Investigating Vulnerabilities in Smart Homes: Access Control Through Blockchain CIS 433 Project Final Report

Mateo Minato

March 15, 2020

1 Abstract

As privacy becomes an increasingly mainstream issue, and the Internet of Things grows ever more prevalent, the effectiveness of our current security methods have been called more and more into question. Within the last few years, researchers have found vulnerabilities within these in-home networks that could compromise a users electronic privacy or even physical space. Particularly, researchers observed that access control practices in smart devices were inconsistent and largely inadequate. In this paper, I investigate the efficacy of implementing access control in smart homes through the use of blockchain. Specifically, I use a local Ethereum test network and smart contracts to prototype a smart home access control system as a proof of concept of blockchain based access control. Finally, I discuss my assumptions, limitations, and this projects potential for future expansion.

1.1 Keywords

ethereum; access control; security; privacy; blockchain; solidity

2 Introduction

Homes are becoming more and more smart enabled, with appliances like electric kettles, thermostats, door locks, security cameras, and even overhead lighting being network enabled. Many of these "smart homes" also implement a hub of some sort, such as a Google Home. How easy is it to gain access to network enabled IoT devices and appliances within a users home? Can these security vulnerabilities be used to access a physical location, intercept other forms of electronic communication, or to invade a users privacy? While these questions may seem hypothetical, they have the potentially to dramatically affect the lives of millions. This project will explore potential vulnerabilities with smart devices

in users homes and potential consequences of the exploitation of these vulnerabilities. It will also seek to generate a potential solution for these vulnerabilities.

3 Related Work & Research

This prototype project drew from several sources listed in the references at the end of this report. Most notably, a 2017 research paper from Maesa, Damiano & Mori, Paolo & Ricci, Laura at the University of Pisa entitled "*Blockchain Based Access Control*", and a 2017 research paper from Ouaddah, Aafaf & Elkalam, Anas & Ouahman, Abdellah at the Oscars Laboratory in Marrakesh, Cadi Ayyad University entitled "*FairAccess: a new Blockchain-based access control framework for the Internet of Things: FairAccess: a new access control framework for IoT. Security and Communication Networks*?". Both provided a strong base of understanding regarding the theory of access control through the use of blockchain, and helped influence the design of this project.

I also heavily referenced the Solidity and Ethereum home websites and accompanying documentation for information on how to write and use smart contracts. I reference two online guides on medium.com for information on building, deploying, and interacting with my local ethereum test network using truffle and ganache-cli. All of this related work is included in the references section at the end of this paper.

Prior to beginning design and implementation, it was imperative that research was conducted to determine a number of facts. First, I needed to determine the potential vulnerabilities in current implementations of access control in smart home systems. Then, I needed to determine the efficacy of implementing access control in the blockchain, and determine how to approach the design to address these vulnerabilities. Moreover, there exist a number of popular blockchain platforms, and research was necessary to choose the correct one to use and learn how to use it.

3.1 Vulnerability Research

First, I took a look at known vulnerabilities in smart home devices. I found that a number of attacks have been identified targeting smart homes, a few of which are enumerated below. Importantly, most or all of these attacks rely on exploiting the access control policies of these smart homes, which are frequently managed by a central hub (e.g., Google Home) and can lead to a single point of failure.

- **Lateral Privilege Escalation Attacks**: Some smart devices (such as bluetooth or WiFi enabled door locks, garage door controllers, or anything else that manages access to a physical location) appear secure on their own. Other devices, such as smart bulbs, thermometers, or other appliances apparently require less standalone security. In some cases, hackers

were able to gain access to low-level devices and modify information on higher security devices. This is by far the largest set of attacks, and most of the attacks I researched fall under the umbrella of escalation attacks, often taking advantage of third party apps or irresponsible use of global variables.

- **Lack of System Defenses**: Similarly, some smart home systems (ex. Nest) do not employ transitive access control enforcement. Hackers were able to trick smart home systems into granting them higher level access than desirable and execute malicious routines.

- **Lack of Bare Minimum Protections**: Through lack of any real access control, hackers were able to gain full access to the Hue smart home system, arbitrarily adding third party applications (Trojans) as trusted entities, installing malicious add-ons.

3.2 Solutions Research

Although the security issues involved in smart home access control are complicated, my research suggested that they are often the result of centralized data stores and the allowance for a single point of failure. In the interest of addressing this access control, I postulated (with advice and guidance from notes by graduate students Devkishen Sisodia and Samuel Mergendahl) that a potential solution could lie in embedding access control in a Blockchain network.

After gaining an understanding of the types of vulnerabilities and exploits commonly present in smart home IoT systems, and on the advice of Samuel Mergendahl, I decided to audit various blockchain systems to determine which (if any) would be the most appropriate choice for embedded access control.

- **Bitcoin**: Obviously the original and most famous of the blockchain platforms, bitcoin provides some serious upside as a potential for access control implementation. Primarily, it is extraordinarily well-documented and understood, and is the basis for "*Blockchain Based Access Control*", one of the primary papers serving as a reference for my project. With that said, bitcoin has no native support for direct peer-to-ledger transactions, so despite its numerous benefits, it may turn out to be rather difficult to implement something as automated as access control on the bitcoin blockchain.

- **Ethereum**: Ethereum is the second largest blockchain platform, and provides substantial benefit over bitcoin in the form of smart contracts. Smart contracts are essentially self-executing code blocks that trigger based on an event (e.g., expiration date). They serve as a form of vending machine, allowing peer-to-ledger transactions to occur natively, quickly, and with minimal complexity. In the context of access control, smart contracts could dramatically reduce the complexity of my project, and potentially

lend themselves to a more elegant solution than is possible in the bitcoin blockchain platform.

- **Litecoin**: Although less popular or developed than bitcoin, Litecoin provides a notable advantage in terms of speed (4x faster according to blockchain.info). Although this project will serve more as a proof-of-concept than an industry ready solution, in a commercial grade solution, speed in access control would certainly become a real and important factor. With that said, I am not sure the benefit of speed improvement in the Litecoin network outweighs its biggest downside; that it just is not as popular as bitcoin.

- **FairAccess**: Unlike the three previous blockchain platforms, FairAccess is neither open source nor publicly available in any real form. It was a proof-of-concept form of blockchain based access control developed by The University of Pisa, and is described at length in "*FairAccess: a new Blockchain-based access control framework for the Internet of Things: FairAccess: a new access control framework for IoT. Security and Communication Networks*". Although I would be unable to use and expand upon FairAccess directly, it outlines portions of the project that may have otherwise proved too elusive or complex for a project of this scale.

After examining these blockchain networks, I determined the most beneficial solution would be an implementation using the Ethereum smart contracts but heavily referencing solutions present in the described implementation of FairAccess.

4 Practical

4.1 Design

The architecture of the access control system itself should be familiar; it's reminiscent of any authority based authentication/authorization protocol, except implemented through the use of smart contracts and the Ethereum blockchain. On one side of the network, there exists an edge user, the owner of the smart home, the primary user. Additional users can be added here by the initial user, but for the sake of simplicity, let's imagine we only have the one user. On the other side of the network sit a series of edge devices. For the prototype modeled here, I imagined three types of devices; a lock, a light, and a thermostat. Of course, this is arbitrary. The user can add as many edge devices as they want, including adding multiple distinct devices of the same type, or adding support for new device types.

Between the edge user and devices sits a central authority. The authority, implemented with the user of a smart contract, authenticates users using the standard Ethereum public-private key system. It maintains authorization through the user of a simple map, storing users permission information and allowing

these permissions to be modified only by administrators. Only the authority is allowed to communicate with the devices, so all communication must be initialized by the user, sent to the authority, and passed to the device if and only if the authority deems that the access attempt is allowed. This is largely modeled after the smart home systems we see in homes today–access to edge devices is primarily managed through hub devices, such as Google Home or Amazon Alexa. Unlike the conventional design, however, this Ethereum-based implementation prevents access of the system *any other way.* This directly addresses a number of the vulnerability concerns prevalent among smart home systems today.

4.2 Implementation

First, a smart contract exists as an authority model. In theory, it would sit at the hub device and interact directly with any users. In this prototype, it sits as a deployed node on a private Ethereum network modeled locally using truffle. This private network is built with the ganache-cli, which generates 10 local Ethereum wallets each populated with 100 ether. One of these accounts is used to deploy the authority, which marks its creating user as its owner; this initial user being automatically given administrative abilities. This can be thought of as analogous to setting up a hub device–in the real world, a device such as a Google Home would be plugged in, turned on, initialized, and then some (likely mostly automated) setup process would occur through the owners smart phone or computer.

The user can then add to the authority a device. In my prototype, this is modeled by a smart contract interface called Device.sol, and it's three children, Lock.sol, Light.sol, and Thermo.sol. When the user adds a smart lock to the authority, for example, the authority deploys the Lock.sol smart contract between itself and the edge device. This contract enumerates ownership, and implements a series of functions the device is capable of. Since each child contract of Device.sol is required to implement the same functions, I gave them generic names–in a more scalable model, these generic functions would correlate to some named functions through the use of json configuration files, or another, clearer manner that the edge user could read directly. However, for the sake of this prototype, the function names are left quite generic.

For the purpose of this prototype, and as implementation was somewhat limited by the inability to deploy code directly to my smart devices (they are unfortunately not open source), these function implementations are mocked. That is, if an edge user tells "lock1" to "lock the door", the command is passed through the authority to the contract representing "lock1", and a variable called "locked" is set to true. This state can be queried by any authorized user, and it's current value returned. Obviously, in real life, these functions would actually trigger a

device to physically change states, but I thought this model was effective in the absence of hardware.

At the front end, I used a simple javascript and HTML server application to make queries and view state. This application was really one of convenience to avoid perpetuating endless command line queries, but is an effective (if not particularly pretty) model for smart home operations in the real world. Most smart home devices, and smart hubs in particular, are controlled through the use of a smart phone or computer application to modify data, add edge devices, view state, and generally interact with the smart home network.

Finally, it's worth noting that the implementation of the private ethereum blockchain is inherently complex and may vary system to system. In the interest of providing a method for testing this project without necessitating a potentially arduous and buggy setup process, I recommend the use of Remix IDE, the solidity online text editor. There, you can mock and deploy the included smart contracts, and interact with them using several different ether-rich accounts through a number of existing ethereum test networks, which is (excepting the lack of a front-end application) almost as effective a model as one's individual, local test net.

5 Results & Analysis

Even a rudimentary amount of research shows the need for dramatic improvements in our current access control practices in the Internet of Things. Vulnerabilities are numerous, and most appear to be emblematic of inconsistent practice between vendors. Proprietary protocols vary between products, while many products appear to implement little or no access control what-so-ever. Moreover, the key vulnerabilities identified in smart homes were related either to authorization or authentication. It seems highly beneficial, then, to take advantage of technology already equipped, at its baseline, to handle both of these topics.

Although this project was largely prototypical, it was ultimately an effective and successful model. I was able to demonstrate how smart contracts could be used to implement an access control system throughout a smart home. The nature of the blockchain's private-public key system guarantees authentication, and the implementation of the authority smart contract provides the accompanying authorization with a tiered, limited access system.

5.1 Assumptions & Problems

While I consider the developed system largely successful, there remain some unresolved or unaddressed problems that could impede future development.

First, I largely neglected data size, deployment, and cost. Ethereum uses "gas" in the form of "Ether" to measure computational effort. Operations that are conducted in Ethereum require gas to execute. Although I did not conduct any formal analysis on computational cost ("gas cost") for this project, it's possible that this system in its current iteration when deployed could be rather expensive.

Additionally, smart contracts have byte size limits. In fact, in solidity, smart contracts are restricted to 24KB. Inlining the same code multiple times can add to that size. It's possible that in a large home, with many users and devices, the authority smart contract could top out, preventing the addition of more users or devices, or even crashing the system. I believe this could be addressed with further modularity–for example, instead of using generic functions to interact with edge devices, device function signatures could be known through some configuration file (yaml, json) and then passed to the authority directly to reduce the size of the smart contracts.

Finally, it's possible that there are vulnerabilities I haven't thought of. Security and privacy are complex issues, and largely uphill battles. As with any new system, it's possible, even likely that this implementation introduces new, potentially quite subtle vulnerabilities. These may be easily fixable by expanding the feature set, or reworking small portions of the architecture and design. Or, these may be foundational, and disrupt the idea that a blockchain based access control system is feasible at all (particularly, some experts still doubt the security of blockchain as a whole, as it is a newer and less tested technology). In either case, it's clear more testing is needed.

5.2 Expansion

In the future, the most obvious next steps would be to implement this device at the hardware level, using first raspberry pi (or other, home-baked hardware systems) and eventually existing, proprietary smart home devices such as the Google Home or Amazon Alexa. The feature set could be expanded to include better functional references for edge devices. The programs architecture could also be played with–another implementation I considered was totally decentralized, with each device interacting directly with each user, and no need for a central authority.

No matter what direction the project could be taken, far more testing would be necessary before any decisive claims could be made about its real life potential. As with most initial research, we've shown that there is a) a problem here that needs to be addressed, and b) that blockchain based access control has potential as a solution to this problem, but we cannot yet declare it a sure thing.

5.3 Lessons Learned

So far, the biggest lesson I have learned is that thorough and diverse research can be a tremendous aide in guiding a project. When I started this project, I was not totally sure which direction I wanted to take it, I just knew smart home security was a hot topic right now and that I owned a number of smart home devices. Prior to this project, I had virtually no experience working with any blockchain system, and didn't even really understand the fundamental driving technology. If I were to do it all over, I wouldn't change too much about my approach, except perhaps seeking a bit more real life assistance in getting the basics of a new, different technology–while I understand Ethereum and Solidity fairly well now, the learning curve for both was rather steep. Although I started the project on somewhat rocky ground, after investing quite a bit of time in research, and finding and analyzing relevant studies, I found that I not only had a much clearer direction for my project, but that my interest in security as a concept had been piqued substantially. I plan to continue learning more about blockchain and IoT security in the rest of my education, and beyond.

6 References

Kafle, et al. "A Study of Data Store-Based Home Automation."ArXiv.org, 4 Dec. 2018.

Dorri, et al. "Towards an Optimized BlockChain for IoT." acm.org.

Maesa, Damiano & Mori, Paolo & Ricci, Laura. (2017). "Blockchain Based Access Control."

Thwin, Thein & Vasupongayya, Sangsuree. (2019). "Blockchain-Based Access Control Model to Preserve Privacy for Personal Health Record Systems."

Rouhani, Sara, and Ralph Deters. "Blockchain Based Access Control Systems: State of the Art and Challenges." IEEE/WIC/ACM International Conference on Web Intelligence (2019).

Ouaddah, Aafaf & Elkalam, Anas & Ouahman, Abdellah. (2017). "FairAccess: a new Blockchain-based access control framework for the Internet of Things: FairAccess: a new access control framework for IoT." Security and Communication Networks.

Benemerito, Sam. (2019) "Deploying Smart Contracts with Truffle".

Ram, Prashant. (2018) "How To Setup a Private Ethereum Blockchain and Deploy a Solidity Smart Contract on the Blockchain".

A Appendix

A.1 Softwares Used

- **Truffle**: Truffle Suite provides a development environment for the Ethereum Virtual Machine for easy development of smart contracts.
- **Ganache**: A personal blockchain for Ethereum development used to deploy contracts, develop applications, and run tests.
- **Solidity & Remix IDE**: Solidity is the primary language for writing Ethereum smart contracts. Remix is an open source tool for writing and testing Solidity contracts in your browser.
- **npm & homebrew**: Common package managers for Mac OSX.
- **Metamask**: A chrome extension that allows management of local or cloud-based Ethereum accounts and wallets.
- **Geth**: The command line interface for running a full ethereum node.
- **Live-Server**: A small, easy to use development server for HTML/JavaScript/CSS.

A.2 Softwares Developed

The software package developed is a series of smart contracts along with the front-end (JavaScript and HTML) live-server, and the accompanying ethereum private network used for local testing. Because it is, for all intents and purposes, impossible for me to pass over my local ethereum network, I highly recommend testing without the front-end live-server and using Remix IDE for in browser execution of the smart contracts. The accompanying README.txt has detailed instructions on how to use Remix to test interactions between these smart contracts, and some sample tests you might perform to see how these smart contracts interact to provide access control.