zkTAP: A Zero-Knowledge Trustless Authentication Protocol

By

Majdi Hassan

Joseph Kuo

Lilan Yang

Design Document for ECE 445, Senior Design, Fall 2019

TA: Evan Widloski

1 October 2019

Project No. 33

1 Introduction

1.1 Problem and Solution Overview

On January 19, 2010, Muhammad al-Muhbound was relaxing in his hotel room, when a group of four individuals opened his hotel room and assassinated him. It turns out that they had reverse engineered the RFID hotel "master key" and they were able to gain entry to his room. This underscores that many RFID authentication systems do not employ adequate security.

We can formulate the problem to better understand what happened that night: Alice (the tag) and Bob (the tag reader) want to communicate with each other over an unsecured channel (wireless). Eve (the assassins) is trying to eavesdrop their conversation.

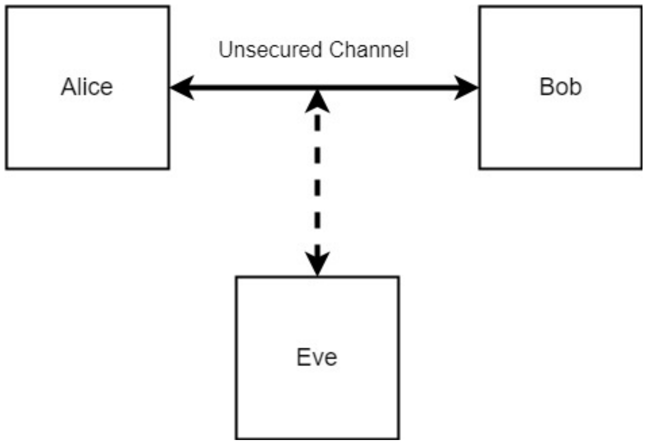


***Figure* 1 : *Illustrative Example* 1**

Unfortunately for Alice, there is nothing actually stopping Eve from eavesdropping.

Current implementations such as Mifare DESFire employ symmetric key cryptography, to deal with this problem. This require that all parties know a secret key. In order to send a message, would require both Alice and Bob to know a shared secret key. Alice would first encrypt her message using the secret key and then transmit it over the unsecured channel. In order for Bob to read her message, he needs to decrypt the encrypted messaged from Alice. While Eve has a copy of the encrypted message, she cannot read the contents of the message without knowledge of the secret key. The advantage is that these tend to be computationally inexpensive, and thus seem to be a good choice for a RFID tag. However, the most obvious drawback is that it is difficult to securely distribute these secret keys! This is since after one member releases the secret key out, now everyone can eavesdrop.

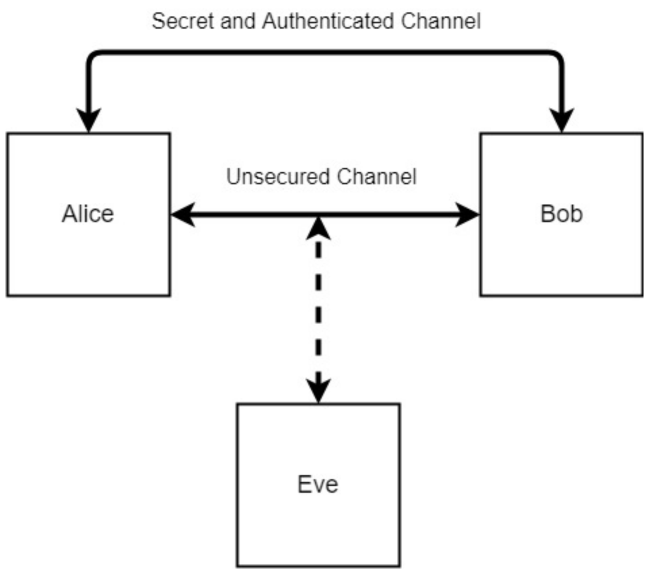


***Figure* 2 : *Illustrative Example* 2**

Sometimes these keys get leaked, such as in the London Rail system, where attackers were able to get free rides on their underground system [1].

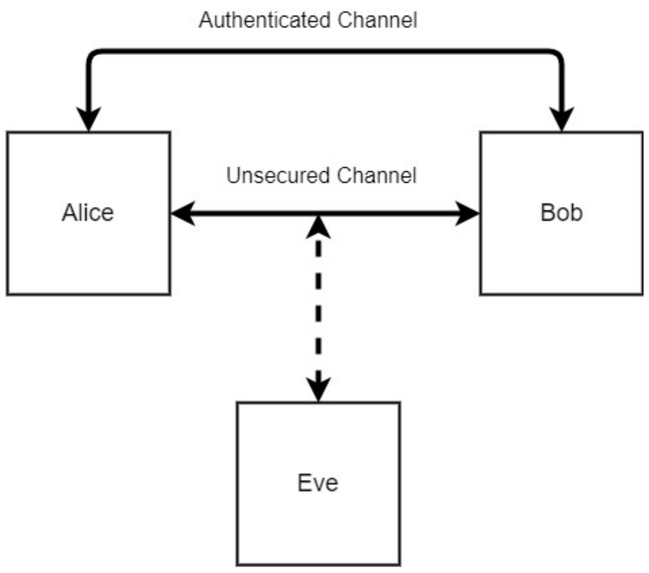


***Figure* 3 : *Illustrative Example* 3**

In contrast to symmetric-key cryptography, we will be employing a form of public-key cryptography (PKC) instead. One possible application goes as follows, every member now generates a private-public key pair. The public key is freely disseminated. In order for Alice and Bob to communicate, Alice uses her private key and Bob's public key to encrypt a message. Now in order for Bob to decrypt the message, Bob needs to use his private key, along with Alice's public key in order to decrypt the message. Eve cannot read the message as it requires knowledge of either Alice or Bob's private key.

This has several benefits when compared to the symmetric-key cryptography system. In a symmetric-key cryptography system, it becomes more difficult to securely distribute the secret key as the number of users increase. In contrast, PKC only requires that the recipient's public key is authentic. Now suppose that one member of the party decided to reveal their private key. Only communicated to and from that user is comprised. On the other hand, if that same user were to reveal the secret key in a symmetric-key cryptography system, everyone's communication would now be comprised.

We propose to create a RFID Tag/Reader System that uses public-key cryptography. The tag will be an active RFID tag which will use a MSP430 to perform the public-key cryptography and it will feature a hardware random number generator.

1.2 High-Level Requirements

- The response given by the correct tag must be accepted by the reader with all but negligible probability (on the order of being rejected 1% of the time).
- A user must be authenticated in under 10 seconds.
- An eavesdropper must not be able to impersonate a user (except by guessing responses).

2 Design

2.1 Block Diagram

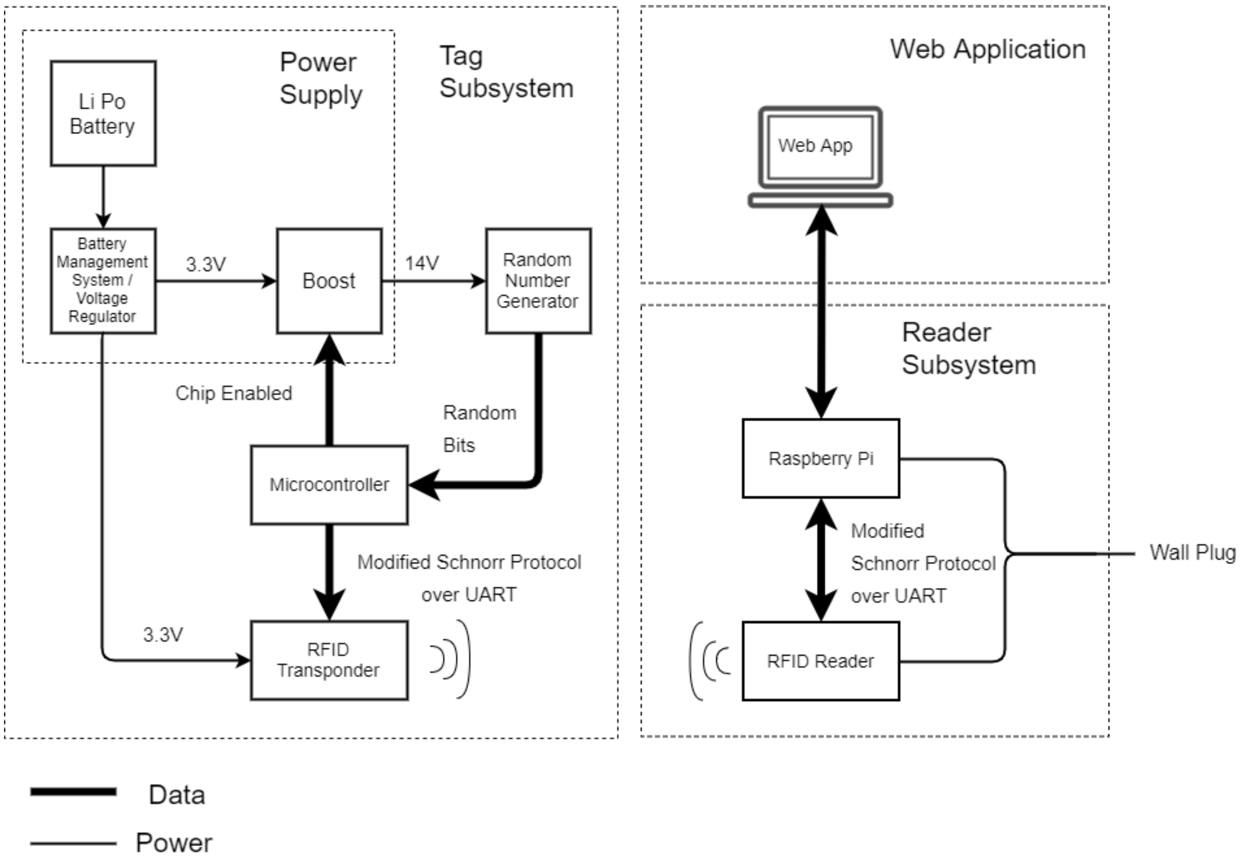


Figure* 4 : *Block Diagram

We assert that our protocol satisfies our first requirement. A short and straightforward proof is attached in Appendix A.

For requirement 2, we will make the assumption that the processing power of the reader is at least a magnitude greater than the microcontroller. Thus any computational costs of the reader are therefore negligible. Thus we can reduce our problem into: generation of a 255 bit random number, 2 Elliptic Curve Multiplications, 1 addition (mod p). An elliptic curve multiplication on Curve22515 takes $12.34 * 10^6$ clock cycles on MSP430 microcontroller [2]. A MSP430 micrcontroller can run between 8 MHz-24 MHz depending on the generator of the microcontroller [3]. Lampert circuits can be safely sampled at a rate of around $10^4 \sim 10^6$ Hz [4]. Thus we can expect that the random number generation portion takes roughly 0.03 seconds. In addition, we will assume that addition take negligible time. We will assume the worst case scenario which would imply that our protocol would take about 3-4 seconds. We could likely get this down to 1 Elliptic Curve multiplication if we allow for precomputing. There are two ways to further speed this up, we could a 32 microcontroller or use a FPGA + microcontroller combination.

Requirement 3 can only be realized if we have a random number generator on the tag. Consider the dishonest reader example. The tag's response is $s = d + a + tc$. If a reader were able to predict the value of t, then that reader would be able to extract the secret key a, as s, d, t and c are all known to the verifier. By introducing a random number generator, we can maintain privacy of the tag.

In the case of an honest verifier, an eavesdropper actually can't figure out whether or not authentication was successful, as the value d or d' can only be calculated by the tag and by the reader.

2.2 Overview of Cryptography Protocol

Before introducing our cryptosystem, we would like to introduce the following things. Let p be a prime number, then the finite field F_p represents the set of all integers from 0 to p-1. In addition, we define addition to be integer addition (mod p) and mutliplication to be integer multiplication (mod p) (where multiplication does not contain 0). Let G be a finite, cyclic, and multiplicative group. Also, G contains a generator g, where the elements of it are $g^i : 0 \leq i \leq q - 1$ where q is the order of G. We will be using elliptic curve cryptography, where G represents a point on an elliptic curve over a finite field. We will be using Curve25519, which has the following parameters.

$$p = 2^{55} - 19$$

$$y^2 = x^3 + 486662x^2 + x$$

$$q = 2^{256} - 1$$

G has an order of 2^{252} plus a big number. We will not be including this big curve as it has not be including this big curve curve as it has no elegant form of representation. Note that the following operator [A].x represent the x-coordinate of the curve point.

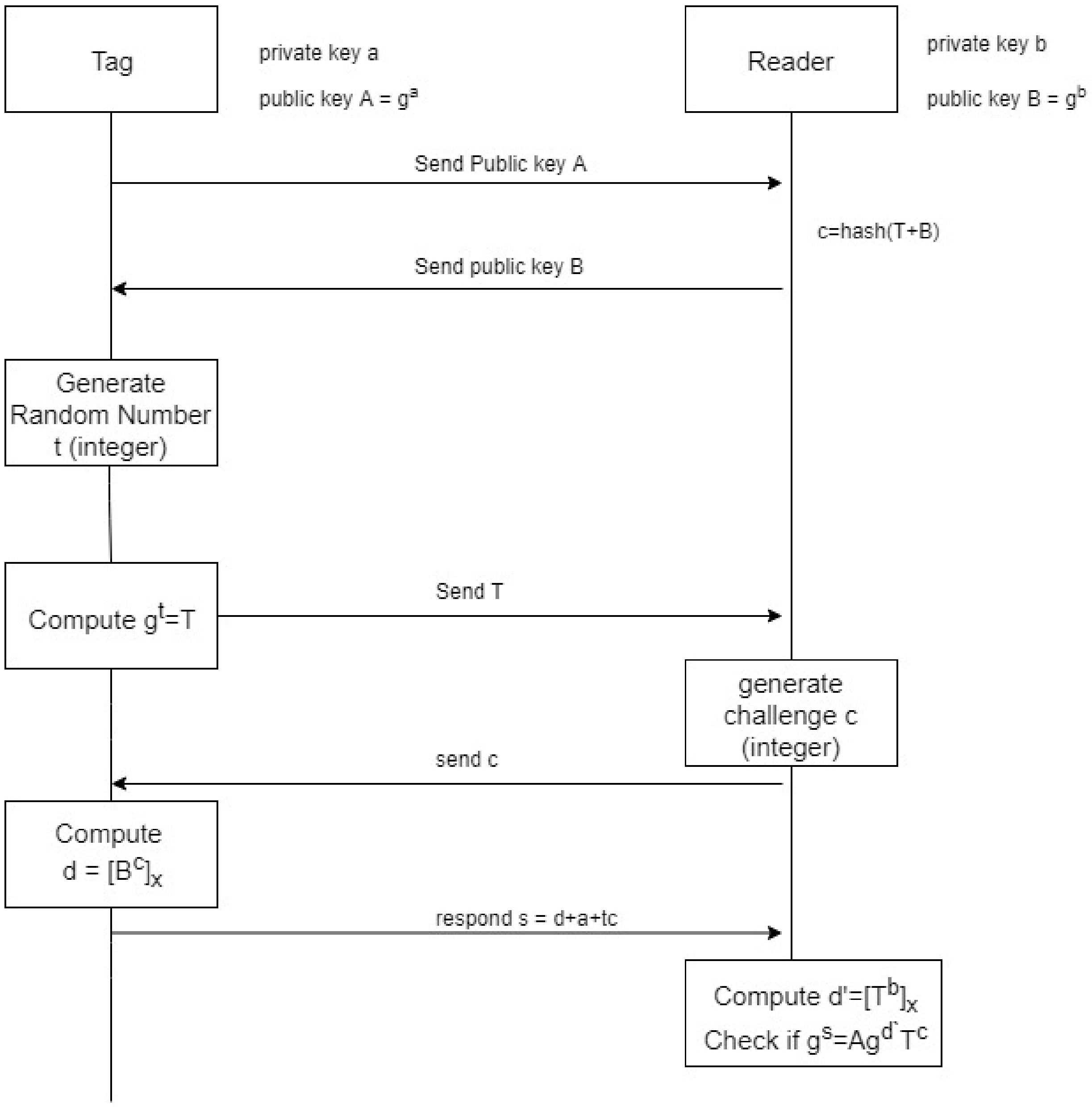


Figure* 5 : *Protocol

This follows several main steps. Our protocol is based off of the Schnorr protocol outline here [5].

1. Exchange public keys.
2. Tag generates t, a blinding factor and sends a commitment to the reader.
3. The reader generates a challenge.
4. Tag responds to the challenge.
5. Reader verifies that the challenge is correct.

The diagram shows the interaction between the RFID reader system and the RFID tag system. The interaction between the two systems is executed in a series of challenges and answering those challenges. The reader will send out a challenge to the tag which only the tag can solve because it holds the key information needed for the challenge. If the tag responds correctly, the reader will authenticate the tag

2.3 Physical Design

2.3.1 Tag System

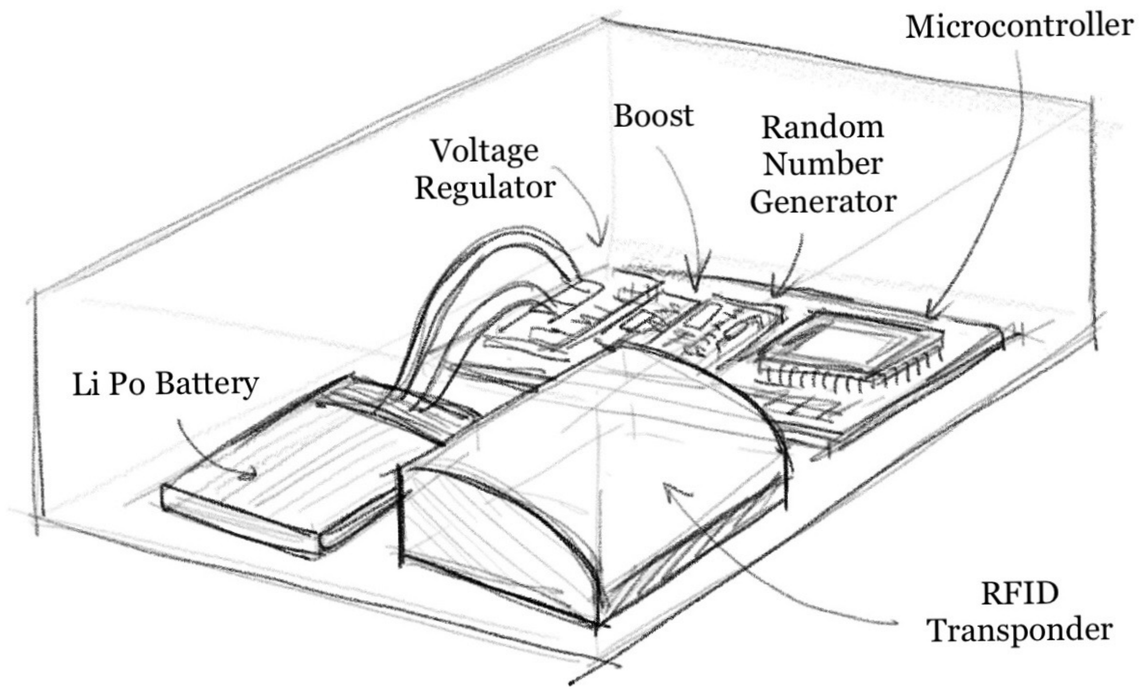


Figure* 6 : *Physical Tag Design

In order to group all the components for the tag system, our plan is to put them inside a box that is large enough to hold Li Po Battery, PCB and RFID Transponder. The desired Li Po Battery would approximately have the size of a card and RFID transponder would be similar to I-PASS transponder mounted on the windshield of the car. We will have two separate PCB's, one with the power supply system and other board will feature the hardware random number generator, microcontroller and transponder.

2.3.2 Reader System

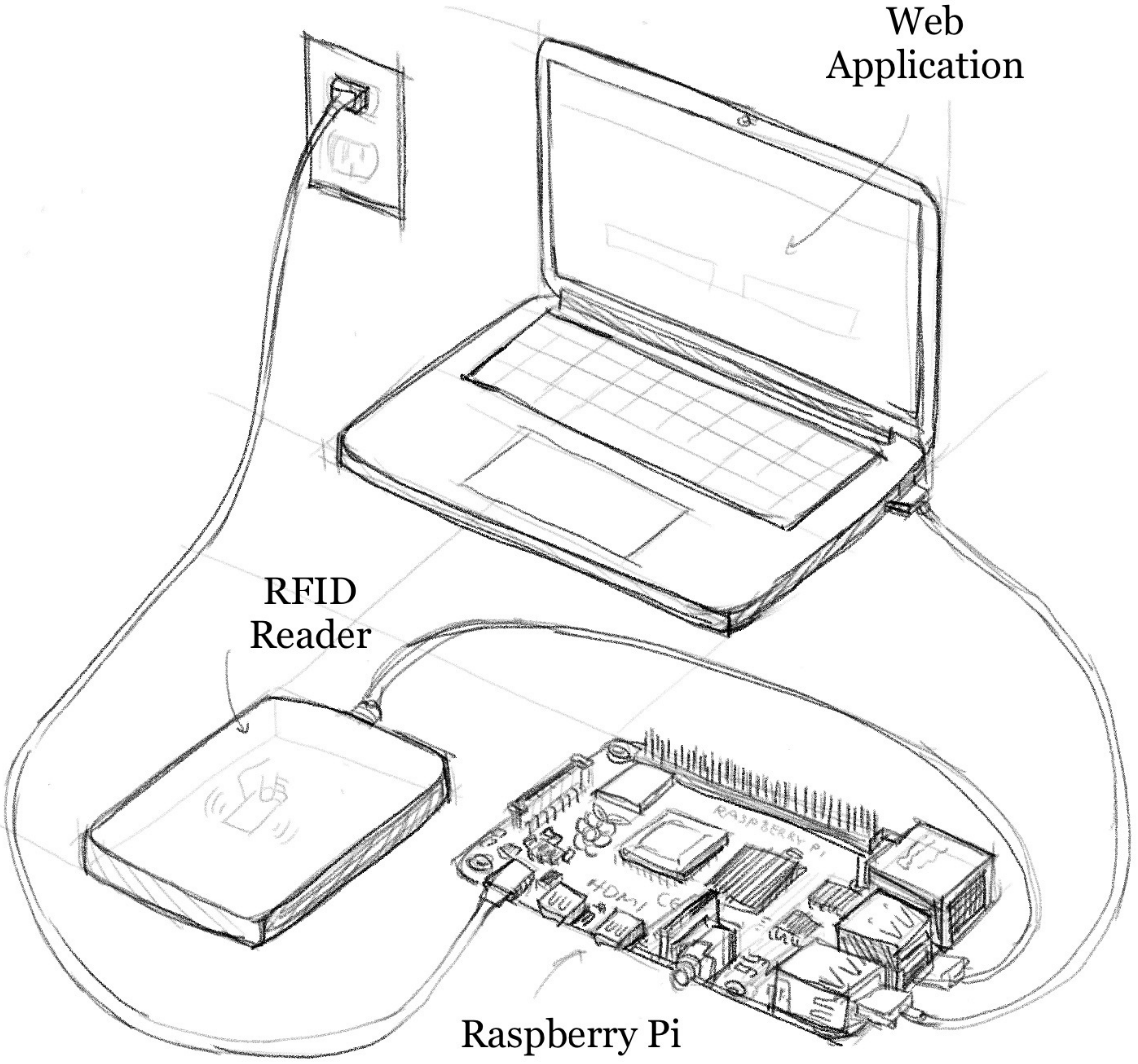


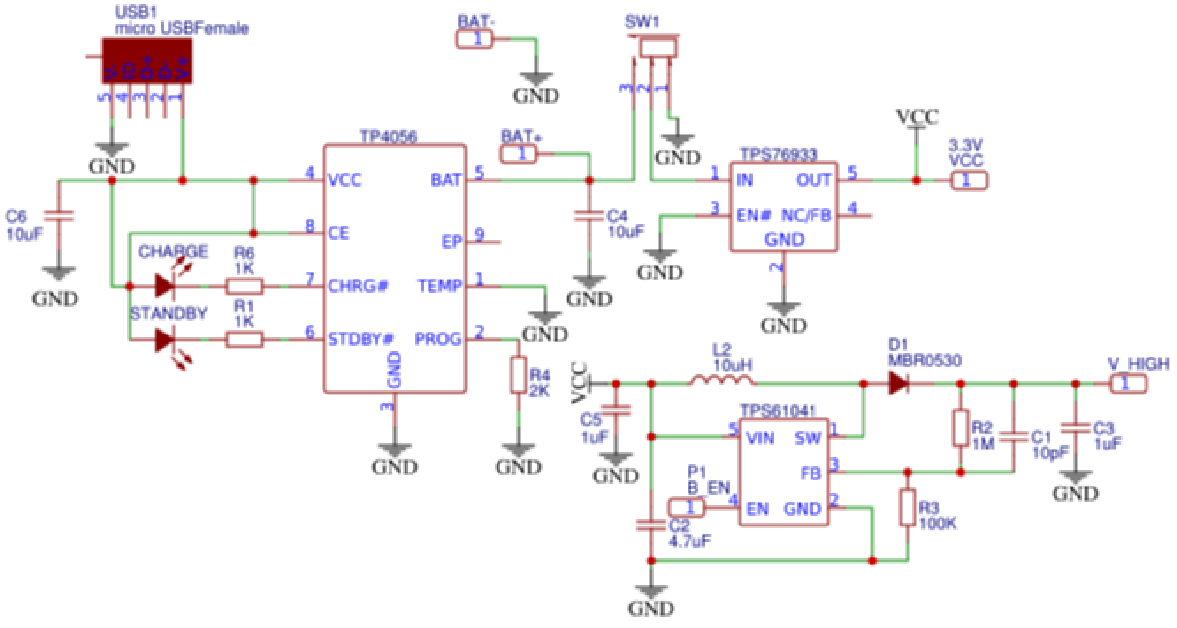
Figure 7 : Physical Reader Design

Another critical part of this project is reader's side, which is composed of a commonly found RFID reader, a Raspberry Pi, both powered by the wall, as well as web application that visually present the functionalities. RFID Reader will read our tag and the data will be transmitted to our web application.

2.4 Subsystem Overview

2.4.1 Tag System

Power Supply:

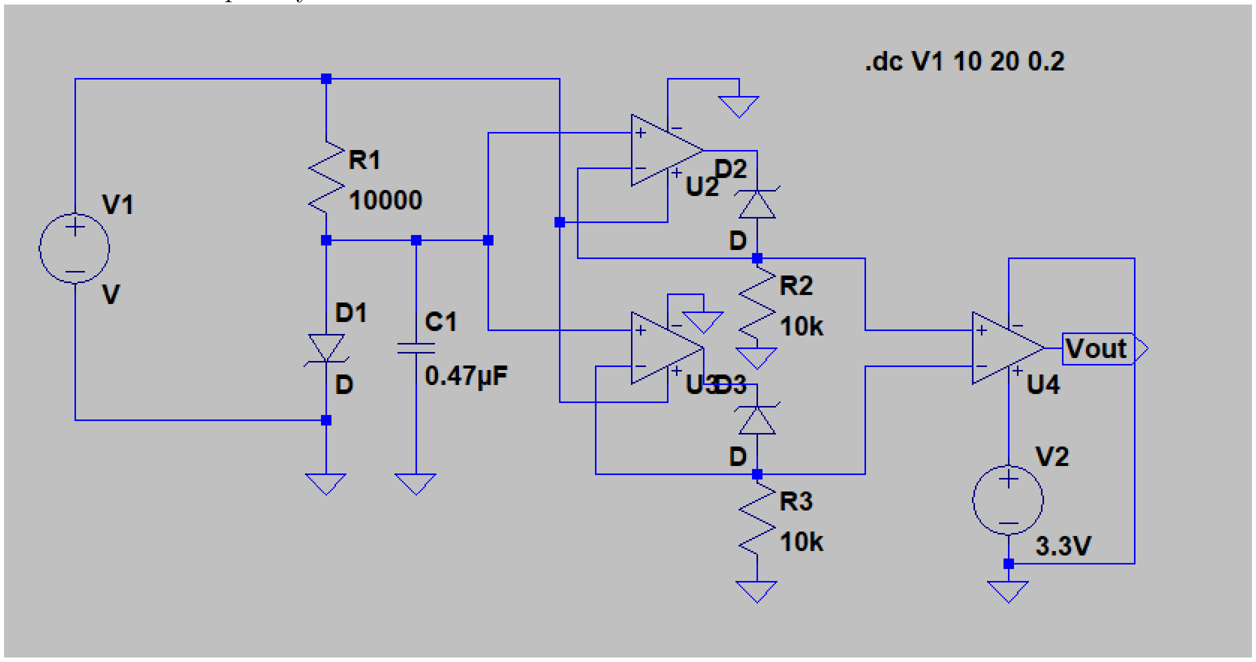


While not shown in the schematic, the ports BAT+ and BAT- will be connected to the battery. We will be charging our battery using USB. In addition, we will be including a toggle switch to allow the user to turn the tag off. The power supply has to supply around 12.5-14V and 3.3 V. In order to save power it will receive a signal from the main microcontroller on whether or not is should power the random number generator. In the schematic, the TPS76933 is a linear regulator with a low drop out voltage, the TP4056 is a charge controller and the TPS61041 is a boost converter. The port B_EN refers to "Boost Enable." Furthermore, all ports will be connected to rest of the tag PCB via pin headers.

Requirement	Verification
1. When the battery cannot supply sufficient power to power the random number generator, it must shut off.	1. First model the battery with an constant voltage power supply, which is initially set a 3.7V. Attach various resistive loads across the 3.3V and 14V terminals. Monitor the voltage across the 14V terminal and measure it shuts off when it dips to the 12.5V. 2. Next perform a similar test, except with the battery.

Random Number Generator

Our random number generator provides random numbers to be used to create blinding factors. We intend on using the randomness of breakdown current in a Zener diode. This will fed into a 1 bit Analog to Digital Converter and sampled by our microcontroller.



Requirement	Verification
1. Must produce statistically random numbers. 2. Must have a data rate of 100 kbps.	1. Have the random number generator pass the Duke Universities Diehard Test, which is a trusted source to test if random number generators are statistically random. We plan on running the test on a Raspberry Pi. The Diehard Test expects random numbers in the form of 32 bits and so we will write a program that returns 32 bits sampled from our random number generator. 2. Start sampling the random number generator at 100 Hz and measure serial correlation of the output. Gradually increase the sampling frequency until the magnitude of the serial correlation exceeds 0.05.

Microcontroller:

Our microcontroller is responsible for sampling bits from the Random Number Generator, performing

Elliptic Curve operations, encoding our responses to the reader and sending information to the transponder to be sent to the reader. We plan on using a Reed-Solomon error correcting code as we can share some data primitives (such as finite fields) with our Elliptic Curve portion. We will not be implementing our own Elliptic Curve as such implementations are tricky and subject to numerous side channel attacks.

Requirement	Verification
1. We would like our circuit to consume as little power as possible, especially when the tag is at idle. In order to do this, we need it to shut off non essential components, when the tag isn't in use. As a goal we would like to get power consumption to be under 10 mW.	1. To verify this, we will allow the power to idle and measure the power consumption from an external power supply.

RFID Transponder:
The RFID transponder is responsible for transmitting data to the reader and receiving data from the reader. To accomplish this, we will be use a nRF24L01+. In order to keep the power draw low in idle, it will also send an interrupt signal to the microcontroller to wake it up.

Requirement	Verification
1. We would like to transmit our protocol as quickly as possible and so it should be transmit data at a rate of at least a net data rate 1 Mbps (data sent - ECC). 2. The data sent should resistant toward corruption.	1. To test this, we will set up the transponder and receive so that it will send an ACK message whenever it receives a packet and won't send another packet until it receives an ACK message back. We will hold the transponder roughly 5cm away from the reader and continuously send messages of "Hello World!". 2. While testing for the able, we will collect the sampled "Hello World!". This will be considered sufficient if we are able to receive 1000 messages in a row without any corruptions.

2.4.2 Reader System

RFID Reader:

We would like to use a third party RFID Reader to transmit and receive the proofs from the tag. It enables contactless communications with the tag.

Requirement	Verification
1. It reads the RFID tag. 2. It has to support Error Correction Codes.	1. (a) Upon Tag being scanned, the reader authenticates user and sends information to web app. If web app displays information, then authentication is successful (b) The LED hooked up to the LED output lights up, which indicates that the reader scanned a tag

Raspberry Pi:

It is responsible for interacting with the transmitter and has to host our web application.

2.4.3 Web Application

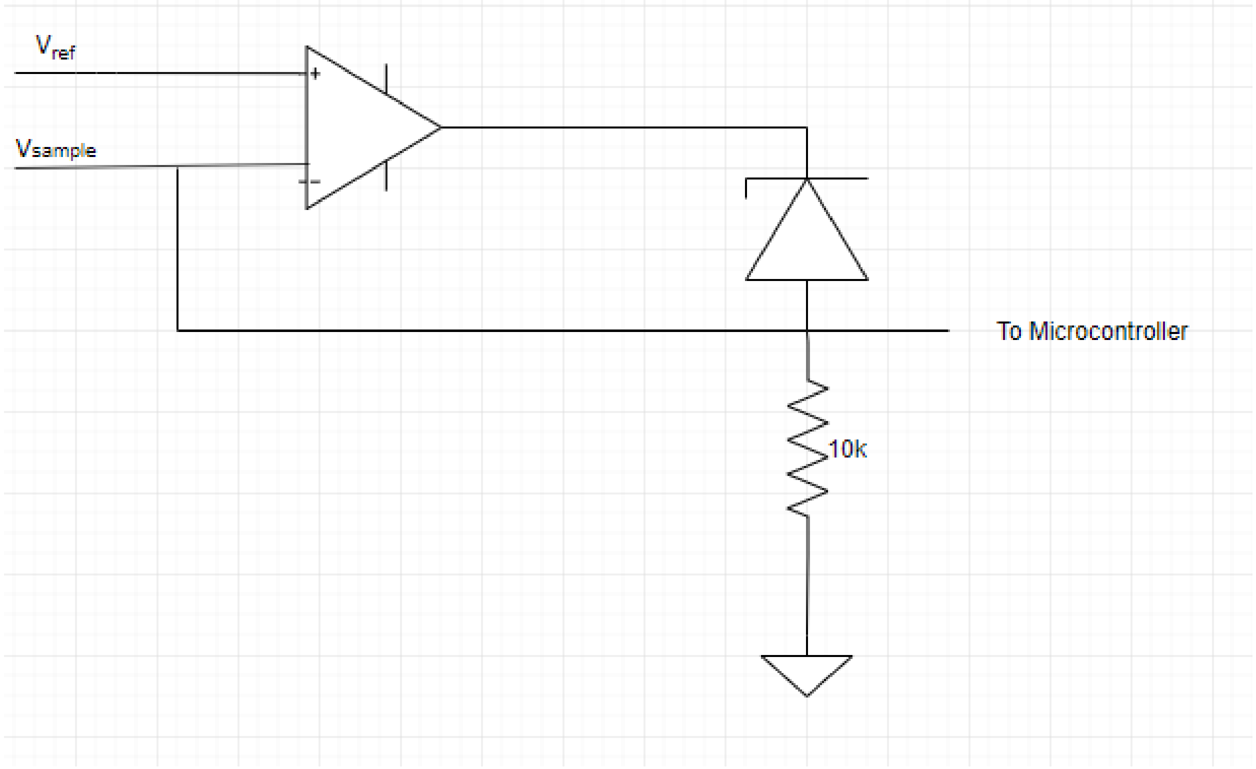
We would also create a simple web application as an user interface. Our goal is to better visualize all the functionalities of the tag. We will using Django and Python to build the web application. User will be able to see list of doors, whether they have access or not and request for access, while admin will grant or revoke user access or add new doors to the list.

Requirement	Verification
1. The web application would be able to indicate if a user has been successfully authenticated. 2. Additionally the interface would allow users to request for access whiles admin to add new users and revoke access.	1. (a) Have the web application receive data from any sensor and display it until the RFID reader and tag system are setup (b) Have the web app receive data from the RFID reader system 2. (a) Test if Tag gets rejected after revoking access (b) Test if Tag gets accepted after adding new user

3 Tolerance Analysis

Our random number generator poses some risk to our project, namely that it can be difficult to pass 3rd party tests. In addition, it is possible that outside disturbances such as temperature, can affect the quality of the bits that our random number generator outputs. Our protocol requires 255 bit random numbers, however for simplicity, we will round this up to 256 bits. According to the NIST [4], the number of bits that we will need to generate are $\frac{256}{entropy}$. If we take a conservative estimate and assume that each bit produces 0.7 bits of entropy, then we will need to sample 366 bits and then feed it into a cryptographically secure cipher such as salsa20.

Another possible point of contention is that it might seem that our random number generator is sensitive to hardware variances. To analyze our circuit, we will first simplify it as such.



Upon further inspection, we realize that this circuit resembles a closed-loop amplifier. The input impedance on a typical op-amp is on the order of 1MΩ 10TΩ, which is many orders of magnitudes greater than than our $10k\Omega$ resistor. Thus it is a valid assumption to use an ideal op-amp model. With this $V_{ref} = V_{sample}$ and so it isn't hard to see that the average of value of V_{sample} would be V_{ref}. In order to this, we ran LTSpice simulations across a number of possible scenarios.

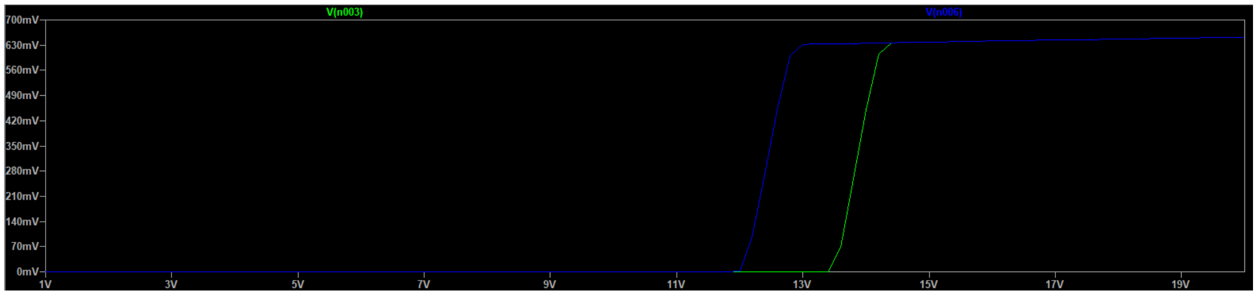


Figure 1: In this scenario, we ran our circuit with two completely different manufacturers and two different avalanche voltages. What we found was that so long as both diodes are in avalanche breakdown, our output is unbiased. In this image V(n003) (the output from the top diode) has a avalanche breakdown voltage of around 12V and V(n006) (the output from the bottom diode) has one of 13V.

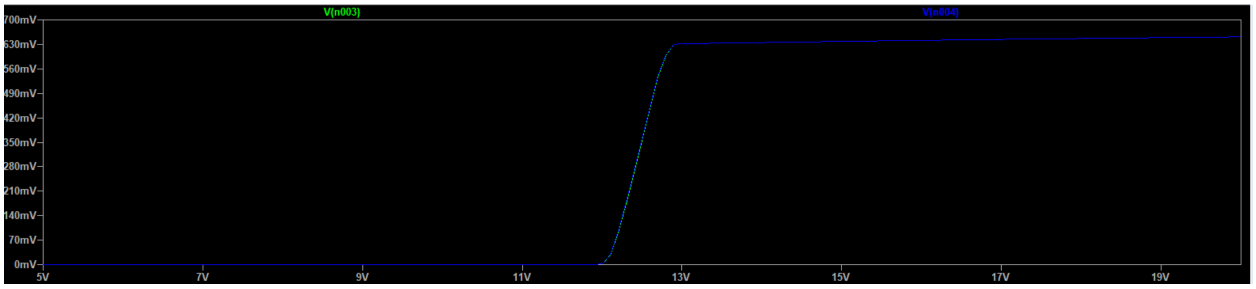


Figure 2: In this scenario, we gave different pull down resistors to each of the diodes. V(n003) has one with $10k\Omega$ and V(n006) has one with $8k\Omega$. This is definitely outside of the manufacturing tolerance for a resistor. In our simulation, we find that has no effect on our output. In all likelihood, having completely different resistances might affect our sampling rate, however LTSpice is not able to simulate the quantum effects of a Zener diode.

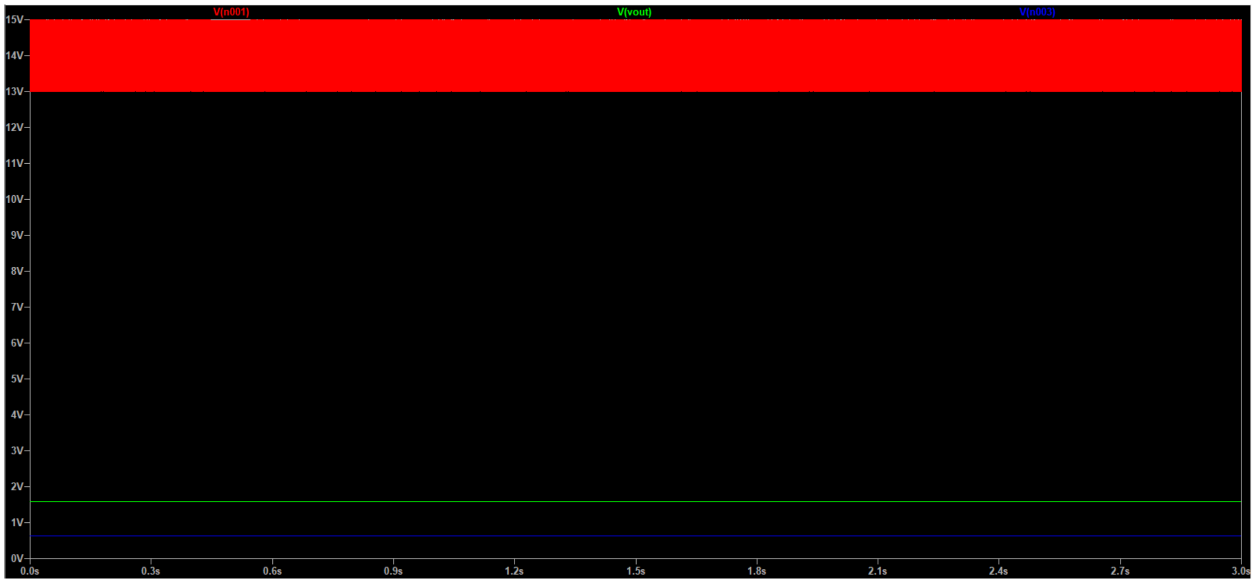


Figure 3: In this test, we simulation an extremely noisy power supply with a mean of 14V and an amplitude of 1V and a frequency of 20kHz. V(n001) is our power supply. V(n002) is the output from the top diode and V(n001) is the reference voltage being fed into our diode circuit. As we see, as long as we are in avalanche breakdown, there is no effect on our output.

4 Ethics

In regards to safety, the issue that may arise pertains with our use of a lithium battery. If not properly understood when it comes to its implementation, fires and other accidents become likely risks to occur and would pose danger to 'the safety, health, and welfare of the public' [6]. In order to mitigate these risks, proper practices must take place. First, the specifications of the battery must be thoroughly looked over and understood, so when it comes to implementing it in conjunction with the rest of the hardware, there is no overload or improper design such as short circuiting or placing near an area of high temperature that could cause the battery to catch fire and result in a hazardous situation. According to the University of Washington, certain practices such as proper procurement, storage, charging practice, and disposal should be taken into account [7]. For example, acquire the battery from a trusted source, place them away from flammable materials, disconnect batteries that exhibit any unusual behavior such as heating up or releasing some sort of smell, and dispose of batteries safely by taking them to designated facilities.

In order to avoid any potential FCC violations and potential licensing issues, our RFID transmitter and receiver will operation in the 2.4 GHz, which is part of the ISM band [8].

Curve22519 is under public domain and so we don't have to worry about patent infringement.

We understand the risks and 'societal implications of conventional and emerging technologies,' which is why we are creating this product. As a result, our top priority is to protect the user's private information [6].

5 Cost

Make sure that any tables of costs are numbered, given titles, and cited directly in the text.

5.1 Parts

Part	Cost
Voltage Regulator	$\$0.33 * (10) = \$3.3$
PCBs (JLCPCB)	$2
Microcontroller (MSP430)	$2.53
RFID Transponder + Reader	$2 * \$3.36 = \$6.72$
Raspberry Pi 3 Kit	$49.99
Assorted RLC	$10.00
Battery Samsung 25R	$2.99
BMS	$1.19
T1561041 Boost	$1.36
ATtiny	$1.23
Total	$81.53

5.2 Labor

The average starting salary of a ECE graduate from University of Illinois at Urbana-Champaign is $96,518 as of 2016-2017. We have three people working on this project and we estimate that we will work about 10 hours per week for 16 weeks. Under these assumptions, the total cost of labor is shown in Equation 3.2.2:

$$\frac{\$96,518}{1yr} * \frac{1yr}{2080\ hrs} = \$46.40/hr$$
$$\frac{\$46.40}{1hr} * 3 * 2.5 * 80\ hrs = \$27,840$$

For a combined cost of $27,931.53.

6 Schedule

Week	Joseph	Lilan	Majdi
9/30	Design Document	Design Document	Design Document
10/7	Order Parts	Begin design of Web App	Tutorials on RF circuits
10/14	Begin PCB	Programming and data transmission research	Check PCB & RF transmission research
10/21	Solder PCB & build power circuit	Have a prototype ready for web app	Build RF tag and validate
10/28	Program ECC onto Microcontroller and validate library	Have web app accept generic input from sensor	Build RF reader and validate
11/4	Integrate Microcontroller plus random number generator with rest of project	Have web app accept transmission from RF reader	Validate Tag and Reader system work and transmit to web app
11/11	Preparing for Mock demo and presentation	Preparing for Mock demo and presentation	Preparing for Mock demo and presentation
11/18	Review of project and handling of defects/improvements	Review of project and handling of defects/improvements	Review of project and handling of defects/improvements
11/25	Testing and Validation	Debugging and Testing Web App	Testing and Validation
12/2	Demo Prep	Demo Prep	Demo Prep
12/9	Final Paper	Final Paper	Final Paper

References

[1] N. Courtois, “Card-only attacks on mifare classic,” University College London.

[2] Z. Liu, J. Großschädl, L. Li, and Q. Xu, “Energy-efficient elliptic curve cryptography for msp430-based wireless sensor nodes,” in *Information Security and Privacy*, J. K. Liu and R. Steinfeld, Eds. Cham: Springer International Publishing, 2016, pp. 94–112.

[3] “MSP430F552x, MSP430F551x Mixed-Signal Microcontrollers,” Texas Instruments, Dallas, TX, 2008.

[4] B. Lampert, R. S. Wahby, S. Leonard, and P. Levis, “Robust, low-cost, auditable random number generation for embedded system security,” in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, ser. SenSys ’16. New York, NY, USA: ACM, 2016. [Online]. Available: http://doi.acm.org/10.1145/2994551.2994568 pp. 16–27.

[5] J. Hermans, A. Pashalidis, F. Vercauteren, and B. Preneel, “A new rfid privacy model,” in *ESORICS*, 2011.

[6] “IEEE IEEE Code of Ethics,” 2019. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[7] “Stay safe when using lithium batteries,” April 2018. [Online]. Available: https://www.ehs.washington.edu/about/latest-news/stay-safe-when-using-lithium-batteries

[8] T. Mooring, “FCC Allocation History File,” Federal Communications Commission Office of Engineering and Technology Policy and Rules Division, May 2019.

Appendix A Proof of Requirement 1

Requirement 1 can be rewritten as, $g^s = Ag^{d'}T^c$. Now we need to show that this shows true for all users.

Lemma A.1. *The value d and d' are equivalent so long so as either the value t (blinding factor generated by the tag) or the value y (secret key of the reader) is known.*

Proof. Recall that $d = [B^t]_x$ and $d' = [T^b]_x$, where $[]_x$ is the x coordinate of the elliptic curve point represented by the value in brackets. Substituting values in, we get $d = [g^{bt}]_x$ and $d' = [g^{tb}]_x$ and thus we see that the values are equivalent. As we can see knowledge of either t or b is required to construct d. □

Proof. Let s represent the responsive given by the tag. We need to show $g^s = Ag^{d'}T^c$ hold true.

$$\begin{aligned}
g^s &= Ag^{d'}T^c \\
&= (g^a)g^{d'}g^{tc} && \text{(substituting variables)} \\
&= (g^a)g^{d}g^{tc} && \text{(from lemma a.1)} \\
&= g^{a+d+tc} && \text{(the tag's response is s = a + d + t c)} \\
&= g^s
\end{aligned}$$

□