Hardware Security Module

ECE 445 Design Document
Team 63
Frankie Papa, Calvin Fisher, Nick Schiesl
TA: Evan Widlowski
2/23/2020

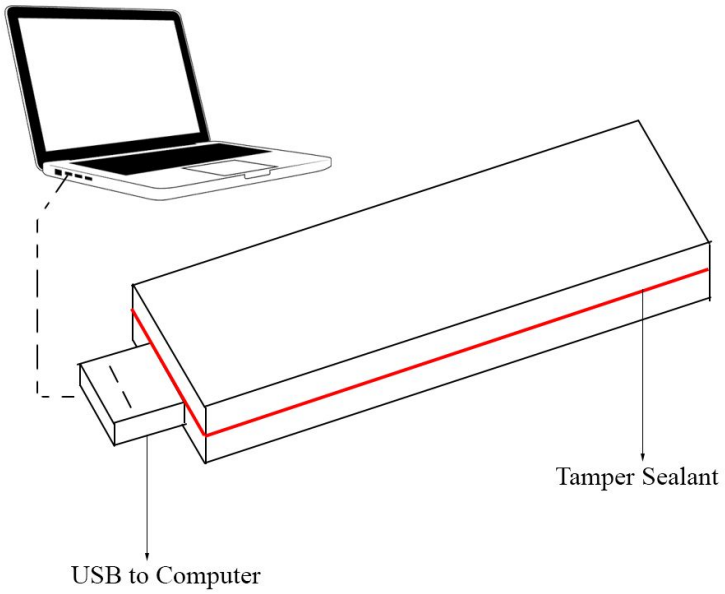
1 Introduction

1.1 Problem and Solution

Hardware Security Modules (HSMs) are devices that have a protected keystore coupled with crypto hardware that provides features such as encryption/decryption or random number generation [2]. In practice, these products can be very expensive and may not be able to do everything that is desired. For example, one of our members faced an issue while working with the Trusted Platform Module (TPM), a cheap and standard HSM in many personal computers, where he was not able to persist symmetric keys (a feature desired in the project he was working on). These devices can also be very costly ranging from low cost options at around $110.00 to high cost units at around $32,000 [4]. The problem that we are looking to solve is that of finding a low-cost, high-security option. A typical low cost HSM has limited key storage space and limited support, or no support, to symmetric algorithms [4]. Our solution to this is to fill this gap by producing a cheaper HSM that has the ability to store a large amount of keys, provides symmetric encryption algorithms, and has a random number generator with high entropy.

In order to provide a more secure product at this lower price we will make sure that it fulfills the main requirements for FIPS 140-2 Level 3 with room for additions later to fully fulfill each requirement. In NIST's documentation on FIPS 140-2 Level 3 the main features that we are implementing would be at least one approved algorithm/security function (AES Encryption), the use of tamper-evident coatings or seals, and systems in place which make it more difficult to gain access to the modules inside as well as the zeroization of keys in the event that the cryptographic module is opened [5]. We have found that many of the higher end HSMs that are FIPS 140-2 Level 3 or 4 certified and have higher performance are priced near $1000.00 or more and we aim to provide a device capable of some of their features at a fraction of the price [2].

1.2 Visual Aid



1.3 High-Level Requirements

1. The device must be able to store thousands of AES-128 keys in nonvolatile memory
2. The device must be able to encrypt and decrypt files as large as 1GB
3. The random number generator on the device can generate high entropy AES-128 keys
4. Tamper evidence module must be able to zeroize the keystore in the event of physical tampering

2 Design

2.1 Block Diagram

In order for our design to be successful our device is going to need a power supply, a random number generator, a tamper evidence module, and a control module (which doubles as the encryption module). With the parts that we have chosen our power supply will come from the USB connector which has a voltage rating of 30V. The voltage regulator will then supply 5V to each component and 18V to the RNG circuit. The flash memory has 2GB storage which allows us to store a lot of 128-bit keys. The tamper evidence buttons and the conductive wire mesh sensor will alert the microcontroller of tampering and wipe the keys from the keystore. The RNG circuit will generate AES-128 bit keys on command from the microcontroller which will store them in the flash memory. Lastly, inside of the control module the microcontroller will route all data to each piece. Doubling as the encryption module, the microcontroller will also use AES encryption to encrypt or decrypt data sent in through the USB connector using a designated key from the keystore.

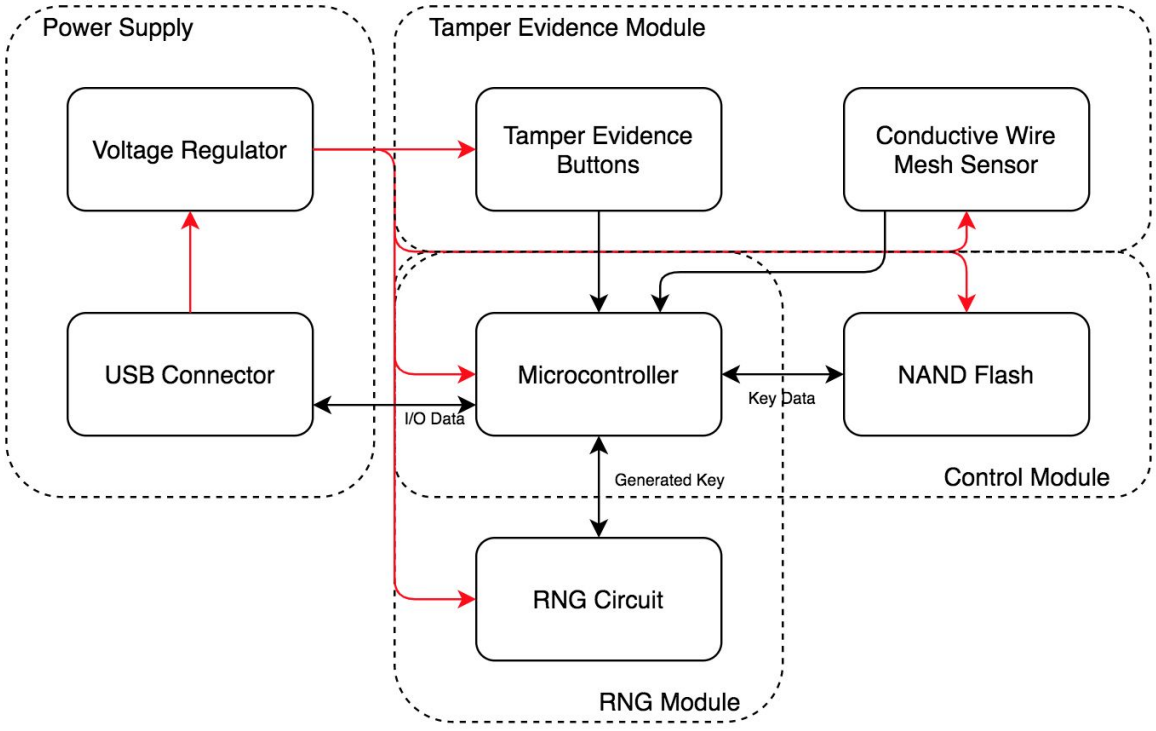


Figure 1. Block Diagram

2.2 Power Supply

The main source of power for our device will be through a USB power supply. This power supply will be regulated by a 5V and 3.3V regulator that will supply the correct voltage to the rest of the device's components.

2.2.1 USB Connector

The USB Connector will be the main power source for our device with an output of 30 volts. This higher voltage will allow us to power all of our devices and supply enough current for all of the components.

2.2.2 5 Volt Regulator

The 5 volt regulator will create a 5 volt output from the USB connector. This output will be used to power the random number generator module and other components of our device.

2.2.3 3.3 Volt Regulator

The 3.3 volt regulator will generate a 3.3 volt output from the 5 volt bus output from the 5 volt regulator. This output will be used to provide the correct voltage to our microcontroller and flash memory components. It will also be used to supply power to several other components in our device.

2.3 Control Module

The control module (or the encryption module) is in charge of connecting all of the pieces of our device as well as encrypting data transmitted from the USB connector. At the core, the microcontroller will be programmed to encrypt data, wipe data in case of tampering, and store input or generated keys.

2.3.1 Microcontroller

We have chosen a STM32F7 series microcontroller from STMicroelectronics for our device because we wanted a high performance, programmable controller which can be used to route all incoming data from the USB connector as well as utilize a CBC AES encryption algorithm to encrypt or decrypt incoming data. The microcontroller chosen has a speed of 216MHz which should give us high speed encryption capabilities. It will also be constantly polling the tamper evidence module which will alert the controller when to zeroize the NAND Flash memory. We will be referring to the microcontroller's programming manual to write software to encrypt/decrypt data.

2.3.2 NAND Flash

The Winbond Flash Memory that we chose has 2GB of memory which technically allows us to store 125,000,000 128-bit keys. This is likely smaller due to software preloaded on the chip, but we are only looking for the ability to store thousands of keys.

2.3.3 USB Connector

The USB Connector powers and interfaces with the microcontroller. This will allow us to send data to the microcontroller which can be used in multiple subsystems of our device. We will likely have command codes which preface our data which tell the microcontroller whether we are looking to encrypt data, decrypt data, or store a new key. These parameters will be fed into our program which will then do the desired computations or command.

2.4 RNG Module

The HRNG or Hardware Random Number Generator is essential for this encryption device. We are hypothetically creating a hardware security device for the market, so security is top priority. The HRNG will be a big step up from the PRNG or pseudo random number generator that most programs use. In a PRNG a "seed" is passed through a black box algorithm which spits out a "random" number. Obtaining the seed used makes the random number obsolete. In our case, obtaining a seed means obtaining the AES-128 bit key for an experienced hacker. Our HRNG will not use a seed and will instead use digitized transistor noise to generate random bits.

In our HRNG a reverse biased or negative voltage applied transistor will generate noise. This noise will then be amplified by 2 more resistors. The noise is then "digitized" by a Schmitt Trigger Inverter. We now have a readable output of truly random bits.

[Circuit Schematic Here]

2.4.1 Noise Generator

2.4.2 Signal Digitizer

2.4.3 State Selector

2.4.4 Requirements & Verifications

2.5 Tamper Evidence Module

3 Cost and Schedule

3.1 Cost Analysis

When determining the cost for development we assume an average salary of $80,000 per person of our 3 member team. We also assume 10 hours of work per week and a 16 week timeframe for completing the project. Using this we calculated the development cost for labor as follows.

$$\frac{\$80000}{year} \times \frac{1\ year}{52\ weeks} \times \frac{1\ week}{40\ hours} \times 3\ people \times 16\ weeks \times 10\ hours \times 2.5 = \$46,153.85 \quad (1)$$

The table below details the estimated cost for parts:

Part Name	Manufacturer	Part #	Quantity	Cost/part
Microcontroller	STMicroelectronics	STM32F730R8T6	1	$4.96
NAND Flash	Winbond	W29N029VSIAA	1	$3.82
USB Connector	TE Connectivity	1734035-4	1	$1.46
Buttons	Omron Electronics	B3FS-1000P	4	$0.65
Assorted Resistors, Capacitors, ICs,	Digikey	N/A	1	~$10.00

etc.				
PCB	PCBway	N/A	1	~$5.00
Plastic Housing	N/A	N/A	1	~$10.00
Total Cost	-	-	-	$35.89

Assuming that we will be making three prototypes for testing and demoing, the cost for parts will amount to $107.67. Adding this to the cost of labor gives us a total cost of $46,261.52.

3.2 Schedule

Week	**Frankie**	**Calvin**	**Nick**
3/2/2020	Begin CBC AES algorithm design	Begin RNG circuit design	Begin tamper evidence module design
3/9/2020	Program microcontroller and test algorithm	Assemble circuit and begin testing	Order parts and begin assembling circuit
3/23/2020	Design PCB and design key storage abilities	Design PCB and verify entropy	Test circuit and design PCB
3/30/2020	Assemble circuit and begin testing of the encryption module	Assemble circuit on PCB	Assemble circuit on PCB
4/6/2020	Refine encryption module	Test and refine PCB	Test and refine PCB
4/13/2020	Assemble prototype	Assemble prototype	Assemble prototype
4/20/2020	Mock demo and refine prototype	Mock demo and refine prototype	Mock demo and refine prototype
4/27/2020	Demo and begin final paper	Demo and begin final paper	Demo and begin final paper
5/4/2020	Final presentation	Final presentation	Final presentation

4 Ethics and Safety

While designing our product we have to keep in mind the ethics involved in producing a HSM with stringent security requirements. It would be highly unethical to advertise a product that is supposed to

fulfill FIPS 140-2 level 3 requirements, but then fails in some aspect (perhaps some other way of tampering with our device). It would also be unethical for us to produce a faulty product; this requires us to create and administer strict tests to our device in order to assure to our customers that we have a working product.

Working in the senior design lab will require us to follow strict safety measures to prevent any injury. Using soldering irons or hot air can put us at risk of burning ourselves.To prevent this from happening will follow the correct safety precautions that were explained in our lab safety training and our soldering assignment. We will always power these devices off while not in use and use correct soldering techniques to prevent any risk of injury. Another hazard would be when we power our device with the lab power supplies. We will follow all safety guidelines to prevent risk of electrical shock or electrical shorts.

5 Citations

[1] S. W. Smith, "Hardware Security Modules," *Dartmouth*, 2010. [Online]. Available: https://www.cs.dartmouth.edu/~sws/pubs/hsm-draft.pdf. [Accessed: 24-Feb-2020].

[2] J. Schlyter, "Hardware Security Modules," *internetstiftelsen.se*. [Online]. Available: https://internetstiftelsen.se/docs/hsm-20090529.pdf. [Accessed: 24-Feb-2020].

[3] "TestU01," *Empirical Testing of Random Number Generators*. [Online]. Available: http://simul.iro.umontreal.ca/testu01/tu01.html. [Accessed: 24-Feb-2020].

[4] S. Dickinson, "HSM Buyers' Guide - Documentation Reference Material," *OpenDNSSEC*. [Online]. Available: https://wiki.opendnssec.org/display/DOCREF/HSM Buyers' Guide. [Accessed: 24-Feb-2020].

[5] D. L. Evans, P. J. Bond, and A. L. Bement, "FIPS PUB 140-2," 12-Mar-2002. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf. [Accessed: 24-Feb-2020].

[6] C. Platt and A. Logue, "Really, Really Random Number Generator: Make:" *Make*, 01-May-2015. [Online]. Available: https://makezine.com/projects/really-really-random-number-generator/. [Accessed: 24-Feb-2020].

WEBSITES:

https://www.cs.dartmouth.edu/~sws/pubs/hsm-draft.pdf HSM PAPER
https://internetstiftelsen.se/docs/hsm-20090529.pdf HSM COST+BASICS
http://simul.iro.umontreal.ca/testu01/tu01.html RNG ENTROPY
https://wiki.opendnssec.org/display/DOCREF/HSM+Buyers%27+Guide HSM BUYER'S GUIDE
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf FIPS 140-2 REQS
https://makezine.com/projects/really-really-random-number-generator/ DIGITIZED INVERTER RNG FROM AMPLIFIED TRANSISTORS