NoID- Sovereign Identity Management System

Achuthan Panikath, CSE, Ayan Sengupta, CSE, George Chiang, CSE, and Stephen Lendl, CSE

***Abstract*—Identity card design and management in its native form allows for unnecessary information being shared between parties. This, combined with the systemic inefficiencies and personal biases, enable the actors that abuse their power to pervade and mismanage personal information. We propose a sovereign system that intelligently manages private information hosted on your regular cards such as drivers' licenses and passports by hosting all this information on a virtual wallet accessed by a mobile app and a blank card (we call it the NoID card). The system is designed in such a way that it only communicates with validated authorities that possess a custom scanner and another version of our mobile application. These authorities can only ask questions that they are permitted to ask, based on tiered privilege access determined by the regulating organization. The questions will be posed via the app to the card holders' application and the data shared from the card-holder's virtual wallet will be the minimal version that answers the question. NoID effectively solves the issues of data leaks and abuse of power while furthering user convenience and improving security.**

I. Introduction

IDENTITY cards as used today rely on potentially outdated photographs of the cardholders and displays sensitive information for all to see on the surface of the card. Given that identity cards such as drivers' licenses are commonly used to verify one of age, address, name, state, and license validity but rarely all these categories, the data about the unrequired information counts as data leaks, particularly when accessed by organizations or individuals who do not require to see such sensitive information. Data leaks also exist at the level of an individual category where the date of birth is shared when only the age in number is required to be verified as greater than the legal minimum.

The rising issues of systemic discrimination and violence further puts into focus the impacts of unnecessary information being shared. Interactions between organizations and cardholders rely on the psychological impacts of power dynamics, biases, and the misplaced sense of authority such as in the case of an individual pulled over by a police officer. Regulating interactions by strictly defining privileges and rights and ensuring that the interactions abide by these metrics will substantially reduce the impact that these psychological elements have on information exchange.

A. Significance

Data leaks are one of the biggest asset crises we are facing today in the software world. When images of drivers' licenses got leaked online, about 80000 people were immediately exposed [1]: *"Each driver's license photo exposes multiple pieces of information about that individual, including license number, full name, birthdate, home address, gender, hair and eye color, height and weight, and a photo of the individual, among other things."* Data leaks targeting software enterprises have always been in the forefront of conversations. The magnitude of a successful attack exposing millions of people in an instant such as the devastating hack at Target Corp [2]. However, on a small scale every day, there is data being shared that was otherwise unnecessary. To prove our age, we share everything on a license or a passport at a movie theatre or a liquor store. A cop who just pulled someone over to warn them that their taillight is off, gets to see everything about them, when all they should ask for is if the license is valid. On the flip side, people validate identities by comparing looks of an individual to that of a potentially completely different looking individual on card. Further, the hassle of possessing multiple IDs to validate the same identity in multiple systems is crippling the efficiency and functionality of the identity management that has been largely stagnant and barely functional in the last few years.

B. Context and Competing Solutions in Marketplace

The questions posed by the stagnancy in identity management has been attempted to be answered at multiple levels. Estonia implemented its own e-ID [3] which is built on a KSI blockchain ledger. The system works for 99% of the public services as the required form of authentication. However, the ID still physically possesses all the information that is being verified and stored digitally. This continues the cycle of potentially identity theft or mismanagement which will not be the case with NoID. Natwest recently launched a state-of-the-art credit card that uses a miniaturized biometric module to authenticate via fingerprints [4]. This system is most definitely useful and is incorporated into our system but is in no way comprehensive enough to tackle all the complex interactions and data leakages that can still occur past a validated fingerprint usage. NoID uses fingerprint validation, blockchain and encrypted databases to host virtual wallets that interact with black cards and offers privilege-checks to the other half of the system (the half that is requesting information off of cardholders) to plug most of the holes that exist in the identity management system today.

C. Societal Impacts

There are two main stakeholders that would be provided an alternate form of interaction: members and merchants. Members constitute the regular cardholder who would be carrying around one or more forms of identification. These individuals will now possess a mobile application and one

blank card and will be able to determine who gets what data when requested with the assurance that only people in the system, authorized to ask for their data will be asking the questions and that only the minimal version of the answer will be provided to them. Possession of both the mobile device and the card will be important so one concern would be having a functional mobile device on them. Merchants will be the individuals working in organizations who need to validate clients' identities. They will be provided a scanner and a mobile application which will reflect the tiered privilege that they have been given by the organization. Merchants will have the assurance of proper and authorized validation without having to worry about fakes or eyeballing photographic similarities. They will also be protected from having unnecessary interactions with the clients as they can only ask the questions that are presented on the app.

A stakeholder that is potentially negatively affected is the government as it would require of them to structurally align themselves with the model here. This would have organizations/companies to work with NoID to setup the registration, access privilege definition and scanner provision. However, this minor inconvenience is outweighed by the benefits of a system that can remove human error and bias to a large degree and expedite the process of identity management and reduce the chances of identity forgery.

D. System Requirements and Specifications

We have defined the system requirements in Table 1. Each of the moving pieces of our system and the setup as envisioned at the start of this project is provided below.

Requirement	Specification
NoID Card	▪ RFID card ▪ Store fingerprint data ▪ Reading and writing stored information ▪ Max thickness of 1 inch
NoID Scanner	▪ Bluetooth compatible ▪ Process requests and sends in 5 secs ▪ Maintain power for 12+ hours in constant use ▪ Lightweight
Member mobile	▪ Network capability ▪ NoID Mobile app ▪ Password protected phone ▪ Username/password app
Merchant mobile	▪ Network capability ▪ NoID Mobile app ▪ Password protected phone ▪ Username/password app ▪ Bluetooth capability

Table 1: Requirements and Specifications

II. DESIGN

A. Overview

Given that our solution needs to consider an end-to-end system that not only provides standard security but also offers convenience in the face of a disruptive interaction, we decided to use a combination of technologies for a reliable system. Although we anticipated the cases where the lack of a mobile device could be seen as limiting, we believe that more often than not, mobile devices are on the individuals that own them. We claim that the possession of an individual's mobile device with its in-built password protection and ownership model along with the possession of our NoID card with its fingerprint validator is enough to authenticate user and prevent fraud. To complete a transaction, a member, as defined above would need to validate fingerprint, log in to the app on the phone, and permit to share the data being requested. This request will be coming from a merchant that is tied to the organization provided scanner and who has logged in to their app to send the request. The card- scanner interaction will be provided the unique card number to the merchant to make the request and all the interactions are spatially limited in that the technologies such as Bluetooth and RFID used in these interactions will need to happen in close range. We considered many different scenarios of data exchange which we discuss at length in Appendix (Section A) but decided to go with this interaction as it guaranteed the safety and convenience features, we were going for. An entirely software interaction was also considered but dropped given that the ownership of a phone/ potential of a non-working phone was extremely limiting and insecure. The possession of two systems with a biometric element substantially raises the bar for security. Though the technologies involved such as Bluetooth and RFID are using standard and secure protocols for communication, we are offering the packet in its encrypted form for additional security. Miniaturized fingerprint modules are new to the market and is hard to implement without additional resources and so the design is not as cleanly packaged as we would have liked.

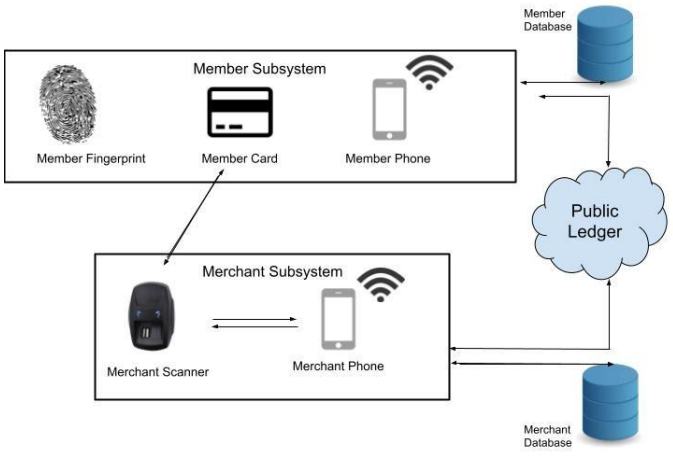

Figure 1: System Block Diagram

As can be seen in Figure 1, we have two major subsystems. The member subsystem includes the fingerprint, card and the application on the member phone while the merchant subsystem has the scanner and the app on the merchant phone. The member and merchant phones do not explicitly interact

with each other nor do the merchant get access to the member's fingerprint. The data hosted in the member database is rendered immutable by establishing verification checks on a public ledger. The same goes for the access privileges of a merchant. Date based checking can be implemented to see if the overwriting has happened unofficially recently. A layer of this system that should be explained is the registration process which is not in the purview of this paper but is discussed further along in the paper.

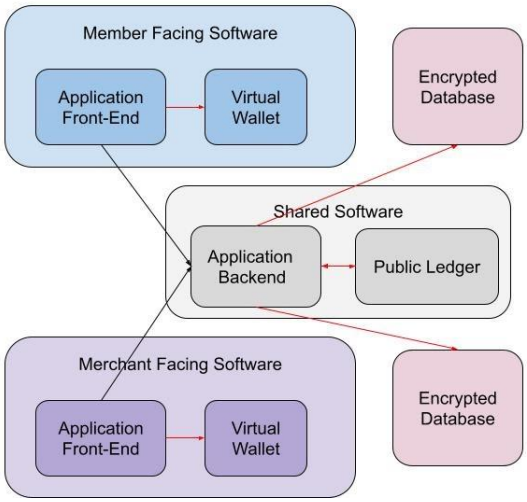

Figure 2: Software Block Diagram

A deeper look at the software diagram as shown in Figure 2 shows how the member and merchant phones do not ever directly interact but rather go through the application backend which serves as the endpoint manager and the brains of the project. It is here that most of the mediation happens which we will describe at length later in the paper. This portion of the system calls for the strictest security features and proper data management.

The public ledger is the next big piece in the system where checksums are stored for validation to prevent forgery of information by leveraging on the principles of blockchain technology and node-based public validation.

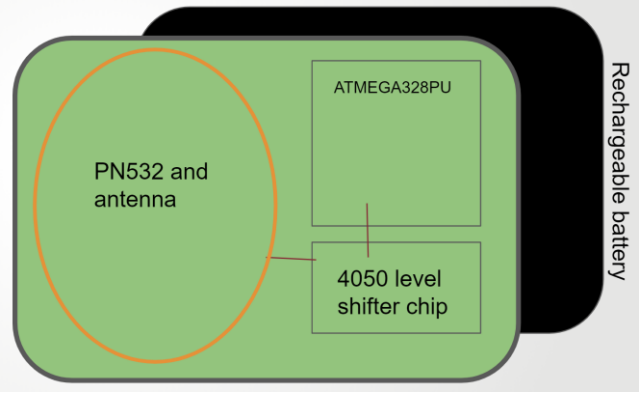

Figure 3: NoID Scanner Block Diagram

The scanner system shown in Figure 3 is responsible to mediate between the NoID card and the merchant phone. The card to scanner interaction is a typical but secure RFID communication where the card's unique ID is provided to the scanner. The scanner then passes along the id of the card to the merchant's mobile app using Bluetooth.

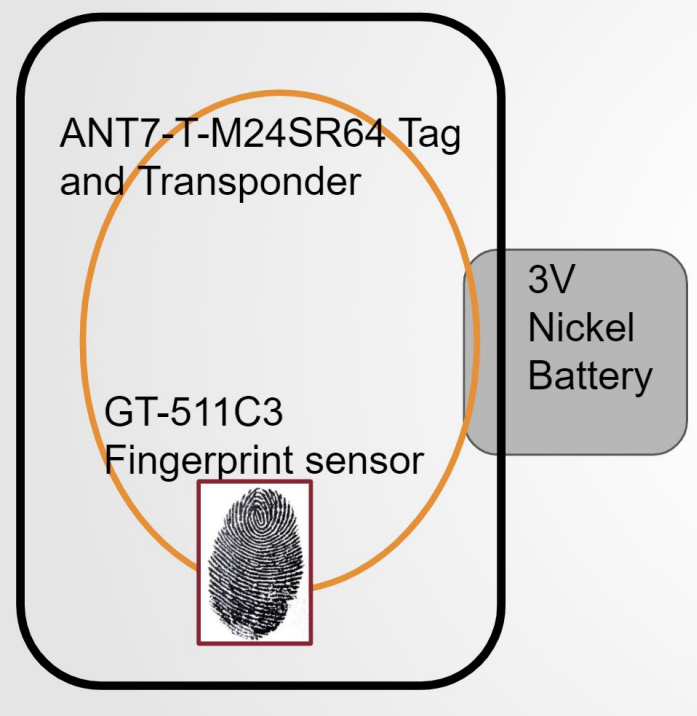

Figure 4 : NoID Card Block Diagram

The NoID card has been designed as shown in Figure 4 to incorporate both the fingerprint sensor and the RFID communicator. The tag can store minimal data and will pass it along to the Scanner upon the request. The fingerprint stored locally in the card will be verified by the scanner in order to prove ownership of the card.

NoID Registration Setup:

There is an external entity in the setup of the NoID environment. We require an entity such as an organization for the maintenance of the systems, the databases, the public ledger. This organization is also in charge of verifying the uploaded credentials by each member when they register by checking it with the corroborating evidence brought in person by the member to the organization. At this location, the biometric information presented is overseen and stored in its encrypted form in the NoID card which is then handed over to the member.

A similar process is followed for issuing access privileges to a newly registered merchant company. A super admin of a merchant company is given permission to assign access privileges based on the tier systems to the merchant employees who are part of the company. These merchants need to verify their credentials and employment status with documentation at the organization where a scanner will be provided to them along with access to the application.

B. Application Backend

The Application Backend is developed using JavaScript, using the fundamentals of full stack development in a Mongo-Express-React Native-Nodejs (MERN) stack. This forms the server which communicates with the application frontend for the member and merchant using dedicated APIs, the two encrypted databases implemented in a dockerized MongoDB container and the public ledger implemented using Hyperledger Fabric as shown in Figure 2. We have implemented gateway encryption for the encrypted databases on the application backend rendering the databases unreadable in its native form without the gateway defined and implemented in our system. This block receives and processes the data to be stored for members, the requests from the merchants, the permissions from the members and the privilege-based response modeling from the members' wallets. We use AES-256 and SHA-256 for encryption and password hashing respectively at different points in the backend, including to generate the Checksum used to populate and verify against the Hyperledger. Although, we never dealt with full-stack development in course, we were able to cross apply programming and security knowledge that we learned over the years in college and in internships/projects to develop this subsystem. Enterprise level full-stack engineering, Node.js, docker, express, MongoDB, security features are some of the topics that had to be thoroughly studied to implement this backend effectively. We used Postman to test the APIs developed and had to implement multiple security features by going through lists of standard vulnerability defenses in a typical MERN stack. Understanding where data rests and when data is finally deleted is important and had to be tested by logging data regularly to regulate proper data handling Encryption, hashing and communication protocols with different subsystems had to be constantly polished and checked to ensure reliable and consistent performance.

The databases always store the encrypted form of the data with encryption and decryption only happening at the gateway of our application backend providing state-of-the-art security. This increased testing complexity but attempting to log from the database outside the scope of our backend gave us satisfactory results regarding the security of the databases. Hashed passwords in the databases and the checksum generated during database entry that is subsequently stored in the Hyperledger further assured security of the database when checked outside the scope of the backend both theoretically and practically.

C. Application Frontend

The Application Frontend is developed using React Native, which is a JavaScript framework for creating mobile applications on both iOS and Android. The mobile app serves as the user interface for both members and merchants, wherein either user can choose to either register an account or login to an existing one. After a successful login, a member and merchant can perform a typical transaction which involves a merchant sending a request to view or verify a specific piece of information from the corresponding member. The requests that a merchant can make will be based on their access-privilege while members can accept or deny any requests for information. Although such an interaction appears to be peer-to-peer, neither the merchant nor the member is directly communicating with each other. All requests and responses are sent and verified through the backend before being redirected to the intended recipient.

None of our coursework has ever been directly related to working with React Native, or even mobile applications in general. However, many of the core fundamentals in programming learned from our courses remain relevant overall and significantly reduced the learning curve of working with new languages and frameworks.

Through usage of powerful libraries such as Axios and React Router Native, we were able to establish communication and a flow of data between the front and back end. Other libraries such as React Native Bluetooth HC05 and Encrypted storage were needed for local functionality such as storing sensitive data on the mobile device using device encryption or communicating with the scanner through a proximal connection.

D. Public Ledger

The public ledger utilizes Hyper Ledger Fabric to create a blockchain network that is then used to verify the checksum value that is generated for each member. Our current network consists of a singular dedicated ordering node that is responsible for initiating all transactions on the ledger and two organizations with each organization having one peer node that take part in the consensus algorithm and validate transactions. Each organization also has its own certificate authority that offers membership services to each organization. Each node mentioned is a separate docker container running locally on laptop running Ubuntu 20.0.4 Focal Fossa.

Understanding the concepts of blockchain and then implementing them in a production level network was quite challenging as none of our course work relates directly to this, however the security concepts learnt in security engineering courses certainly make the learning curve a lot easier.

The final ledger network is able to fully communicate with the back end using the hyper ledger fabric node software development kit to establish a connection profile and invoke the chaincode functions on the network. We also developed our own chaincode to write and update assets (UID and checksum of members in this case) on the ledger.

E. Scanner and RFID module

The hardware is split into two systems, the scanner which is distributed to certified merchants and a RFID card module that is given to individual members.

The RFID module is a mix of an RFID card and key FOB, using the ANT7-T-M24SR RFID tag from STMicroelectronics and the GT-511C3 fingerprint module to verify biometrics and has a nickel battery 5V power supply to power the processor on the fingerprint module. This system is quite simple, the RFID module has internal logic that if the VCC pin is tied to a voltage higher than a certain threshold, the RFID does not transmit. So, the RFID is left inactive for most of the time, not allowing for reading or writing to be done, and a pull-up resistor and some circuit logic allows for the fingerprint module to toggle this, so when it receives a valid fingerprint it has stored, it will hold that VCC pin low for a set amount of time, allowing for the communication with the scanner, after which it will switch the RFID module back to being inactive.

The scanner is made up of two main components, the PN532 NFC chip with an antenna built into the circuit itself, and an ATMEGA328PU to handle to act as the main processing unit. The scanner will first act as a reader, reading the UID of an RFID tag, add its own personal UID, and then mirror that tag, acting now as a tag, allowing for the built-in functionality of our phone app to transmit this information to the merchant's phone, where the merchant will then be able to make a request to view some piece of information about the account linked to that UID of the tag.

We first worked to get a working flow of communication from a regular RFID tag to the scanner, then attempting to establish communication to the phone, then to the app within the phone. We then wrote our own code to the ATMEGA that would read a tag and then switch to mirroring the tag. We went through a lot of prototyping and experiments to initially establish any kind of connection, and then built up from those areas.

This section used a large swath of learning we did in several classes, from Professor Malloch's independent study courses to Professor Baird's classes on digital and circuit logic to Professor Kelly's circuits classes.

THE REFINED PROTOTYPE

A. Prototype Overview

The prototype we have developed to demonstrate the functionality makes a couple of assumptions.

1) NoID card has been properly registered, with ID validation done against passports/licenses.
2) NoID scanner has been issued to the merchant as per guidelines.

We create a member in the database and assign them a Unique ID, which is then stored in the card that would belong to that member. Our Hyperledger fabric setup permits us to create an entry in it which be storing the Checksum hash value generated in our database corresponding to the data chunks added to the member. Our card-scanner interaction will pass this value using RFID communication, which is then further passed to the merchant phone using Bluetooth technology. The merchant phone-frontend would then pick a question from a sample list of questions offered to them, prompting a permit/deny option on the member phone end. Upon accepting, we kick off a search for the data associated with this member in the database. Using the checksum associated with the data entry stored at the time of data creation against the Hyperledger fabric, we assure data integrity and pull the data associated with the member from the database. Our backend then performs some sample computations to pass on the minimal version of the data to the merchant front end completing the transaction. Figure 5 demonstrates this workflow for better comprehension in a simplified manner.

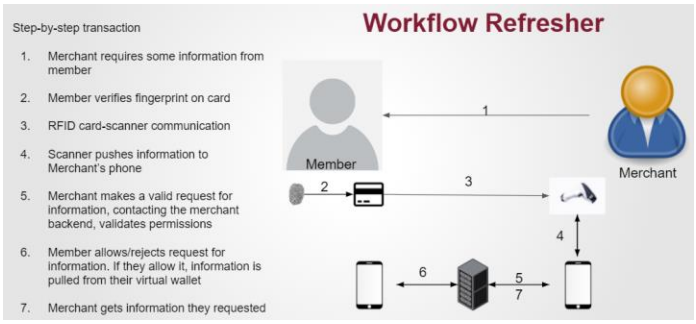

Figure 5: NoID prototype workflow

B. List of Hardware and Software

Software:

- Hyper Ledger Fabric
- MongoDB
- Docker
- Express
- NodeJS
- React Native
    - Axios
    - React Native BluetoothSerial-HC05
    - React Router Native
    - React Native Encrypted Storage

Hardware:

- PN532 chip and antenna
- Adafruit PN532 Breakout Board
- HC-05 Bluetooth module
- ATMEGA328PU IC
- ANT7-T-M24SR RFID tag
- GT-511C3 Fingerprint module
- AA Battery housing with barrel jack input

C. Custom Hardware

Our design phases for PCB design were in two distinct parts. We first relied heavily on the PN532 breakout board [7] by Adafruit to start filling in gaps of how we were going to

communicate. Once we were able to pin that down, we moved from an Arduino Uno board to our own circuit to house the ATMEGA328 and PN532 in combination. Once we knew our ATMEGA circuit worked the same as the Uno board when communicating with the PN532, we created a second, final PCB that housed both the ATMEGA and PN532 with its antenna on the same PCB. Our final PCB in the end was non-functional from a mixture of difficulty soldering the PN532, a 40-pin SMT part that is highly sensitive to high temperatures. We only ordered 3 IC's because of their high price per unit, and we were unable to successfully solder all 3 using a variety of methods like hotplate, oven and hand-soldering. In addition to having a non-functioning PN532, through some testing once we received the circuit, we also believe the resonant impedance of the antenna is off, and may or may not successfully receive 13.56 MHz signals.

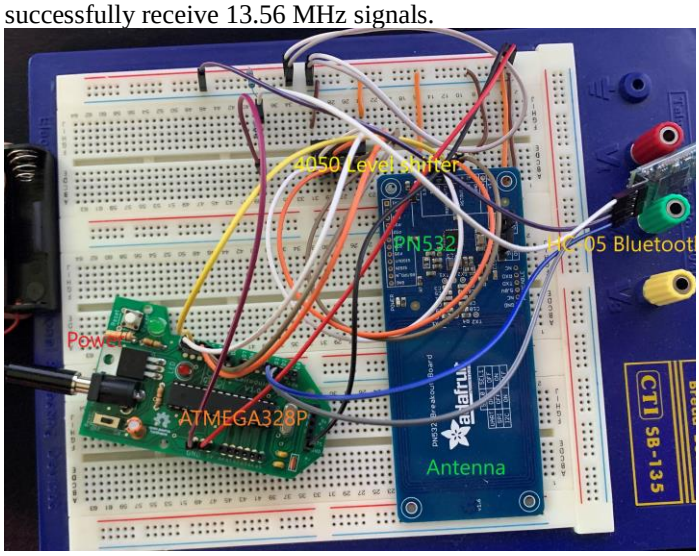

Figure 6: NoID prototype scanner with custom ATMEGA IC

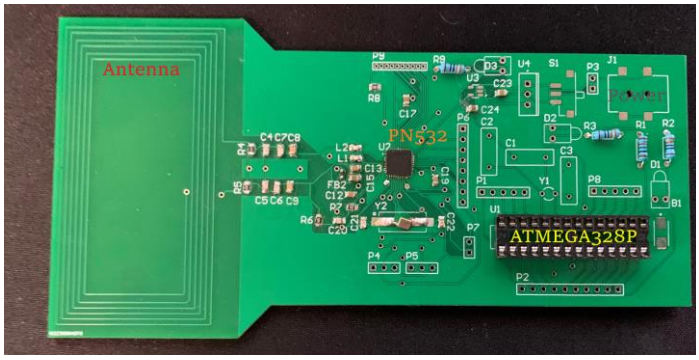

Figure 7: NoID prototype final PCB

In addition to creating a custom scanner PCB, we also had to create a custom circuit for the fingerprint and tag. The card circuit is composed of the custom ATMEGA328P IC, ANT7 tag and the GT-511C3 fingerprint module [8] with power input from a AA battery using a barrel connector. The card module is fully functional and meets all the specifications.

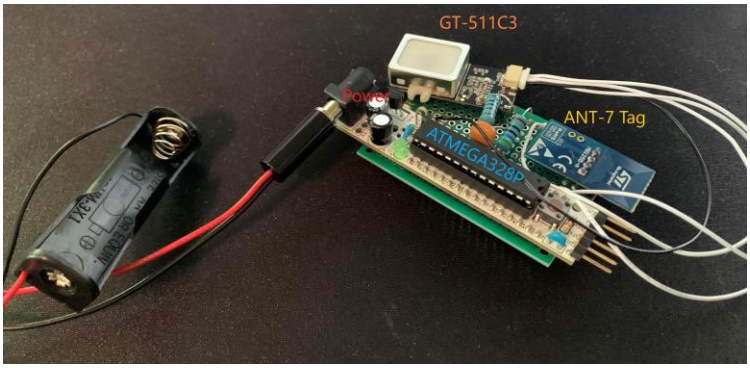

Figure 8: NoID final prototype fingerprint card

D. Prototype Functionality

All the elements present in our diagrams were functioning in our prototype. From the member subsystem, members can successfully activate an RFID using a fingerprint to pass their unique identifier to the merchant scanner. Afterwards, a merchant can login to their merchant account to read the unique identifier from the merchant scanner and proceed to send an information request to the backend. The member can then login to their member account and view the incoming request before choosing to accept or deny the request. The members response is then returned to the backend and an appropriate answer is generated in the backend logic. The merchant can then view the answer generated from the backend and respond appropriately. Each step during the transaction process is verified and recorded in the public ledger which we can view.

F. *Prototype* Performance

Based on the system requirements and specifications listed in Table 1, we were able to meet all the core specifications for our prototype. Some specifications were not able to be met but we were able to find design alternatives that met our requirements and enabled prototype functionality, while some of the more trivial ones that could be not had no real negative impact on the prototype.

The first specification we had issues with was involving NFC, as some of our devices were unable to properly communicate with the scanner through NFC even if they had NFC support. We attributed this to compatibility issues between the scanner and the NFC chips contained in our mobile devices. As a result, we resorted to using a Bluetooth connection through the React Native Bluetooth Serial library for the HC-05 Bluetooth module. This resolved our communication issue between our phones and the scanner as our phones could now communicate with the scanner to retrieve the unique ID from the fingerprint.

Another specification that was not ultimately able to be met was having our card functionality be integrated into an actual blank card as such a prototype would require state-of-the-art development kits that are well outside of our budget. We spoke with the evaluators and they felt it was perfectly

reasonable to change the prototype to be modeled after card to a key FOB. Nonetheless, this specification was not a core requirement of the prototype and we were ultimately able to meet all the other promised deliverables for the member card.

I. Conclusion

NoID in its current form is a powerful prototype that can be used for dependable storage and verification of IDs. However, for this project to be truly effective, it must be scaled and launched at a governmental level. The tools used to develop the different layers of the project would then substantially change given that the constraints are lesser and the resources greater. NoID when implemented as a substitute for multiple identity cards as the centralized form of identification would exponentially increase the ease of use, increase the complexity of the ledger, making it that much more secure and would be effective in tackling its primary goals of reducing the abuse of power and plugging data leaks in day-to-day life.

ACKNOWLEDGMENT

We would like to thank our advisor Professor Krishna for working with us on a weekly basis throughout the development of this project. Our SDP faculty Professor Baird Soules, Professor Kris Hollot, Shira Epstein, and Professor Chuck Malloch were instrumental in the progress of this project. Our evaluators, Professor Wayne Burleson and Professor Beatriz Lorenzo always left us with insightful thoughts that helped shape the prototype into its current form. UMass Engineering department, our fellow classmates and M5 Makerspace were accessible and dependable resources over the last year. We would not be able to meet our goals had it not been for all those mentioned above.

REFERENCES

[1] Phil Muncaster UK / EMEA News Reporter, “Over 80,000 ID Cards and Fingerprint Scans Exposed in Cloud Leak,” *Infosecurity Magazine*, 17-Nov-2020. [Online]. Available: https://www.infosecurity-magazine.com/news/80000-id-cards-fingerprint-exposed/. [Accessed: 08-Apr-2021].

[2] T. Kitten and R. Ross, “Target Breach: What Happened?,” *Bank Information Security*. [Online]. Available: https://www.bankinfosecurity.com/target-breach-what-happened-a-6312. [Accessed: 08-Apr-2021].

[3] “ID-card - e-Estonia,” *e*, 18-Oct-2019. [Online]. Available: https://e-estonia.com/solutions/e-identity/id-card/. [Accessed: 08-Apr-2021].

[4] “NatWest first UK bank to unveil biometric credit card,” *Leave feedback for*. [Online]. Available: https://www.rbs.com/rbs/news/2019/10/natwest-first-uk-bank-to-unveil-biometric-credit-card.html. [Accessed: 08-Apr-2021].

[5] "Getting Started — hyperledger-fabricdocs master documentation", *Hyperledger-fabric.readthedocs.io*, 2021. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.2/getting_started.html. [Accessed: 04- May- 2021].

[6] Achuthan Panikath, Ayan Sengupta, George Chiang, Stephen Lendl, NoID, (2021), GitHub repository, https://github.com/achupanikath/SDP2020

[7] Adafruit Learning System, “Adafruit PN532 RFID/NFC Breakout and Shield,” *Adafruit*, 17-Mar-2020. .

[8] ADH Technology Co, Ltd, “GT-511C1R Datasheet,” *Sparkfun*, 22-Feb-2016. .

APPENDIX

G. A. Design Alternatives

Initially, we were set on having the fingerprint module be a part of the scanner system, but after some conversations with the SDP faculty, particularly Professor Soules, we were convinced to migrate that portion to the RFID tag system, which was needed and great change in the design. After this change, we were having more trouble with the tag system, keeping it as originally proposed in PDR, keeping the tag in a card-like shape. We met with many people to help make some decisions, and Professor Burleson told us that as long as it’s handheld it will be useful. So, we changed the design specifications to reflect more of key FOB that is used in cars, and that relieved many of the issues we were facing at the time.

As mentioned previously, we also had originally proposed the usage of NFC for communication between mobile devices and our scanner, but we encountered difficulties when performing inter-device communication. We resorted to using Bluetooth with our HC-05 module as an alternative which was able to successfully communicate data between the phone and scanner.

B. Technical Standards

To develop our application, we used the MERN stack (MongoDB for data base, Express, React, Node.js) which is a standardized software development stack used in production level projects.

For hardware standards, we primarily relied on the Altium software to confirm our PCB prototype was standard and performed as we wanted. There was also built-in software in Altium that helped with antenna creation and verifying that it worked in the 13.56 MHz frequency.

C. Testing Methods

Our main methodology for testing our project was to make sure our subsystems worked independently, such as the hardware communication or the backend Hyperledger. We made sure the systems worked on their own and performed as we wanted, and then proceeded to link all of the subsystems together, continually validating that the subsystems performed.

Once we had the final PCB, we had to verify the functionality of the antenna. We used the built-in antenna creation software within Altium to the best of our ability, both using the Adafruit Breakout Board as a guide and doing as much research online. Once we received the PCB, we used the oscilloscope to begin testing to see if the antenna had the correct impedance, and to the best of our ability we think that the antenna is just within the threshold for 13.56 MHz communication, however we were unable to test its full functionality

D. Project Expenditures

Part	Cost
PCB printing prototype 1	45$
ANT7 tags	40$
PCB population parts	120$
PCB printing final board	50$
Total	255$

E. Project Management

Overall, our team was well-organized and meshed well with each other when it came to working as a team. Since we were all Computer Engineers, we knew that the hardware portion of the project would most likely be our toughest challenge, but Stephen chose to undertake the hardware lead role since he was the most comfortable with the challenge. The software end of the project, which included the backend, frontend, and public ledger was more in line with what most of the team was familiar with. Although each team member tended to focus on their own responsibilities for developing each subsystem, the team was able to meet frequently for updates on progress and to ensure that each team member was on the same page. Achu stepped forward as team lead to spearhead the delegation of responsibilities as well as developing the backend. Ayan and George took the initiative on public ledger and mobile app development, respectively. Furthermore, team members did not hesitate to reach out for help or give assistance when needed, such as when George reached out to Achu to begin the integration of the frontend and backend workflow or when Achu volunteered to assist Stephen and Ayan with their respective subsystems. Although there were times when there was a lull in communication, such as during the holidays, these periods were infrequent and the team was able to quickly recoup and get back on track.

F. Beyond the Classroom

Achuthan Panikath: I have been in charge of both backend development and security engineering for the NoID ecosystem. As a backend engineer, I had to create and manage a secure database developed in MongoDB in a dockerized environment. This along with the Nodejs backend and the Express middleware makes for a JavaScript intensive backend with dedicated API development. These technologies were primarily learned during internship experiences and cross applied to create an enterprise standard application backend. Multiple levels of backend and middleware securities were integrated into this to create an industry standard system. In my capacity as a security engineer, I was also responsible for outlining and evaluating the different kinds of threat vectors at different stages of our ecosystem and mounting defenses and security features at each of those levels. This was an extensive process that involved using a lot of theoretical knowledge that I learned from Professor Burleson, Professor Ganz and from Mitre's instructors during the CS590J. Further, I was able to assist Stephen in his hardware responsibilities, Ayan with Hyperledger integration and George with React Native front end setup. As the team coordinator I was also charged with ensuring appropriate division of duties and responsibilities to ensure timely solutions to problems.

Ayan Sengupta: Although almost no problem that I faced was directly related to exactly what we learned in class; a lot of the concepts were very useful. Concepts like data structures and algorithms played an important role in understanding a lot of the documentation and example codes for the Hyperledger fabric network and chain code. A basic understanding of networks also helped when establishing a connection between the test-network and the backend.

Stephen Lendl: Being the designated hardware lead of the team, I had to learn a lot past what we learned in the circuit and hardware classes, although they provided a much-needed foundation for my learning. I relied heavily on our previous classes when it came to getting prototypes working using Arduinos.

George Chiang: Coming in to this, I had no prior experience working with React Native. As a result, this project was a significant learning experience for me as none of the classes I had previously taken had really gone in-depth with regards to working with JavaScript or mobile applications. Although previous courses provided a helpful foundation for programming paradigms, much of what I accomplished was from a self-taught perspective and involved extensive trial and error.