim's web browser.

The attack begins when a hacker injects a pernicious script code as a parameter in-to an URL and dispatches it to a particular victim or even to a set of them.

A Reflected XSS can look like this:

<script>window.location='http://www.aboualywebpage.com/?victimcookie='+document.cookie</script>.

The URL is usually sent to the victim in various ways, however sending it via the e-mail is the most common way, and once the targeted person decides to pursue that infected URL to the web site, the page will be loaded in the web browser which allows the attacker to gather all maintained cookies on the client's computer by the client's browser for any victim.

This type of XSS attack happens when the web server does not appropriately scour the output server to a visiting client' computer. (Sharma, P., et el., 2012)

- Sorted or persistent XSS, takes place when a sly script code is inserted directly into a vulnerable web site or web application with a view to being maintained in that webpage on a long-term basis, and therefore, all the website visitors will be automatically infected. Out of all the other types of XSS attacks, Persistent XSS is the most threatening one (Wang, Y., et el, 2011), since it is more dangerous than the other forms of XSS attacks (Chen, J.F., et el., 2012)
- DOM-based XSS also known as type 0 XSS, as the name suggests, is a type of XXS attack that mainly targets manipulating the DOM (stands for document object model), environment on client-side in lieu of dispatching the malicious script to web server, and therefore the server has no chance to verify the payload. (Baranwal, A.K., 2012)

  DOM based XSS attack takes place when a client provides malicious data in JavaScript with the help of a number of methods such as document.write, eval(), etc. (Nagar, N., et el, 2016)

⇒ Non-persistent XSS, persistent XSS and DOM-based XSS all are different techniques for getting a malicious script code to show up on a vulnerable site.

⇒ These three types of attacks differ from each other in the way they are carried out and warded off.

2.4.2 SQL Injection

SQL Injection is a very common technique to attack a website, that actually should have been scotched and should not work any longer but unfortunately it still does.

SQL (Structured Query Language) is the standard programming language for creating, manipulating, and querying data maintained in relational databases.

In that connection, and for facilitating the process of data manipulation (DM), SQL comes with a set of commands, namely, CREATE DATABASE, CREATE TABLE, DELETE, ALTER DATABASE, INSERT INTO, DELETE, UPDATE and SELECT

All the aforementioned SQL commands can be illicitly misused to manipulate data stored in a database using a type of mechanism that attacks web-based applications called SQL Injection

SQL injection attacks constitutes one of the most critical threats to web apps, as they enable cybercriminals to get absolute access to the databases underlying the applications

and to probably the critical data these databases include. (Halfond, W.G., et el, 2006)

This type of attack takes place when some unreliable content is banded together with SQL code prior the SQL code is made ready.

That mischievous content is not necessarily derived from a user but could be also mistakenly generated, while it turns into malicious when an attacker dispatches some values to an application with a view to infixing it into a SQL string, which provides him/her with the means to rig the output of queries, perusing data or even illicitly mutating the maintained data.

As an example: imagine a webpage that lets a customer to input Customer ID and gets back user data.

So, if a customer types the 'CustomerID' as 10 or 1=1, the SQL would get interpreted to

```
SELECT * FROM Customers WHERE CustomerID = 10 or 1=1.
```

The aforementioned SQL query would fetch all rows from the Customers table where 1=1 is true.

Cybercriminals use various methods in order to run an SQL injection attack. However, Blind SQL Injection, Error Based SQL Injection and Union-Based SQL Injection are the most frequently used ones.

- Blind SQL injection is a type of SQL injection attack wherein a cybercriminal attempts to do guesswork using query string.
- Error Based SQL Injection is described as a mechanism in which a hacker defines potential system vulnerabilities by intentionally bringing about a SQL database to extrude errors onto the UI (stands for user interface)
- Union-Based SQL Injection as the name implies, it is a type of SQL Injection that relies on either the operator UNION or UNION ALL with a view to extending the outputs reverted by the original query.

There are several techniques that are used in traditional Web applications to avoid and deter SQL Injection vulnerabilities, for instance,

- Averting creating SQL queries through user input by maintaining all the dynamic input values detached from SQL queries.
- Detecting tries at SQL injection instantly by keeping an eye on query logs
- Escaping all the dynamic inputs before inserting them into any SQL string.

2.4.3 Insufficient auditing in database

An audit trail is fairly defined as a security-pertinent logbook, that gives graphical demonstration of the concatenation of processes which have affected at any time a particular step, action, or procedure. ("Audit trail," Wikipedia,)

Due to the lack of having a scanty or at least an efficient auditing framework, the system might experience or end up with heavy losses with respect to data stored in the database by system vulnerabilities profiteering cyber criminals, and therefore, the importance of audit tracking brings out as a preventive of cyber-attacks.

2.5 Mitigating security vulnerabilities

ReactJS and Blockchain can be an effective solution to tackle SS attacks, SQL injection attacks and insufficient auditing in traditional database.

2.5.1 React:

ReactJS automatically escapes variables, which in turn, plays an important role in warding off cross-site scripting injection through string HTML with pernicious JavaScript code. Here is a user input example form ReactJS

```
constructor(props){
    super(props);
    this.state =  {
        user : {
            id: "",
            ownerFullName: "",
            carMake:"",
            carModel: "",
            extraDes:"",
            photos:[],
        },

        errors:{
            ownerFullNameError: "",
            carMakeError: "",
            carModelError: "",
            extraDesError: "",
            photosError: "",
    }}}

        const { classes } = this.props;
        const {ownerFullName} = this.state.user;
        const {carModel} = this.state.user;
        const {carMake} = this.state.user;
        const {extraDes} = this.state.user;

render() {
    return (
        <div>{ ownerFullName }</div>
    );
```

```
}
```
This is simple code snippet from my project which represents a form that receives user input. The variable "`ownerFullName`" was placed in the render function to demonstrate that whatever value that variable holds it will be automatically escaped and presented as a string.

ReactJS guarantees that using markup can be carried out through the render functions. In other words, With ReactJS it became a way harder for markup to be let in, unless the elements were coded by the application developer himself/herself in the render function.

On the other hand, ReactJS has some disadvantages with respect to XSS attacks, for instance a malicious script can be passed and executed via some properties that are provides by React for instance, `<iframe src="{ } />, <img src={ } />, dangerouslySetInnerHTML etc.`

2.5.2 The security issues that are tackled with Blockchain technology

Blockchain technology guarantees to its users a large number of qualities that differentiates itself from different dominant traditional types of databases obtainable in the market.

Here are the most eminent of them considering PoW-based BCs:

- High level of security, assurance and integrity: Blockchain is simply a chain of blocks maintained on the hard drives of a large number of miners distributed all over the world. Each block of the blockchain has a cryptographic hash of current block as well as the previous block in the blockchain.
- All the data stored on block-chain is publicly available and can easily be tracked by any one on the network. Nothing can be approved without a consensus by the majority of the network participants.
- Blockchain technology provides a high level of diaphaneity, since it is accessible by any individual with relative ease.
- Blockchain technology annihilates a lot of unduly labored in operational activities by automating several of them.
- Blockchain technology is not perishable, for the reason that the data stored in the network can be only be modified or added by mutual consensus.

- Blockchain technology is classified as an unbackable, in other words, a cyber-criminal would need to overcome the network by at least 51% which is not possible anyhow.

3 DAPP IMPLEMENTATION

I decided to implement a simple Proof of Concept decentralized application for a car gallery using ReactJS, Truffle framework, web3js, MetaMask, Solidity, and Ganache. The DApp allows any user to add his/her full name, car make, car model, additional in-formational as well as unlimited number of images of the car. The user can afterwards edit, update or delete his profile (Figure 4).

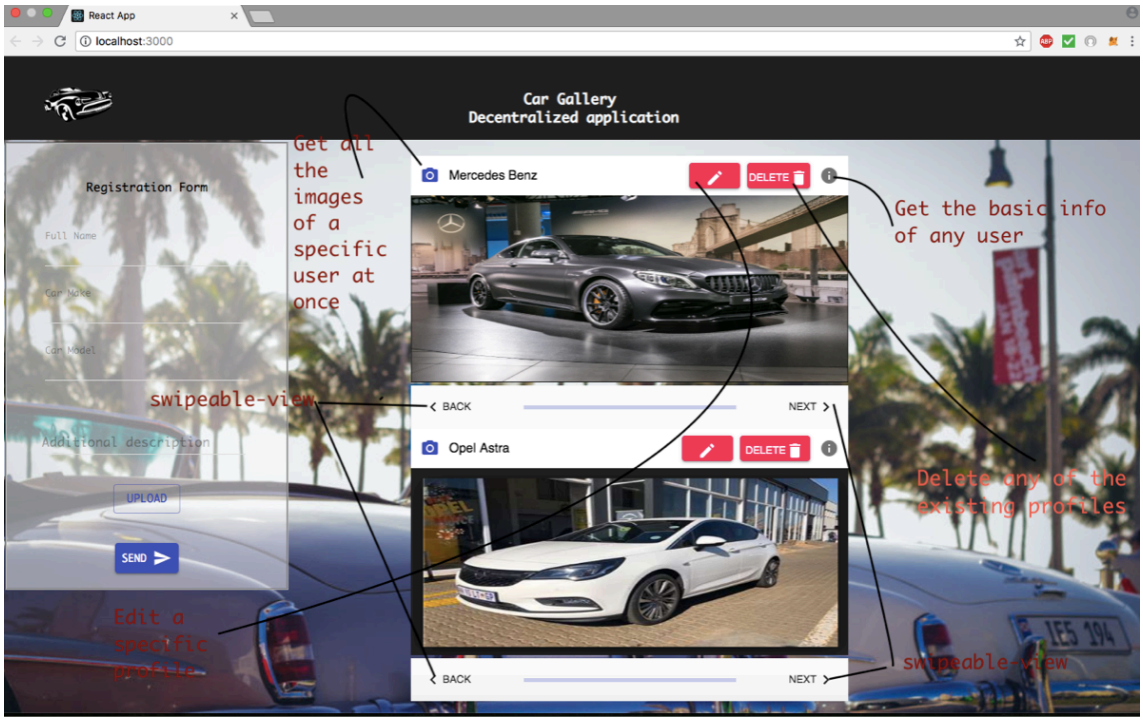


Figure 4 A screenshot of my DApp (client-side-ReactJs)

3.1 Web3.js

I have followed the following steps in order to get web3 installed, configured and integrated with my application. (Manoj P R, 2018)

1. install web3 using `npm install web3` via command line/terminal
2. import web3 into the application

```
import * as Web3 from "web3";
```

3. initialize the web3.js object using the provider object
```
web3 = new Web3(web3.currentProvider);
```

1. Set a provider (HttpProvider) for the instance of web3 in order to access the Ethereum local node
```
web3 =new Web3(new Web3.providers.HttpProvider('http://lo-
calhost:7545'));
```
2. Verify the running providers to confirm that no overwriting, for the running MetaMask provider, occurs
```
if (typeof web3 !== "undefined") {

                web3 = new Web3(web3.currentProvider);

               console.log("Web3 is detected ");
                            resolve(web3);


                          } else {
                            web3 =new Web3(new Web3.provid-
          ers.HttpProvider('http://localhost:7545'));
                            console.log("Cannot find web3");
                            resolve(web3);
                          }
```

3.2 Truffle

Truffle can be installed using the following command via command line/terminal

```
npm install -g truffle
```

(truffleframework.com, 2019)

3.3 Ganache

Ganache GUI can easily be installed, launched and used by non-technical individuals. When starting Ganache GUI for the first time after installation a click on its icon is enough to get launched.

In order to get a smart contract deployed on Ganache, the following two commands are needed:

```
truffle compile
truffle migrate
```

The common way to execute queries to a deployed Ethereum contract is through the Ethereum JavaScript API (web3.js library)

```
import MyCustomerContract from '../build/contracts/CustomerContract';
import initWeb3 from "./initWeb3";
import truffleContract from "truffle-contract";
const contract = (async ()=> {
    const web3 = await initWeb3;
    const Contract = truffleContract(MyCustomerContract);
    Contract.setProvider(web3.currentProvider);
    const instance = await Contract.deployed();

    return instance;

})();
export default contract;
```

3.4 Smart contract:

I coded a smart contract that allows the DApp users to add their names and their car specifications, with the ability to update or delete their profiles afterwards. The smart contract was written using solidity programming language.

Here are the main properties of each customer

```
struct Customer {

        uint256 id;
        string ownerFullName;
        string carMake;
        string carModel;
        string extraDe;
        string[] photos;
        bool exist;

    }
```

Here are two functions, while the first is meant to add photos at the time of creating a new profile, the second is to update the profile with some other photos.

```
// adding some photos while creating the progile
function addCustomerPhotos(uint256 index, string carImage) public { cus-
tomers[index].photos.push(carImage);}
   // updating an image by index
function updateCustomerPhotos(uint256 index, uint256 _index, string
_carImg) public { customers[index].photos[_index] = _carImg;}
```

Here is a function that allows updating customer data

```
//Updating customers by index
function updateCustomer(uint256 index, string _fullName,string _car-
Make,string _carModel,string _extraDe) public {
        // A shorter way to update a customer
       // Customer memory  updatedCustomer = Customer(index, _fullName,
_carMake, _carModel, _extraDe, false);
        if(!isCustomer(index)) revert();
        customers[index].ownerFullName = _fullName;
        customers[index].carMake = _carMake;
        customers[index].carModel = _carModel;
        customers[index].extraDe = _extraDe;
        customers[index].exist = false;

    }
```

And finally, a function that makes it possible to delete a specific customer, actually it doesn't delete anything since it is not possible to delete anything from Blockchain, but it resets the customer values to default values.

```
function deleteCustomer(uint256 index) public returns(bool success) {
        if(!isCustomer(index)) revert();
        delete customers[index].photos;
        delete customers[index];
        delete customerIndex[index];
        //customerIndex.length--;
        return true;
    }
```

4 RESULT

There are a great number of Web application vulnerabilities. However, cross-site scripting (XSS) and SQL injection are categorized as the biggest threat to Web applications for the time being due to their popularity and seriousness. The table below gives information on the main types of XSS attacks, native JavaScript is vulnerable to, and on the other hand, analyzes successes and failures in dealing with XSS attacks in ReactJS (Table 5).

Table 5 Analysis of XSS attacks in native JavaScript and ReactJS

	JavaScript	ReactJS
XSS attack	Although, XSS attacks are potential in several scripting languages they are most popular	ReactJS seems to be secure against most of XSS attacks, thanks to, the way it generates DOM nodes and its ability to automatically escape passed string values via user input. **Main drawbacks:**

	in JS. XSS attacks fall into three types: • **Stored XSS**, where the pernicious script ensues from the infected server's database. • **Non-persistent XSS**, where the pernicious script ensues from the victim's request on the client-side. • **Type-0 XSS**, where vulnerability mainly exists on the client-side.	• Data passed into props, is not automatically escaped before it gets rendered into React DOM. • There are many properties available in ReactJS that still allow XSS attacks, including, but not limited to, `<a href="{}" />` and `<img src={ } />`. Accordingly, a classical cross-site scripting attack using a URL that contains a malicious script, like `<a href="javascript:alert(1)">Button</a>` will still work in ReactJS, as follows `ReactDOM.render(<a href="JavaScript:alert(1)">Button</a>,document.getElementById('root'));` Once the button is clicked, the embedded script into the href attribute will get executed

There are many methods that are used by cybercriminals to attack a Web application. However, there is one, known as SQL injection, that is specially achieved by targeting the backend database by executing sly SQL statements with a view to getting the application to unpredictably behave for the benefit of the cybercriminal.

On the other hand, there is another database vulnerability that is considered as a severe threat to Web applications, known as, insufficient auditing. Auditing is the observing and registering of chosen user database actions. Insufficient auditing is one of the security vulnerabilities that deeply bother Web applications while this security vulnerability was efficiently overcome in DApps using consensus mechanisms, e.g. proof-of-stake (PoS) or proof-of-work (PoW).

In Proof-of-work (PoW) the network participants use supercomputers, which in turn, use a lot of power to do some work that is considered to be mathematically intense but, on the other hand, very easy for the other network participants to verify. Taking Bitcoin as

an example where a large number of miners across the world vie to decode intricate mathematical puzzles created from sets of transactions, recognized as blocks, each of which is verified before it gets added to the network. Bitcoin miners are first required to find out the "nonce", a random 32-bit number used to commence creating a hash.

As an alternative and to tackle the ingrained problems in proof-of-work, e.g. the high amount of electricity required for implementing proof-of-work, proof-of-stake, another consensus algorithm, is used to achieve distributed consensus.

The table below summarizes the difference between SQL Injection attacks on Web applications and decentralized applications and also talks about insufficient auditing vulnerability in traditional databases and how this issue was tackled in Blockchain (Table 6).

Table 6 Analysis of security vulnerabilities in both traditional databases and Blockchain

	Traditional Databases	Blockchain
SQL-injection	SQL injection, as the name suggests, is a type of attack that mainly targets Web-based application's database server with pernicious SQL statements. In this types of attack, malicious SQL statements are injected and executed with a view to • Accessing and manipulating data • Closing down the My SQL server • Destroying a database. Databases that deals with SQL, including, but not limited to, MySQL, MS SQL Server, IBM DB2, Oracle and Microsoft Access are definitely much more	There might be some claims that SQL can be used with Blockchain and therefor it is subject to SQL injection attacks. However, here are some facts about blockchain that make SQL injection attacks ineffective. • Blockchain is public digitized ledger managed by a massive number of computers distributed around the world which makes it impossible for any attack to occur on Blockchain network without that getting observed during a very short period of time. • Blockchain relies on a very essential concept which is called "Proof of work or pow for short". Proof of work mainly guarantees that a cybercrime is not eligible to alter the former disbursed coins and transactions without cracking the current and preceding computational power applied since the beginning of

	vulnerable to SQL injection attack compared with Non-SQL databases that don't need to worry about such vulnerability. The reason behind this security vulnerability is due to calls made from the server-side to the database and not the database itself.	hashing the Blockchain until the actual block he chooses to hit the chain on and therefore neither data maintained in each block can be modified nor a block can be deleted without affecting and changing subsequent blocks.
Insufficient auditing	Inadequate auditing in traditional databases can have disastrous results on individuals and a considerable number of businesses.	• A change that takes place on the blockchain network can easily be tracked, and for a change to occur, consensus with overwhelming majority of networks participants must be reached. • Blockchain provides a high level of transparency, while the identity of each client is hidden using powerful cryptography, all data stored are accessible to the public.

5 DISCUSSION

In this thesis I have examined the use of ReactJS and DApps for mitigating the inherent existence of common security vulnerabilities in traditional web apps. The research questions posed for this thesis aimed at elaborating the aim. Below follows a discussion regarding each research question:

5.1 The main components of centralized & decentralized applications.

Almost every Web application existing on the internet comprises of three fundamental components, namely a client, a server and a database server.

Decentralized applications can use the same technologies as Web applications for developing their client-side. In other words, any scripting language that makes use of API calls to its server side can be used for building the client-side of any decentralized application.

However, the server-side in decentralized applications differs from its counterpart in web application and that's because, well, the main idea behind decentralized applications is to be totally out of central control.

Decentralized applications use self-executing smart contracts stored on distributed ledgers known as blockchains.

5.2 The main difference between the server-side in traditional Web applications and its counterpart in decentralized applications.

Web-based applications have mainly central servers while decentralized applications run on P2P network of a great number of computers distributed all over the world.

There are a large number of programming languages that can be used to program the server-side in traditional Web applications, while in DApps some traditional server-side programming languages, e.g. Java, Python and C++ and a wide range of blockchain programming language, e.g. Solidity and Simplicity can be used as well to write smart contracts

5.3 The way the client-side interact with the server in both the traditional Web application and the decentralized application cases and the most common techniques used to achieve the interaction.

A decentralized application has decentralized nodes (servers) arranged into P2P network, and for the client-side code to communicate with the server-side code, a library like web3.js is used whilst in the case of web applications, a client interacts a server using HTTP.

5.4 The three most common security vulnerabilities in Web applications and the way they are handled in decentralized applications

Cross Site Scripting (XSS), SQL injection and insufficient auditing in database are considered to be most common security vulnerabilities in Web applications.

ReactJS and Blockchain can be an effective solution to tackle those three security vulnerabilities.

5.4.1 ReactJS

ReactJS is a splendid library that makes it possible for developers to build elaborate and interactive user interfaces (UIs). One of the strengths of ReactJS is that it is supported by Facebook and a huge community of experienced developers.

ReactJS is a client-side library that handles the view layer and neither deal with data storage on the server nor have any functions in the core code concerning SQL, and therefore ReactJS does not provide any security with regards to SQL injection, since it is the server-side processes that are required to take care of SQL injection,

ReactJS is reasonably secure against most XSS attacks and has gone a long way in that direction, so a traditional attack will not simply work with ReactJS. However, there are still some other ways to get around this to get a malicious code injected and executed into a reliable Web-page that uses ReactJS on the client-side.

5.4.2 Blockchain

On the other hand, decentralized applications are one of the most essential inventions in the recent years for its significant influence on a vast number of individuals and their businesses. Decentralized applications and blockchain seem to have a lot of uses for the time being and in the future.

Generally speaking – security with regards to decentralized applications is seen to be significantly attained by exposure instead of by mystery- all the data maintained in blockchain is overtly recognized and therefore, it is extremely hard for even a small piece of data to be tampered without that getting observed by someone on the network.

6 CONCLUSION

Both ReactJS and decentralized applications are considered to be immune to a large number of widespread cyber security attacks thanks to the key technologies underlying both of them.

However, nothing is totally secure. ReactJS perfectly escapes html to string which can be seen as a pretty big endeavor in an attempt to avert cross-site scripting but there are still other props (properties) that let XSS attacks to get through.

On the other hand, Blockchain technology is based on cryptography. The implementation of blockchain seems to be steely but it cannot be securer than the cryptographic algorithms it leans to and their implementation.

Throughout history there have been many potent shatterproof algorithms been broken, and many faulty executions of crypto-code have been experienced.

ReactJS and decentralized applications are substantially secure but there are and always will be attempts to get them both subjugated and surrendered.

REFERENCES

DANNEN, C. (2017). Introducing Ethereum and Solidity: foundations of cryptocurrency and blockchain programming for beginners.

Retrieved March 13, 2019, from https://friendlyuser.github.io/project/ipfs-dapp/featured.png

CROCKFORD, D. (2008). Javascript: the good parts. Beijing, O'Reilly.

HAVERBEKE, M. (2015). Eloquent JavaScript: a modern introduction to programming

IPFS. Retrieved April 23, 2019, from https://ipfs.io/

Retrieved April 23, 2019, from https://metamask.io/

Web3js. Retrieved April 23, 2019, from https://web3js.readthedocs.io/en/1.0/

Introduction to web APIs. Retrieved April 23, 2019, from https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Introduction

Baranwal, A.K., 2012. Approaches to detect SQL injection and XSS in web applications. EECE 571b, Term Survey paper.

Cook, S. (2003). A Web developers guide to cross-site scripting.

Berners-Lee, T. (1989). Tim berners-lee.

Nagar, N. and Suman, U., 2016. Analyzing virtualization vulnerabilities and design a secure cloud environment to prevent from XSS attack. International Journal of Cloud Applications and Computing (IJCAC), 6(1), pp.1-14.

Mohamed, M.A., Altrafi, O.G. and Ismail, M.O., 2014. Relational vs. nosql databases: A survey. International Journal of Computer and Information Technology, 3(03), pp.598-601.

Hider, P., 2004. The bibliographic advantages of a centralised union catalogue for ILL and resource sharing. Interlending & Document Supply, 32(1), pp.17-29.

Árnason, V., 2004. Coding and consent: moral challenges of the database project in Iceland. Bioethics, 18(1), pp.27-49.

A brief history of CSS until 2016. 17 December 2016, Bert Bos, Style Activity Lead | Colophon https://www.w3.org/Style/CSS20/history.html

Wikipedia contributors. (2019, January 2). Cascading Style Sheets. In Wikipedia, The Free Encyclopedia. Retrieved 01:56, January 3, 2019, from https://en.wikipedia.org/w/index.php?title=Cascading_Style_Sheets&oldid=876431447

Pilgrim, M., 2010. HTML5: up and running: dive into the future of web development. " O'Reilly Media, Inc.".

Wikipedia contributors. (2018, December 23). HTML. In Wikipedia, The Free Encyclopedia. Retrieved 21:22, December 28, 2018, from https://en.wikipedia.org/w/index.php?title=HTML&oldid=874990638

Wikipedia contributors. (2018, December 19). History of the web browser. In Wikipedia, The Free Encyclopedia. Retrieved 21:36, December 28, 2018, from https://en.wikipedia.org/w/index.php?title=History_of_the_web_browser&oldid=874501618

Kurilova, D., 2012. The Beautiful World Wide Web or the Current State of CSS.

FENIX.CASH "What is the difference between smart contracts and dapps?" Quora. https://www.quora.com/What-is-the-difference-between-smart-contracts-and-dapps

Khatwani,Sudhir. "What are DApps (Decentralized Applications)? – The Beginner's Guide". https://coinsutra.com/dapps-decentralized-applications/ . 2/02/2018

Snyder, A., 1986, June. Encapsulation and inheritance in object-oriented programming languages. In ACM Sigplan Notices (Vol. 21, No. 11, pp. 38-45). ACM.

"What is the technical history of web development?".Quora.https://www.quora.com/What-is-the-technical-history-of-web-development Nov 28 2017 "What Are Dapps? The New Decentralized Future" https://blockgeeks.com/guides/dapps/.

Rajivjc"Let's get started with your first Ethereum DApp!". Medium..https://medium.com/@rajiv.cheriyan/lets-get-started-with-your-first-ethereum-dapp-f09feb59dd78 . Jun 8, 2017

Materzok, M., 2016. Certified Desugaring of Javascript Programs using Coq.

Wium Lie, H., 2017, March. CSS and User-Adapted Web Presentations. In Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval (pp. 5-5). ACM.

Luu, L., Chu, D.H., Olickel, H., Saxena, P. and Hobor, A., 2016, October. Making smart contracts smarter. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 254-269). ACM.

Halfond, W.G., Viegas, J. and Orso, A., 2006, March. A classification of SQL-injection attacks and countermeasures. In Proceedings of the IEEE International Symposium on Secure Software Engineering (Vol. 1, pp. 13-15). IEEE.

Wikipedia contributors. "HTML5." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 12 Jan. 2019. Web. 17 Jan. 2019.

Prusty, N., 2017. Building Blockchain Projects. Packt Publishing Ltd.

Sharma, P., Johari, R. and Sarma, S.S., 2012. Integrated approach to prevent SQL injection attack and reflected cross site scripting attack. International Journal of System Assurance Engineering and Management, 3(4), pp.343-351.

Thiemann, P., 2005, April. Towards a type system for analyzing javascript programs. In European Symposium On Programming (pp. 408-422). Springer, Berlin, Heidelberg.

Manoj P R, 2018. Ethereum Cookbook: Over 100 recipes covering Ethereum-based tokens, games, wallets, smart contracts, protocols, and Dapps. Packt Publishing Ltd.

Carlozo, L., 2017. What is blockchain?. Journal of Accountancy, 224(1), p.29.

Lie, H.W., 2000, September. Abstraction levels in Web document formats. In International Workshop on Principles of Digital Document Processing (pp. 120-127). Springer, Berlin, Heidelberg.

Galán, E., Alcaide, A., Orfila, A. and Blasco, J., 2010, November. A multi-agent scanner to detect stored-XSS vulnerabilities. In 2010 International Conference for Internet Technology and Secured Transactions (pp. 1-6). IEEE.

White, B., 1996. The World Wide Web (WWW). In HTML and the Art of Authoring for the World Wide Web (pp. 13-20). Springer, Boston, MA.

Sharma, T.N., Bhardwaj, P. and Bhardwaj, M., 2012. Difference Between HTML and HTML 5. International Journal Of Computational Engineering Research, 2(5), pp.1430-1437.

Cisneros, J.L.B., Aarestrup, F.M. and Lund, O., 2018. Public health surveillance using decentralized technologies. Blockchain in Healthcare Today.

O'Neil, E.J., 2008, June. Object/relational mapping 2008: hibernate and the entity data model (edm). In Proceedings of the 2008 ACM SIGMOD international conference on Management of data (pp. 1351-1356). ACM.

Bos, B., Lie, H.W., Lilley, C. and Jacobs, I., 1998. Cascading style sheets, level 2 CSS2 specification. Available via the World Wide Web at http://www. w3. org/TR/1998/REC-CSS2-19980512, pp.1472-1473.

Buterin, V., 2014. A next-generation smart contract and decentralized application platform. white paper.

Karafiloski, E. and Mishev, A., 2017, July. Blockchain solutions for big data challenges: A literature review. In Smart Technologies, IEEE EUROCON 2017-17th International Conference on (pp. 763-768). IEEE.

Raggett, Dave, A History of HTML, Chapter 2, Addison Wesley Longman, 1998, http://www.w3.org/People/Raggett/book4/ch02.html

Grosskurth, A. and Godfrey, M.W., 2005, September. A reference architecture for web browsers. In null (pp. 661-664). IEEE.

Bratt, S., 2007. Semantic web and other W3C technologies to watch. Talks at W3C, January.

Wang, Y., Li, Z. and Guo, T., 2011, August. Program slicing stored XSS bugs in web application. In 2011 fifth international conference on theoretical aspects of software engineering (pp. 191-194). IEEE.

Gupta, S. and Gupta, B.B., 2016. XSS-SAFE: a server-side approach to detect and mitigate cross-site scripting (XSS) attacks in JavaScript code. Arabian Journal for Science and Engineering, 41(3), pp.897-920.

Retrieved March 3, 2019, from https://reactjs.org/docs/faq-internals.html

Retrieved March 15, 2019, from https://blog.ethereum.org/2014/07/22/launching-the-ether-sale/

Bisht, P. and Venkatakrishnan, V.N., 2008, July. XSS-GUARD: precise dynamic prevention of cross-site scripting attacks. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 23-43). Springer, Berlin, Heidelberg.

Wikipedia contributors, "Audit trail," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Audit_trail&oldid=889033450 (accessed March 26, 2019).

Hydara, I., Sultan, A.B.M., Zulzalil, H. and Admodisastro, N., 2015. Current state of research on cross-site scripting (XSS)–A systematic literature review. Information and Software Technology, 58, pp.170-186.'

Parvez, M., Zavarsky, P. and Khoury, N., 2015, December. Analysis of effectiveness of black-box web application scanners in detection of stored SQL injection and stored XSS vulnerabilities. In 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST) (pp. 186-191). IEEE.

CHEN, J.F., WANG, Y.D., ZHANG, Y.Q. and LIU, Q.X., 2012. Automatic generation of attack vectors for stored-XSS. Journal of Graduate University of Chinese Academy of Sciences, 29(6), p.8151820.

Buterin, V., 2017. White paper (2013).

https://ethereum.stackexchange.com/questions/4522/what-was-the-approximate-cost-of-1-eth-during-pre-launch-aug-2014/4526

Leprohon, M.A., 2017. ECMAScript 6 and the evolution of JavaScript: A deeper look into the language's new features.

Boyd, S.W. and Keromytis, A.D., 2004, June. SQLrand: Preventing SQL injection attacks. In International Conference on Applied Cryptography and Network Security (pp. 292-302). Springer, Berlin, Heidelberg.

Westerlund, M. and Kratzke, N., 2018, July. Towards Distributed Clouds: A Review About the Evolution of Centralized Cloud Computing, Distributed Ledger Technologies, and A Foresight on Unifying Opportunities and Security Implications. In 2018 International Conference on High Performance Computing & Simulation (HPCS) (pp. 655-663). IEEE.

Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system.

Wikipedia contributors, "JavaScript," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=JavaScript&oldid=892991120 (accessed April 23, 2019).

Wikipedia contributors. (2019, March 13). HTML. In Wikipedia, The Free Encyclopedia. Retrieved 09:49, April 23, 2019, from https://en.wikipedia.org/w/index.php?title=HTML&oldid=887642352

Wikipedia contributors. (2019, April 20). Cascading Style Sheets. In Wikipedia, The Free Encyclopedia. Retrieved 09:50, April 23, 2019, from https://en.wikipedia.org/w/index.php?title=Cascading_Style_Sheets&oldid=893264882

Wikipedia contributors. (2019, March 30). InterPlanetary File System. In Wikipedia, The Free Encyclopedia. Retrieved 09:50, April 23, 2019, from https://en.wikipedia.org/w/index.php?title=InterPlanetary_File_System&oldid=890112808

Wikipedia contributors. (2019, April 10). Ethereum. In Wikipedia, The Free Encyclopedia. Retrieved 09:51, April 23, 2019, from https://en.wikipedia.org/w/index.php?title=Ethereum&oldid=891804112

Wikipedia contributors. (2019, April 21). Blockchain. In Wikipedia, The Free Encyclopedia. Retrieved 09:51, April 23, 2019, from https://en.wikipedia.org/w/index.php?title=Blockchain&oldid=893385950

Ganache. Retrieved April 23, 2019, from https://truffleframework.com/docs/ganache/quickstart

Solidity. Retrieved 09:51, April 23, 2019, from https://solidity.readthedocs.io/en/v0.4.24/

What is Swarm. Retrieved April 23, 2019, from https://swarm-guide.readthedocs.io/en/latest/