# TRUST ANCHORS -
# What are they?
# Why do we need them?
# How do we make them?

## Background

In our previous paper “Security in IoT” we discussed the different types of communications between a sender and a receiver. In summary these are trusted and untrusted. In trusted communications the message can be relied upon as accurate, not tampered with and from a source that we trust. We act upon these messages with high confidence that they are valid and trusted. An example of this is a message from a partner to pick the children up from school. We trust the sender, we trust the method of delivery of the message (voice) and the consequence of this message being false is high. Interestingly if we trust the message, but the action is of very high consequence we usually require verification, and maybe from another source if very important (transferring cash from your bank account to another for example).

Untrusted messages are from sources we do not trust but the consequence of not acting on the information of the message is very low. We do not care if the message has been tampered with because we do not act upon the message in a direct way. An example of this is Twitter, Facebook or the general media. These messages are there to broadly inform and not to direct, we do not act directly upon these messages.

## What is a trust anchor?

A “trust anchor” is an irrefutable piece of information that can be exchanged between the parties of the communication so that they know for certain with whom they are communicating.

In order to have trusted communications between a receiver and a sender there are 4 things needed. These are: the trusted source, the trusted receiver, authentication and obfuscation. The trusted source is needed to allow us to believe the information we receive is from a reliable and trusted source. The trusted receiver is needed to allow us to believe that the information we send is sent to the correct recipient. Authentication is needed to ensure that if the information is tampered with whilst in transit to us we will know about it. And obfuscation is needed to ensure no one else can read the information and understand what it means.

An example of this is a bank card with smart chip and pin. The bank issues a credit card to the user and issues a PIN to be used only with that card. The bank trusts the bank card because of the PIN number. When a user places the bank card into the automatic teller it requests the PIN number. If the PIN number is paired with the details on the card, then it is valid and trusted. Subsequent transactions are thus authenticated by this verification process.

## Why do we need a trust anchor?

To perform trusted communications, we need a trust anchor at the sender and receiver. This trusted piece of information must be sent and verified. But the sender must know that they are talking to the correct receiver, and the receiver must know that they are talking to the correct sender.

Thus usually there are 2 pieces of irrefutable evidence that the sender and receiver can agree upon. One piece is used sender → receiver, and one piece is used receiver → sender.

The critical requirement for the trust anchors is that they cannot be impersonated, altered or copied. If they can then all subsequent communication between the sender and receiver cannot be trusted, ever.

An example of this is the bank card again. If a bank card is cloned and is subsequently used to purchase goods, then so long as the PIN is not known the communications cannot be authorised. This will occur if a cloned card is used at a store with a Chip & Pin reader. However, in many situations to purchase goods over the Internet the PIN number is not needed. When this occurs, and the user notices unusual activity on the bank account, the bank will invalidate the card and re-issue.

In the above example the irrefutable information that the bank placed on the card has been copied and the card impersonated for the purpose of fraudulent communications, instructing the bank to transfer funds in exchange for goods or services. We live with the possibility of our bank cards being cloned because of the convenience of using them. In addition, the bank insures the user from financial loss if the card is used in fraud. Note that if the PIN is used with the card, the bank will not reimburse the user from the fraud because they assume the PIN is only known to the user.

It is convenient for the bank to use credit cards because of the financial industry, insurance industry and government backing. If there was no insurance on fraud, customers would continue to use cash (or other valuable substances - gold) as the only method of paying for goods and services. That would be bad for banks, customers and governments who have developed whole industries around credit and digital currencies.

## How do we build a “trust anchor”?

In our example of a bank card the PIN is sent via post to our home address. The card is only activated by the PIN, so that the user needs both before any transactions can be performed. The

PIN MUST be memorised and this forms the trust anchor – in our brain. It is impossible to remove this information without our co-operation, i.e. it cannot be biologically removed.

You can see that in practice there are 2 vulnerabilities to this approach – (1) the supply chain and (2) the human. The supply chain is the Bank to User transfer method of the Card and the PIN. The bank sends the card and PIN to the registered address. However, if the card and PIN can be intercepted, copied and forwarded to the user, then a clone of the Bank card and PIN will exist. The bank assumes that this is very rare, it trusts that the user is not performing the fraud themselves and that in our society the authorities will investigate the fraud. There is a growing market for peoples' bio-data where names, dates of birth and addresses are sold and purchased by criminals for exactly this type of fraud. Supply chain fraud is very common in all digital systems and is the main access point for criminals trying to hack your devices.

The second problem (2) is that humans are not very good at keeping the trusted information. They are unreliable and will write the PIN down, or change it to a common one used across their mobile device and other cards. They will not shield themselves as they type the pin into the reader. Also they will give up the PIN when subjected to blackmail or torture. Still the Chip and PIN is better than the good old signature – which was easy to copy.

## Silicon trust anchors

The above examples are based around human interaction with bank communication systems for the purchases of goods and services. In this article we are focused on communications between devices in the Internet of Things.

These devices – smart sensors, the cloud, smart actors and interface devices (smart phones) all require trust anchors to allow communication between them. The remainder of this article discusses the techniques to build a trust anchor on these such devices.

In order to build a trust anchor into our IoT device we need 3 things: (1) a trusted processing element from a trusted source, (2) a method to ensure that the processing element cannot be tampered with and (3) a source true random numbers.

Many integrated circuit manufactures provide trusted products that can be used for this purpose. In general, they all share a number of features that we will describe later. They all have methods to detect and act upon tamper attempts. Many of them include random number sources.

## Why do we need these features?

In essence building a "trust anchor" is analogous to building a bridge. A bridge is built upon foundations that must be secured. Each element of the bridge is required to play their part in the structure of the bridge and if they fail the integrity of the bridge may be compromised. The upper layers of the bridge (where people and traffic pass) can only be used if the lower layers are secure. However, an attacker can attack any part of the bridge and thus can weaken its integrity. So it is with our silicon bridge.

## Silicon foundations

We need to have the foundations of the bridge embedded deep into the fabric of the silicon, so that it is difficult for an attacker to compromise its integrity. We can do that by designing the structure of the silicon chip in such a way that it is virtually impossible to probe the transistors that go to make up the device. Looking for 256 transistors that contain a vital key amongst 100,000,000 is virtually impossible, given they could be randomly distributed across the device.

## Embedded memory and how not to use it

Memory at the device level is designed in a regular pattern, with row upon row of memory elements. If we are looking for a 256 bit key and we know it's in memory, we can do an exhaustive search of the memory looking for specific changes in the structure. Thus it is not a good idea to store keys in on-board key storage, because the size of the store is usually very small <64k bits and it makes it easy to reverse-engineer the memory. However, on-chip key storage can be used if the manufacturer goes to extensive lengths to hide the key store amongst other elements within the chip (obfuscation).

We must use manufactures that have built their silicon with security in mind and have added obfuscation and measures that make probing the structure of the silicon difficult to do.

## One- time programming

As with our bridge analogy, we should not allow an attacker to replace parts of the bridge with their own parts and thus compromise the integrity of the bridge in a subtle way only known to the attacker. This is achieved in silicon by only allowing the chip to be programmed once, and not to allow any more access to the programming interface.

## Programmability

In the “good old days” electronic circuits were designed to achieve a specific task, or a limited set of tasks. For example, the early calculators – see below, my first calculator - Sinclair Cambridge Scientific calculator from 1977. These could not be hacked as they were designed to perform a very limited set of tasks and only those tasks.


Fast forward to 2017 and everything we purchase has some form of programmability because the cost of minituarisation of transistors onto silicon has fallen to a point where it is cheaper to add this functionality than to remove it. The average refrigerator will have a micro-processor in it, not because good old fashioned electronic circuits cannot perform the same tasks, but the micro-processor is far cheaper to mass produce. The reason for this is that manufactures can add features to their range by putting functions into software that run on the processor, rather than having a whole range of different electronic circuit boards to achieve different functions. The ultimate custom solution!

Now the silicon chip we purchase from our chosen manufacture contains general purpose processing elements (transistors). These could be a processor such as an Arm core, an Intel Atom Processor, a Z80 processor or a set of general purpose logic gates as in a programmable gate array. In order to make the chip do something useful it must be programmed. The programme can either be a set of instructions for a processor to execute – generally called software,or it

could be a set of boolean functions such as AND, OR, NAND, NOR, XOR, generally called VHDL. The boolean functions are used in mathematical operations such as cryptography or compressing music to an mp3.

The software or VHDL must be programmed into our chip to make it do something useful. However, we do not want to give a hacker the ability to replace some or all of our software and VHDL with their own, so we need to disable the programming interface after the chip has been programmed. We call this one-time-program. Additionally, we do not want to allow the hacker to read the programme or firmware from the chip so that they can understand how our bridge is built.

## Active and passive defence measures

So we are now getting close to our goal of creating a “trust anchor”, however we need a method of protecting our design if a hacker tries to attack our bridge.

The analogy is that if an attacker tries to alter the integrity of our bridge in some way we need to detect this and act upon it. This can be achieved in 2 ways – passive and active methods. Passive methods for our bridge could be painting the structure in plaster and black paint so that an analysis of the underlying structure of the bricks of the bridge will be difficult. Or placing sacrificial bricks around the structure so that if one other bricks is touched the bridge crumples and it is obvious an attempt has been made to attack it. This could be seen as self- defeating that as soon as an unauthorised person tries to touch the bridge it breaks. But what is better – knowing the the bridge is unsafe and therefore use another one, or drive a cross it and it falls apart killing the occupants of the bus?

However, before an unauthorised attempt is made at touching the bridge, we can use active methods to protect it – stationing troops on the bridge to protect it etc. The active methods are thus activated before the passive methods.

This is the classic approach to security – "layers of an onion"

In our silicon these security layers form barriers that an adversary must avoid before they can attack the silicon. They start from the outer layer of the equipment or circuit board with the power supply.

- If the supply is interrupted, then a tamper alert is flagged. A small battery is housed inside the equipment and maintains the active alerts even if the main power has been cut.
- If the equipment housing or Printed Circuit Board (PCB) housing is breached a tamper alert is flagged.
- Temperature, frequency, humidity, pressure, radiation, electromagnetic interference are all parameters that can be measured and a tamper alert flagged.

After the internal battery has died we fall back on passive measures.

To understand these, we need to delve into the workings of silicon chips. Broadly they fall into 2 categories – Static Random Access Memory (SRAM) based and fuse based. In SRAM based devices the property of capacitance (like a balloon stuck to a wall by static electricity) is utilised to hold information. In fuse based devises individual silicon fuses are physically blown or replaced to hold the information.
In fuse based systems a physical power supply is required to change the state of the fuse. Thus at programming time the fuses are altered according to the pattern required. And in a typical device there are 50,000,000 fuses.

In SRAM based devices power need only be applied when a read or a write is performed (reading removes the charge from the capacitor and thus must be replaced from the power supply). The capacitance inside the chip will eventually fade away, but this leakage is in the order of years (this is why the information in
a USB memory stick has a shelf life).

This feature can be used as a passive tamper, in that just like our sacrificial bricks in bridge, if someone tries to read the contents of the SRAM without power the contents are destroyed.

## Summary

In this article we have looked at the reasons for securing communications by using trust anchors. We have established that a trust anchor is an irrefutable piece of shared information that is only known to the parties in the communication. We have also outlined the reasons why a successful trust anchor must not be able to be copied or altered. In humans, memories form our trust anchors, and they are impossible to remove, and also can be forgotten!

In digital computers – digital memory is used to store the trust anchor information. However, unlike biological memory, it is relatively easy to remove this information. In order to protect this information, the silicon trust anchor must employ passive and active techniques. In essence these techniques are used to erase the trust anchor information if a tamper is detected. The reasoning behind this is that it is better to destroy the information than to let it fall into the wrong hands. After all the trust anchor information can always be re-installed, but once the injection is administered to a patient by an IoT enabled automatic dosing machine, it is too late, they cannot be re-installed.

The most important lesson to take from this article is that a trust anchor can ONLY be built from specialised hardware that has been designed for the purpose. This is why IoT has such poor security.

All general purpose computers have security vulnerabilities and can be reverse-engineered. All software running on general purpose computers are susceptible to hacking, with no exceptions.

Only computers that exist on trusted hardware in a closed environment can be trusted. The Blackberry range of mobile phones from RIM were a good example of trusted devices, and have been used by most “agencies” for secure communications. However, general consumers of mobile devices like the ability to customise their devices, so Android and iPhone have become more popular – at the expense of security.

Even Apple's attempts to secure their devices have been circumvented – an Apple is amongst the best of the commercial devices for security ( Dr Sergei Skorobogatov: “demonstrated NAND mirroring attack on iPhone”, http://www.bbc.co.uk/news/technology-37407047). This is because in the battle between security and usability, usability always wins!

--o--