Enhanced MIT ID Security via One-Time Passcode

Emma Christie, Nchinda Nchinda, Hannah Hailan Pang, Hyungie Sung

emchris7, nchinda2, hpang, hyun94

May 13, 2015

Abstract

The current MIT ID card system presents many flaws in various regards. MIT ID cards are used to establish security across campus by restricting access to certain areas through scanners. However, the interactions between MIT ID cards and scanners can be copied and replicated just through sniffing. MIT ID cards are also used to conduct payments, such as with TechCASH. However, the TechCASH exchange system is inefficient in that a physical swipe is required. Bitcoin would be a good alternative, but under the current system, MIT groups cannot be differentiated in Bitcoin transactions. Therefore, the current implementation of Bitcoin transactions is impractical. An enhancement to the MID ID cards is detailed and recommended in this paper. This enhancement will incorporate both a Yubico YubiKey, which is a token for authentication, and an NFC scanner such as an Android application.

1 Introduction

The MIT ID card is invaluable to MIT students, staff, faculty, and affiliates. One card allows the cardholder both access to secure buildings and rooms as well as ability to pay through the MIT payments system, TechCASH. However, the current system has many security flaws. The biggest flaws are in verification of identity (it is easy to sniff and forge MIT ID cards) and secure transaction (only an ID number is needed to take TechCASH out of someone's account).

Given the significant security flaws of the current MIT ID, a new design for a more secure ID card that still supports the functionality of the current ID is desired. In this proposal, we discuss our design, implementation, and security analysis.

2 Motivation

The motivation for this project comes from two areas:

- Relative insecurity of the current Radio Frequency Identification (RFID) technology used for MIT ID cards.
- Inconvenience and limitations of MIT TechCASH, which is the current monetary system used on campus that is linked to a person's MIT account.

These two issues affect many areas of campus life. For instance, getting into any locked and scanner-equipped room or building is done by simply tapping the ID card. Students, faculty, and MIT personnel gain access into labs, dorms, and certain locked rooms like the Reading Room this way. However, a recent student project demonstrated that simply sniffing the frequency at which the scanner and ID communicate was sufficient to be able to duplicate the access. The scanner and ID system, therefore, is not as secure as it should be.

Similarly, the MIT TechCASH system is linked to a person's MIT account, which can be accessed either by a physical swipe of the card or just the MIT ID number. Unfortunately, although an MIT ID number should be kept private, it is often needed by forms sent out by student groups or other similar programs to verify student status. In addition, it is possible to take out TechCASH from someone's account just using his or her ID number. In fact, this is how student groups used to often charge someone (e.g. class council selling sweatshirts, clubs selling care packages, etc.).

Recently, this policy changed in an attempt to provide more security: student groups are no longer allowed to charge students by taking down ID numbers. While this is a step towards better security, it is not foolproof, especially since ID numbers are still not kept secret (as one would a private key) in most settings.

MIT also added Bitcoin as a payment option for TechCASH through the payment gateway BitPay. This system is problematic and imperfect for two reasons. It removes a main feature of using Bitcoin, the ability for users to maintain control of their money and spend it with any entity at any time. By converting it to TechCASH MIT restricts the locale of usability to select locations on and off MIT Campus. Second, BitPay does not permit ACH payments to signify the origin and destination accounts. This would creates an accounting hell if departments, student groups, or other organizations on campus started accepting Bitcoin payments using MIT's BitPay account. As of now TechCASH is the only program accepting Bitcoin payments. Ideally the TechCASH system would use multi-signature wallets to allow MIT affiliates and groups as well as MIT administration to simultaneously maintain control of the money.

For all these reasons listed above, we are proposing a system of identity verification and secure Bitcoin transactions to be a more secure version of the MIT ID. Both of these functionalities are made possible by an implementation of a one-time passcode. The system has inherent security against computationally feasible attacks, making it more secure than the current MIT ID.

2.1 Previous Work

Some previous work has been done by others on the security of RFID, Near Field Communication (NFC), authentication devices, and secure Bitcoin transfers.

On the security of RFID, work published in IEEE detailed attacks and security measures. A paper published by the National Institute of Standards and Technology (NIST) also provided warnings about the security of RFID. In general, these papers demonstrated that RFID is vulnerable to man-in-the-middle attacks, data sniffing, data corruption, and eavesdropping.

Similarly, researchers from Philips Semiconductors analyzed the security of NFC, another way of radio communication. NFC is also vulnerable to some man-in-the-middle attacks, although these researchers also propose solutions. We use NFC in this project because of the shorter communication range (a max of 4 in) and the two-way communication capability.

We use a Yubico YubiKey in this system, which will be discussed in length later. Our research started with learning about the YubiKey's NFC and authentication through Yubico's publicly available GitHub code for their many applications.

2.2 Goals

The main goal of this system, motivated by the security flaws in the current MIT ID verification of identity and payment systems, is to create a more secure version.

Our goals are:

- To design a system that can support unforgeable authorization of identity.
- To add to this system support for a secure payments, in the form of Bitcoin.

3 System Design

This section describes on a high level the components incorporated in our system. It also outlines the interactions between these components in order to conduct verification of identity and Bitcoin transactions.

3.1 System Components

3.1.1 Yubico YubiKey

A Yubico YubiKey was used for both identity verification and Bitcoin transactions. A YubiKey is a unique physical token that provides a variety of authentication methods. The hardware consists of injection molded plastic encasing the circuitry, and the exposed elements on the token consist of military grade hardened gold. These physical attributes protect the token from internal damage due to exposure as well as from attempts to duplicate or record the token. Furthermore, as YubiKeys contain no batteries or moving parts, the tokens will never suffer failures from lack of power or mechanical issues. Power is provided through either a USB connection or through an NFC connection.

For the implementation of our system, we used a Yubico YubiKey NEO (Fig 1). The YubiKey NEO offers both contact and contactless communications. The USB contact communication works successfully with any operating system. The NFC contactless communication works with Android and select other devices that also support NFC communication. In theory it can communicate with any NFC enabled device, but has only been configured to do so with products that have non-proprietary documentation on interfacing with their hardware. There is little barrier to integration with the YubiKey as there exists open source server software as well. The YubiKey NEO supports multiple authentication methods: one-time passcode, smart card, and FIFO Alliance U2F. For the scope of our project, we worked solely with the one-time passcode configuration slot. We will discuss the one-time passcode (OTP) in more depth later in this section.

Given the light and small features of the YubiKey NEO, we believe that physical integration of the token with existing MIT ID cards would be feasible in the future. In the US, bulk orders can be purchased where each YubiKey NEO costs $48. For our project, we were able to gain access to two YubiKey NEOs through the MIT Bitcoin Club.

Figure 1: A YubiKey NEO has a mass of 3 grams and the dimensions of 18 mm x 45 mm x 3 mm.

3.1.2 Android Application

We built an Android mobile application in order to conduct contactless communication through NFC. The Android app reads the verification OTP from the YubiKey. Furthermore, the Android app connects to our own constructed server for handling and recording Bitcoin transactions as well as OTP verification.

3.1.3 Server

In order to handle and record Bitcoin transactions, we constructed a server that receives a user's public Bitcoin address, the payment amount, and the verification OTP from the Android app. The server will then

record the transaction in the database and actually conduct the payments through existing Bitcoin transfer applications. The server requires an internet connection in order to update the Bitcoin ledger. If MIT were to adapt our system, then both the Android app and the constructed server would be incorporated behind the scanners that already exist across campus.

3.2 Verification of Identity and Bitcoin Transactions

In order to handle identity verification and Bitcoin transactions, we used the one-time passcode configuration slot in the YubiKey. This slot permits configuring static text to be sent with an OTP, and we chose this text to be a user's unique public Bitcoin address. This unique address allows for identification and for receiving Bitcoin payments.

3.2.1 One-Time Passcode

A one-time passcode can only be used once, which disallows replay attacks. As the OTP from the YubiKey is used for both verification of identity and Bitcoin transactions, it is worthwhile to overview the contents of the OTP. The verification OTP is comprised from a number of sources, some of which are random. The sources are as follows:

1. Private identity of the YubiKey.
2. Counter fields to track how often the YubiKey has been used.
3. Time fields to track the time between each generation of an OTP.
4. Random number to add an additional layer of security.
5. Closing CRC16 checksum of all the aforementioned fields.

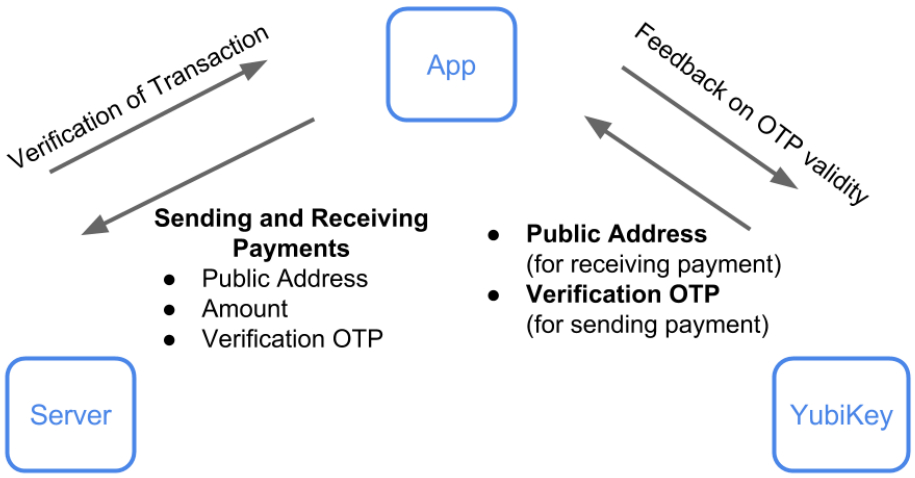

Figure 2: Interactions between the YubiKey, Android app, and our server.

3.2.2 Verification of Identity

The verification of identity is handled through the YubiKey and the Android app. The YubiKey sends along its public Bitcoin address as well as its verification OTP to the Android app through an NFC scan. The Android app then checks that the unique public address and OTP it has received are valid and sends a response showing the validity of the OTP. This communication process is shown in Fig 2.

3.2.3 Bitcoin Transactions

The Bitcoin transactions are conducted through the YubiKey, the Android app, and the server. For the scope of this project, we have focused on sending Bitcoin from the Android app to the owner of the YubiKey. The YubiKey sends along its public Bitcoin address to the Android app through an NFC scan. After verifying the identity from the scan, the Android app then sends the Bitcoin address, the amount of Bitcoin to be transferred, and the verification OTP to the server. The server then records this transaction and conducts the transaction through an existing third-party Bitcoin exchange application. The server then sends a response showing the verification of the transaction to the Android app. This communication process is shown in Fig 2.

4 Implementation

As a proof of concept we developed a sample Android application and corresponding backend on an MIT server. These two components, in conjunction with a properly configured YubiKey, were used to demonstrate how an NFC sensor could transmit arbitrary information, including (but not limited to) date, location, or a Bitcoin address along with an OTP verifying a particular person. The source code for the Android app and server is publicly available for fork or review on on Github. This section goes more into more detail how the components of the system were implemented and how they fit together.

4.1 Verification of Identity

Every YubiKey is configured with a private key that is stored on both the key and the server. The security of this system over the current MIT ID cards' is that this private key is not decipherable from the key's output. The YubiKey's OTP output is made of a 12 byte public identifier and 32 byte code, where each byte is one of 36 alphanumeric characters. Through a medium such as a mobile phone or another NFC enabled sensor the YubiKey sends these two strings to the server. On the server the identifier is matched with the secret key and used to decrypt the code. As explained in Fig 3, the server uses the data contained within this code to verify it's validity. To demonstrate its use, we designed and configured a YubiKey, Android app, and server backend to communicate with each other and the Yubico servers to validate a YubiKey. In this case, the Android app is functioning as any NFC sensor that can connect to MIT servers such as a card access sensor. When the YubiKey taps the sensor, the sensor relays the OTP information to MIT servers along with whatever data it wants. For building and room sensors, this could be the sensor's location or unique identifier. Now that the server is sure of the user's identity, it can take arbitrary actions, such as opening a door, finalizing a payment, or registering the user for an event.

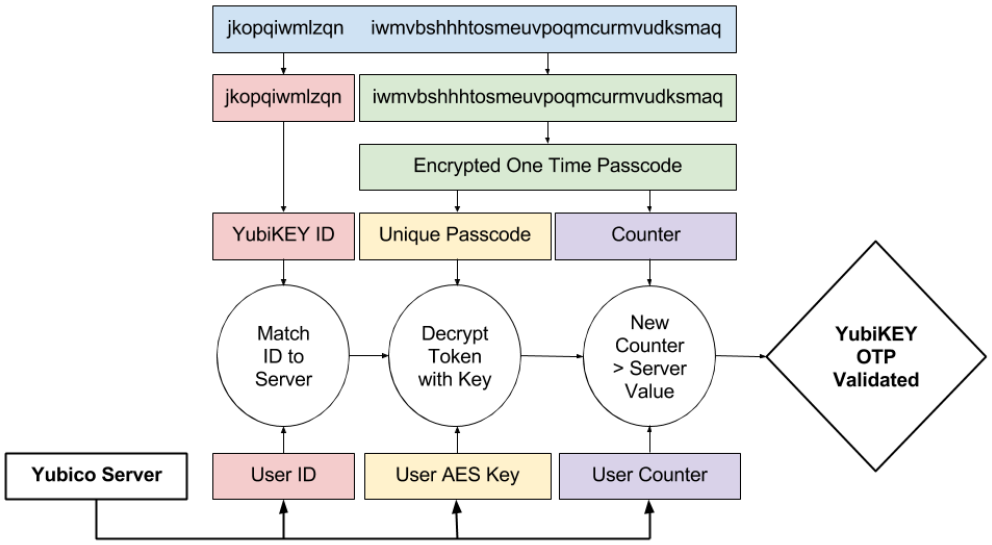

Figure 3: Breakdown of OTP from YubiKey and verification process.

4.2 Bitcoin Transactions

Every YubiKey permits configuring it with static text to prepend the OTP. We chose this text to be a Bitcoin public address to allow users to accept payments with their YubiKey. In our app, we demonstrate how both this static address and OTP can be individually retrieved from separate YubiKeys. In our implementation, selecting the correct Bitcoin address is as easy as touching a YubiKey to an NFC sensor. Bitcoin provides a safe and secure method of payment, which our design capitalizes on by using YubiKeys programmed with a Bitcoin public address.

4.3 App Design

The Android app was designed to demonstrate how the YubiKey could be incorporated into Bitcoin transactions. It includes fields for an amount (denomination unspecified) of money, a Bitcoin public address, and a YubiKey OTP (Fig 4). The address and OTP fields can be filled manually or using a YubiKey. To use a YubiKey, the user can select either "Address Mode" or "Verification Mode", and upon tapping a YubiKey the appropriate field is extracted from the key response and filled into the corresponding field. When the user presses the send button, all field data is submitted to the server. If the server response includes the phrase "status=OK", then the OTP was verified.

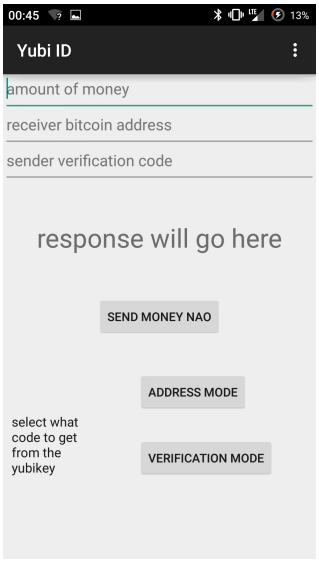

Figure 4: A demo application to showcase parsing data from a YubiKey and server communication.

4.4 Server Design

The server was implemented as a single JavaScript file running on Meteor. We hosted our server through Meteor as there was no barrier to entry and a wide existing community support. It accepts an amount, public address, and OTP, and it uses Yubico servers to validate the OTP. Yubico provides all their source code online; at the time of this writing, it can be found on their GitHub. This means MIT could host their own validation server for students, so the system does not rely on trusting a third party. Due to time constraints and because to do so is not integral to our project, we opted not to fully implement a clone of the Yubico Verification server.

5 Security Analysis

5.1 Threat Model

The security of both the verifaction of identity and the Bitcoin transactions rely on the security of YubiKey's one-time passcode. For our security analysis, we assume that the one-time passcode and the YubiKey itself are secure. We assume that the one-time passcode, specifically, can not be easily decrypted by an adversary. For the YubiKey, we assume that it may not be copied. That is, it is not possible for an adversary to covertly copy a YubiKey if it were to fall into his possession.

Our system is IND-CPA secure. Since the security of both aspects of our system, verification of identity and Bitcoin transactions, rely on the one-time passcode, the game will revolve around it. The game will be played as follows:

1. The examiner gives the adversary unlimited black-box access to a YubiKey and sensor, the encryption oracle. The adversary can use the oracle and other operations as long as they are polynomially bounded in number.

2. The adversary must output two plaintexts, m_1 and m_2. Plaintexts in this case are made up of the following components as outlined in 3.2.1 and reiterated for convenience:

   (a) Private identity of the YubiKey

   (b) Counter fields to track how often the YubiKey has been used

(c) Time fields to track the time between each generation of an OTP

(d) Random number to add an additional layer of security

(e) Closing CRC16 checksum of all the aforementioned fields

3. The examiner chooses uniformly at random from the two messages and encrypts m_b using OTP.

4. The adversary can perform additional encryptions and operations. The adversary outputs his guess for m_b.

If the adversary can win the game with probability greater than $\frac{1}{2} + \epsilon(k)$, then he wins. Since the one-time passcode encryption oracle uses randomness, it guarantees that the one-time passcode is IND-CPA secure. Given that OTP is IND-CPA secure, it implies that both verification of identity and Bitcoin transactions are secure.

5.2 Potential Attacks

We considered a number of potential attacks and mitigations.

5.2.1 Server Attack

One attack we considered was that an adversary could potentially gain access to a Yubico server and respond to OTP verification however he pleased. This has serious consequences, most notable of which is giving people access to areas they are unauthorized for. Another related attack would be a Distributed Denial of Service, DDoS, attack. In this case, the attacker would bring down the system and nobody would be able to access any secure areas.

5.2.2 Corrupt NFC Data

This attack, another form of Denial of Service attack, is one in which the adversary can intercept the signal sent between the YubiKey and sensor through NFC. While the adversary would not be able to use the communicated message to verify himself elsewhere, he could change some bits in the message and prevent the victim from being validated, even if the victim should be.

5.2.3 Brute Force Attack

An attacker could spoof an OTP. The attacker could enumerate through the possible OTPs until a positive verification came back.

5.2.4 Malware on Phone

An attacker could make an seemingly innocuous Android application that users would download, but it would actually be malware. If the malware were able to listen for an NFC communication. He could then change the public Bitcoin address to his own and the adversary could verify sending all the victim's Bitcoin to someone else, presumably himself. The adversary could also use malware to get in between app/server communication. In this case, he could also change the public address and send or receive Bitcoin at any public address.

5.3 Mitigations

For every potential attack described in the previous section, we have suggested possible mitigation.

5.3.1 MIT Hosted Servers

Instead of trusting a third party's server, in this case Yubico, we would implement the system so that MIT would host Yubico's code on its own servers. Even though MIT's servers are not impervious to being attacked or DDosed, with a smaller group of dedicated servers, the chance that MIT is attacked is lower. Instead, it is more likely that an adversary would try to take down all of Yubico's servers, as a larger group of victims would be protected. The MIT Information Security and Technology division already protects the current TechCASH and MIT ID card system from internal and external forces and we expect the same level of security would be provided to our implementation of the MIT ID card system. By using Bitcoin as a method of payment we can also expect the inherent security that comes with it.

5.3.2 Short Communication Distance

NFC, Near Field Communication, is just that. In order for two NFC enabled devices to talk to one another, they would have to be no more than four inches apart. What this means for the security of our systems is that in order for an adversary to intercept a message, he would have to be very close to the communication channel. So close in fact, that an attack is extremely unlikely. A YubiKey integrated into a card could be completely shielded by placing it into a most wallets.

5.3.3 Computationally Infeasible

In order for an adversary to brute force a valid OTP, he would have to enumerate the 44^{36} options due to the 44 byte long character and the 36 ASCII characters. With the technology we have today, it would take too much time for an adversary to enumerate these options and so this mitigation makes a brute force attack highly unlikely. Knowing a person's public identifier reduces the number of potential OTPs to 32^{36} which is still unfeasible to brute force. This could additionally be mitigated by the inclusion of a brute force detector, blocking connections from a certain connection after a reasonably high amount of invalid passcodes. A YubiKey will always provide a valid passcode, so any corruption from a valid code would happen in transmission. Experimental evidence is needed, but in our testing we never needed more than three tries to validate a YubiKey.

5.3.4 Aware Users

Preventing malware attacks is especially difficult, especially if users are unaware that these sort of attacks are in fact possible. The best mitigation here is to educate users to only download applications from trusted sources. Another recommendation is to not keep all of one's Bitcoin in their application. Users have much more to lose if they put all their Bitcoin in one place. Ideally the implementation would be with a multi-signature wallet so if at any point a user feels their account has been compromised then they can move the coins out of MIT's control. Users should also password protect their phones and install anti-virus software as an additional level of security.

6 Future Work

In the future we would like to see our system integrated into the MIT campus. In order to implement the identity verification component of our system, MIT would have to upgrade ID cards and scanners. ID cards would need to include a YubiKey and the scanners would need to be able to communicate with them. With this upgrade, building security would all see a large increase. MIT would no longer be vulnerable to sniffing attacks.

We would also like to implement our Bitcoin service across campus. In order to achieve this goal, we would need to successfully integrate with Bitcoin. Although we did not take this step in our prototype, we believe it will be a feasible task because we were able to accurately verify the requisite information. We would also need to further upgrade the ID cards, that is add the necessary components that would make the

ID card function as the YubiKey + App combination we used in our prototype. To do this, we would need to add a processing unit and wireless connection for verification of OTP and updating the ledger, respectively. Once all these steps were taken, MIT groups could accept Bitcoin instead of TechCASH. This method would be faster, easier, and more secure than the current system.

Even though we did not do an extensive financial analysis on implementing our system, we found that the YubiKey alone is $48, as mentioned earlier. Unless the prices were lowered, we understand that it may not be possible to upgrade MIT IDs with YubiKeys. We hope that purchasing the tens of thousands of keys necessary would further bring down costs.

7 Conclusion

After research into the MIT ID system as it is currently, we pinpointed some serious security issues in the current system. In response, we designed a system to add secure identity verification and payments, via one-time passcode.

The system we designed is feasible as a physical smart ID in the future. Our prototype represents an ID system with a combination of YubiKey and an Android application. In the verification scenario, the Android application acts like a MIT scanner for door access. In the Bitcoin payments scenario, the Android application processes the payments. In both cases, the YubiKey performs authentication by sending a one-time passcode (OTP).

We implemented this system by examining existing Yubico tutorials to write the Android application, and configuring the YubiKey to use OTP for authentication.

We did a security analysis of the system by looking at a threat model from an adversary. Our conclusion is that our system is IND-CPA secure, and we have mitigations for various attacks from an adversary. We also performed some simple future work analyses for the feasibility of implementing the prototype as a real, distributed system at MIT.

8 Supplementary Materials

The source code for the Android app and server is publicly available for fork or review on Github.

9 Acknowledgments

We would like to thank Professor Ron Rivest and the rest of the 6.857 staff for providing guidance and support throughout the entirety of this project. Furthermore, we would like to thank the MIT Bitcoin Club for generously providing access to a Yubico YubiKey NEO. We would not have been able to conduct and complete this project without their help.

References

[1] Ernst Haselsteiner and Klemens Breitfu, *Security in Near Field Communication (NFC)*, Web, 19 Mar, 2015

[2] A. Juels, "RFID security and privacy: a research survey", Selected Areas in Communications, IEEE Journal on, vol.24, no.2, pp.381,394, Feb, 2006

[3] MIT TechCASH, N.p, Web, 19 Mar, 2015

[4] "YubiKey Standard Technical Description", *Yubico*, Web, 1 Apr, 2015

[5] Daniel Ramsbrock and Stephan Moskovchenko and Christopher Conroy, Magnetic Card Swipe System Security, N.p, n.d., Web, 19 Mar, 2015

[6] Melanie Rieback and Georgi Gaydadjiev and Crispo Bruno, A Platform for RFID Security and Privacy Administration, LISA, Dec, 2006, Web, 19 Mar, 2015

[7] Ralf Wondratschek, Reading NFC Tags with Android - Tuts+ Code Tutorial, Code Tuts+, Tutsplus, 16 May, 2013, Web, 12 May, 2015

[8] Security Risks of Near Field Communication N.p, Web, 11 May, 2015