STRATH NAME

DOCTORAL THESIS

Thesis Title

Author:
John SMITH
Examiner:
Dr Man Page

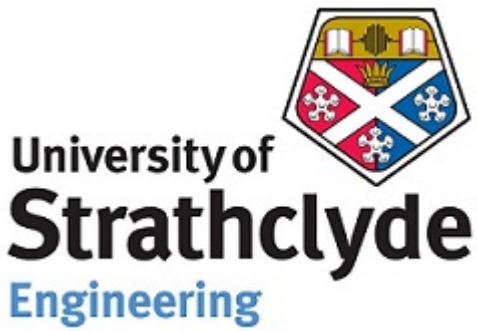
Supervisor:
Dr. James SMITH
Phd Student:
Greg

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

March 12, 2016



Declaration of Authorship

I, John SMITH, declare that this thesis titled, "Thesis Title" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

STRATH NAME

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

Thesis Title

by John SMITH

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too…

Contents

Chapter 1

Introduction

The Internet of Things or IoT is the concept of a huge network of physical objects connected and communicating to themselves and to the world wide web. Devices can include domestic appliances, buildings, cars. As it becomes a rapidly growing concept with over 50 million devices expected to be connected to the web by 2020, (need ref) the security of the transmissions of these devices is becoming a more and more pressing issue. IoT's main benefits are the remote control of devices and appliances, for the device to have the ability to send information about it's state, such as a vending machine reporting that it has run out of a certain item, and to allow the machines to be more automated and to work with other machines, like a home hub device that can turn on the lights and central heating when an occupant is arriving home, with the lights and heating not being connected to each other but to the central hub.

However IoT will be ultimately be useless if it is unsecure. IoT is an emerging field but there have already been some high profile security disasters. Ranging from relatively less serious problems such as some "hackers" been able to glean important wifi information from your internet connected lights and assuming control. There have been three examples of this with a Jeep Cherokee, Toyota Prius and Tesla. The hackers were able to control the accelerator, door locks and brakes, among other things. This highlights a very real problem that will only become more important. Too often security is an afterthought but it really needs to be built into products from the offset.

Within the last three years there have been three high profile security breaches on commercial cars, one on a Cheroke Jeep [1], a Toyota Prius [3] and a tesla [2]

With that in mind the subject of this report is the secure transmission of a users private home temperature data. If they have a system that monitors the temperature of all the rooms, that data can be used to figure out when they are likely to be home or not. So, using an Arduino Due as the base station that talks to the temperature sensors throughout the house, it takes the sensor data signs then encrypts it and sends it using an Ethernet Shield to a remote server.

Chapter 2

Background

more detail description of IoT?

2.1 Cryptography

Cryptography is the practise and study of techniques for secure communication in the presence of attackers. To do so, one can use encryption where by messages are encoded in such a way that only authorised parties or at least parties in possession of the keys can view them. There are two main ways of encryption Symmetric Key encryption and Public Key encryption. In Symmetric Key encryption both parties have the same key the which can encrypt and decrypt messages that are sent between them. The problem is that if Bob wants to send an encrypted message to Alice, he must get the secret key to her. Currently the most secure way for the transmission of secret keys is to hand them over in person, in private. This this project Asymmetric Key encryption and Digital signatures.

2.1.1 Asymmetric Key Encryption

To get round the problem of securing sending secret keys, one can use Asymmetric Key encryption has a secret key and a public key, the public key is generated out of the secret key and are therefore mathematically linked but it is computationally infeasible to calculate the secret key from the public key. This public key can be given out freely and is not a secret. So if Bob sends a message to Alice he encrypts the message with her private key and she can decrypt it with her secret key. This type of key encryption gets past the sharing key problem but it only stops attackers from reading the message. It does not prove the message was sent by a certain person or that the message has not been altered in transit.

2.1.2 Digital Signature

This is a mathematical scheme for proving message authenticity, message integrity and message non-repudation. Similarly to asymmetric key encryption a random private key is created with a corresponding public key. Double check."!! The algorithm takes in a message and a private key and using SHA-512 and? it produces a signature. If Alice signs a message in this way, Bob can use another algorithm to verify the message with the public key and signature.

2.1.3 TweetNaCl

NaCl or "Salt" is a simple to use high-speed library for authenticated encryption. it provides both Asymmetric and Symmetric encryption, It provides authentication and message integrity with SHA-512. The authors are Daniel J. Bernstein, Tanja Lange and Peter Schwabe but at points it relys on third part implementations for parts. The API is simple, having only a handful of methods but uses high speed, high security primitives. /refhttps://labs.opendns.com/2013/03/06/announcing-sodium-a-new-cryptographic-library/

Unfortunately the library wouldn't work completely on a Arduino, one problem is that there is no /dev/random and therefore no randombytes() which means that it can't create keypairs in the usual way. Also, as mentioned later Arduino can't use the C library as is, it needs to be converted into C++. Another problem is that the library is relatively quite large, the Arduino Due has at it's disposal 512KB flash memory and the full library is 3MB. Fortunately the same creators along with Bernard van Gastel, Wesley Janssen and Sjaak Smetsers made TweetNaCl. Which is a tiny implementation of NaCl, still providing speed and security but with a significantly smaller code size 40KB. It retains the same protections against (from tweetNack-201409..) timing attacks, cache-timing attacks, has to branches depending on secret data and no array indices depending on secret data. In addition it is thread-safe and has no dynamic memory allocation. It is portable and easy to integrate, the library is easily added as it consists of two files, there is no complicated configuration to be set up or any dependencies on external libraries. Because of this compactness it is easier to read and understand it's operation. Although not as fast as NaCl it is still fast enough for most applications. "Most applications can tolerate the 4.2 million cycles that OpenSSL uses on an Ivy Bridge CPU for RSA-2048 decryption, for example, so they can certainly tolerate the 2.5 million cycles that TweetNaCl uses for higher-security decryption (Curve25519)." TweetNaCl is still small after compilation at 11KB thus avoiding instruction cache misses. It is a full library and not a set of isolated functions for a simple NaCl application, only six functions are needed. crypto_box for public-key authenticated encryption; crypto_box_open for verification and decryption; crypto_box_keypair to create a public key; and similarly for signatures crypto_sign, crypto_sign_open, and crypto_sign_keypair. It is open source and the developers encourage it to be used as much as possible.

NaCl will move to Ed25519 signature system, what is that?

TweetNaCl encrypts messages by xor'ing them with the output of Bernsteins Salsa20?? stream cipher

NaCl crypto_stream uses Bernsteins X

Again, TweetNaCl uses SHA-512 as it's hash function with the Ed25519 signature scheme and the code is simplified compared to the NaCl implementation

For asymmetric cryptography TweetNaCl uses Bernsteins' Ed25519 elliptic curve Diffie-Hellman key exchange??

2.2 Types of attacks

2.2.1 Replay Attack

When this attack occurs the attacker replays a valid message. If Bob wants Alice to prove who she and she duly provides some encrypted signature to prove so. Eve can capture that signature, She does not know what the signature is but she knows that it is a signature. She can then connect to Bob and use this message to pretend she is Alice. To prevent this attack, use an identifier that is only valid for one use, this can be session tokens or one-time passwords.

2.2.2 Man in the Middle Attack

In this attack there is an attack between two parties, Bob and Alice, who wish to communicate. The man in the middle, Eve, changes messages as they are in transit and manages to pretend that she is the person that the other thinks they are talking to. An example is if Alice asks for Bob's public key, Eve can capture that public key, replace it with her own and send that and because Alice has no way to prove that it is Bob's key or not she accepts it. So when Alice sends a message that has been encrypted with what she thinks is Bob's key, Eve can take it, decrypt it with her key, change the message then encrypt it with Bob's real public key. Which Bob receives and believes the message is from Alice.

2.2.3 Bit-Flipping Attack

This is where the attacker can change the cipher text in some way that cause a predictable change in the plain text. The attacker does not know exactly what the plain text is. If Alice was to send a message to Bob saying that she owes him £100. If Eve knows the format of the message, she can change the number at the end into £1000.

2.2.4 Stream Cipher Attack

stream cipher attacks, if the same key is used chosen plain text attack
Side channel attack, does NaCl protect a bit against this

2.3 Machine to Machine

M2M refers to the direct communication between two devices using any sort of channel. This is a component of IoT when connected in this way small, low power sensors can transmit their data to another device which can collate, perform analysis or some extra computation or pass the data along again before it reaches a human user. Present day applications monioring the health of machinery, digital billboards or sensory networks. An application may multiple sensors in a network that pass sensor data to multiple nodes which do some computation, such as adjust machines to correct errors, replenish stock or manage a system and possibly sending information, such as state across the Internet to a human user

2.4 Technologies used

In this project the Arduino was programmed using C++. The C++ was developed in the Arduino IDE. An Arduino Due, Arduino UNO, DS1820S temperature sensor with resistor and two Ethernet Shield R2 boards. On the server side there is an SQL server, PHP scripts that accept that Arduino data and put it in the SQL server. And a Java web application that uses JBCD.. to access the SQL server and outputs dynamic HTML when accessed. The Java code was developed using Eclipse Jee Mars.

2.5 Secure Transimission of Keys

The problem is more acute with symmetric key encryption as it is the same key used to encrypt and decrypt a message, if the key is compromised messages can no longer be trusted. The only thing and most high tech solution is to meet in person and in private with the person you plan to communicate with and exchange keys there. If the communication with be of a diplomatic nature, the bag used to transport the keys can have special legal protection against being opened and this is called a diplomatic bag(pg 52 of book chpt 2). You could send a key over a another secure channel that you trust but there is always the chance that has been compromised and it is not possible to verify that the person on the other side of the secure channel is who you think it is without first having a secure identifier which you can't get solely through internet communication. There is a similar problem with Asymmetric key encryption, a user can freely distribute their public key and they can be pretty sure that the messages they receive are secure but it is not possible to prove who has sent you the message without their public key signature but initially public key transmission is vulnerable to man in the middle attacks.

Chapter 3

Design

3.1 Microcontroller

The first step in this application is the acquisition of data. For example data, a variable that is monitored for important applications in the real world was chosen, this being temperature. There are a range of temperature sensors out there but one that is compact, cheap, freely available, has a large temperature range, high precision and can derive power directly form the data line, parasite power, is the DS18S20. It returns a 9-BIT byte array so is ready to encrypt straight away.

pretty picture of course

The Arduino Due is a large Arduino, it is the first one with a 32-bit ARM core micrcontroller, has 54 digital I/O pins, 512KB flash memory, 96KB SRAM and a 84MHz. A lot of microcontrollers are 8-bit which means that they are limited in their cryptographic options. With a 32-BIT architecture and a large amount of flash memory it is possible to have a light weight cryptographic library for our application but the Due is still low cost enough that it can be used for an IoT sensor application. Because it is an Arduino it benefits from the large community, wide range of compatible components and large set of open source libraries.

Once the temperature data is on the board there needs to be a way to transmit the data, as the data is being sent over the internet the natural choice is an Ethernet or WiFi shield. Unfortunately although the WiFi shield would be better as wireless improves and becomes more the norm, the more IoT applications will use it, it is much more expensive than the Ethernet Shield. The first revision of the shield is no longer made so the second revision, R2 will be utilised. IT allows for easy connection for the Arduino to the Internet, it uses the Wiznet W5500 Ethernet chip, supports up to 8 different socket connections at a speed of 10/100Mb and has a MicroSD card slot to store network settings.

3.2 Security

There a lot of security options for desktop programmes and communications, examples pls and it can even be an after thought in this more establish...eh more ubiqitous field but in terms of microcontrollers only in recent times have they become powerful enough at a cheap price and therefore popular enough for developers to write or adapt cryptographic libraries suited for microntroller communications.

some have extra chips to do the encryption

Are 4-8 bit crupto librarues just not as good as 32 bit?

TweetNaCl is a full library for full functions, only two files, tiny memory and similar performance to proper shit

Chapter 4

Implementation

4.1 IoT Platform

The basic concept of this platform is an Arduino Due that takes the current temperature of the room from a DS1820 temperature sensor. Then that data is signed and encrypted with TweetNaCL before being transmitted, using an Ethernet Shield, across to an SQL server. A web application takes the SQL data decrypts, checks the signature is valid then displays on a website.

graphic here pls

Why was the Due chosen, 32 bit?

DS1820 is a lost cost temperature sensor that is very accurate, 12 bits of precision? and is also low power. It can scavenge power from the data with the arduino and thus does not need it's own power source.

For the prototype, an Ethernet Shield was used as it is much cheaper than a WiFi shield but ultimately completes the same job. The shield is a simple way to connect arduinos to the internet. The shield used was the second revision, R2 and has a w500 ethernet controller.

What are some of the options for the base station, Due/MSP430? And for the internet connection Ethernet?WiFi shield?

4.1.1 Temperature reading

The Due is used in two separate ways in this project; first is the client to the java web app and second is a demonstration of secure public key transmission with another Arduino Uno and Ethernet shield. The first takes the raw hex values from the DS18S20 temperature sensor, example code for this is fairly common as the Arduino can use the OneWire library which is a proprietary protocol developed by Dallas Semiconductor. In essence the command 0x44 is sent to the device using ds.write(0x44), where ds is at instance of the OneWire class, the sensor reads the internal ADC and copies the data to the scratchpad registers which can be read by sending the command 0xBE then using the command ds.read() which returns the value. Those values are put into an array which is then added to the end of an array where the first 32 bytes entries are 0x00. Then encrypted

4.2 Data transmission

The data is packaged up as a POST request

4.3 Server Side

For the prototype, an Apache server, SQL server and Tomcat server was set up using XAMPP on a desktop. A Java web app was created as that is the language the writer has the most experience in and there are Java implementations of the TweetNaCl library, among other variations, available. The SQL table is a simple table that holds a key, timestamp and the signed and encrypted temperature sensor. (example?). The web app upon being accessed decrypts and checks the signature of each entry, using the keys that it has stored, in the table before converting the raw hex temperature data into more readable integers and displaying in a simple HTML table that can be accessed by the user. When the Arduino has data to send it will make a POST request to a PHP file on the Apache server which takes the data given to it and places it in the SQL server. (security flaw!)

4.3.1 Java web app

The Java Web app details what the server is do when it gets various types of request, be it get or post...(more on requests?). In this type of application you can dynamically printout all the HTML that will be used to make up the page. The usual HTML, head, body tags are printed at the top and the titles in the table are printed as well. (How it gets the keys?!). The web app uses JDBC to create a driver(idk man) to get the connection to the SQL database. Then using Java language it builds up a SQL query to take out all the values from the database and executes that. This puts all the table entries into a result set and the app cycles through that results set getting the relevant information out. The signed and encrypted hex is encoded as string and some leading zeros are lost in the conversion from byte array to string in the Arduino so these are added now before the string is converted back into a byte array. There is a try catch around the crypto_box_open and crypto_sign_open method so the server doesn't crash if one result set has been broken. Following this is the conversion from hex into integer for the user to read (how does it do it?) and finally the values are written to the browser along with the ending html tags.

4.3.2 PHP

PHP is a server-side scripting language

There are two files in the server, connect.php and add.php. The Arduino makes a post request to the add.php which effectively just calls it and the first thing the add file does is call connect which has the server details and makes creates a connection. Following that there is a SQL query that inserts the values sent in the post request into the appropriate table entries then close the connection. cooodee snippets

4.3.3 XAMPP Server

Apache for php files, SQL and Tomcat for web app. Accessible through web by port forwarding. what does the SQL table have?!

how does one access the xampp

4.4 NaCl

The TweetNaCl library as it stands in it's original form is not compatible with Arduinos. The C library compiles without errors but the compiler warns that the TweetNaCl method names are undefined and as a result do not perform their tasks. The method simply returns random numbers, it is possibly that it is trying to access some area of memory and simply returns whatever it finds.(ask greg/james). It is not understand why this is the case but it is a simple case of converting the library into C++ syntax. With a header file that has the main methods used in the project and the #defines and a cpp file with the TweetNaCl code. This is added in the same way to the Arduino IDE and in the code an instance of the class is created and methods are accessed with the dot operator.

The keypair, crypto_sign, crypto_box and equivalent opens were used. These are simple to use, abstracted methods that make this library easy to use. For the encryption the method needs the message to be encrypted which needs to have the first 32 bytes be zero, an empty array that needs to be at least the size of the message with the leading zeros, the length of the message, the nonce, arduino public key and the servers private key. This will reveal the temperature data with the signature. To remove the signature, the crypto_sign_open method needs the server secret signature key and the signed cipher array.

4.5 Machine to Machine

secure M2M public key transmission talk more about that

Chapter 5

Strength Of Security

5.1 Sign then Encrypt

Encryption stops unauthorised parties from accessing information, assuming authorised parties are the only party with access to a secret key. Signatures prove the message was not altered in transit, integrity, that the message came from a known sender and that sender cannot deny sending the message. These are two distinct operations and the order of these operations matters. If a user was to encrypt a message then sign and send it, as the signature is added to the message it can be stripped off by an attacker and replaced with their own. If the attacker managed to intercept the transmission of public keys in the first place then they can now impersonated the real recipient. Or they could change the message content through a bit banging attack and because there is no hash of the message to be compared to the receiver can't know if the message integrity has failed. In addition the attack can add any signature to the message, encrypting then signing does not prove that the owner of the signature was aware what was in the message as it was encrypted. In this application the message is signed first so the hash is taken of the plaintext message and it can be proved that the signer was aware as to the contents of the message and then it is encrypted.

can protect against the attacks mentioned earlier because...

5.2 Public Key Transmission

If they keys are preinstalled they can be used to sign and encrytp new public keys before they are sent. Or if they are not, the most secure and safe to make sure the keys are the same is to have a human user manually inspected the keys and carefully ascertain that the one sent is correct. Which has been demostrated in this prokect with the Arduino Due to Arduino UNo comms.

changing the public key in transit then it's being sent for the first time, human operator notices the different (mitm)

5.3 Bit Banging

bit banging the encrypted message In this project a test was set up to demostrate the cryptoghtraphy caused detected that the message had been altered in transit. After a signed message was encrypted it was altered before being decrytpted and the signature checked

As the format of the message is known, key areas of the message were altered and the following output notice

TABLE

5.4 Whats the name for gleaning info based on the cipher text

sending encrypted the one and 9999 yes and no

To encrypt the hex of the word YES, produces

,0xAE ,0x43 ,0xCD ,0xA5 ,0x8E ,0x54 ,0xE9 ,0x30 ,0x59 ,0xB1 ,0xD5 ,0xA5 ,0xBF ,0x24 ,0x9D ,0xEE ,0x69 ,0xDB ,0x37 105, 219, 55

To encrypt the hex of the word NO, produces

,0x2E ,0xBC ,0xCC ,0x6B ,0x9E ,0xDB ,0x29 ,0x64 ,0xF6 ,0x26 ,0x49 ,0xEF ,0xE9 ,0xFA ,0xBD ,0x9C ,0x7E ,0xD1 126, 209

From this message it is possible to tell the number of bytes sent if you know that the encryption method adds 16 extra bytes but other than that this test shows there isn't a discernable difference

To encrypt a message of all zeros ,0x5B ,0x74 ,0x76 ,0x4C ,0xF4 ,0x19 ,0x65 ,0x37 ,0xC4 ,0x53 ,0xAF ,0xB0 ,0xCE ,0x18 ,0x1 ,0x1 ,0x30 ,0x9E

To encrypt a message of a 1 ,0x49 ,0x1B ,0x1E ,0x52 ,0x10 ,0x38 ,0xD7 ,0x38 ,0x30 ,0x65 ,0x71 ,0xBC ,0xEE ,0x65 ,0x3D ,0x6 ,0x31 ,0x9E

To encrypt a message of 99999 ,0xC ,0xC ,0x29 ,0xE2 ,0x4E ,0x81 ,0x67 ,0x5 ,0x78 ,0x49 ,0x8C ,0xA1 ,0x4F ,0x69 ,0x8 ,0xBB ,0x9 ,0xA7 ,0x5D ,0x63 ,0x4D

and the other attacks

Chapter 6

Results

link to objectives

6.1 Recording of temperature

Using the thing it can record temp, sign, encrypt to ward off the attacks mention in the background. The temperat

6.2 Sending Data

The device successfully transmits data in the form of a POST request across to a web server.

6.3 Power Consumption!

6.4 Web App

Users can access the temperature table anywhere

6.5 Secure transmission of Public keys

Two arduino Dues in the same network can send their public keys secrurely and a person has checked that

show lots of pictures?

6.6 Machine to Machine

Two arduinos sent up in a client host relationship were able to pass data

Chapter 7

Critical Evaluation

7.1 SubSection

With the way it is set up at the moment the xampp add.php connect php and getKey.jsp are accessable through a browser. Some extra server side security is needed so that the only authorised parties are allowed access.

7.2 Other SubSection

Isn't a good rekeyer

Ethernet shield? Wifi is better but not cheap.

Chapter 8

Conclusion

8.1 SubSection

security measures will never be fully secure so all we can do is make it not worth the attackers while. At the moment there are so many devices that simply have no protection (web security “hack” example pls), so by having even the minimal amount of protection you can dissuade would be attackers because they have plenty of easier targets. Or make it so that the reward of breaking into your system simply is not worth the effort.

8.2 Other SubSection

Appendix A

Appendix Title Here

Write your Appendix content here.

Bibliography

[1] Andy Greenburg. *Hackers Remotely Kill a Jeep on the Highway—With Me in It*. 2015. URL: http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[2] Jeremy Korzeniewski. *Tesla Model S successfully hacked by Zhejiang University team*. 2014. URL: http://www.autoblog.com/2014/07/18/tesla-model-s-successfully-hacked-by-zhejiang-university-team/.

[3] Jill Scharr. *Hackers Hijack Prius with Mac Laptop*. 2013. URL: http://www.tomsguide.com/us/hackers-hijack-prius-with-laptop,review-1797.html.