Notes

Tuesday, 4 June 2019 6:10 PM

Security Engineering

What is engineering?
Engineering is the act of identifying problems and developing solutions for that problem.

What attributes make an awesome security engineer

- Good at ***analysis*** - there is a primary importance of analysis above all else
  - Don't just do or say the first thing that comes to your mind
  - Think from both a defender's and an attacker's perspective
  - Be aware of strengths, disadvantages and recommendations
- Need ***awareness*** and the ability to farm/research history and other domains
  - Learning from history and past mistakes
  - Being able to identify features that look familiar and build on previous solutions
- ***Case studies***
  - When things go wrong, we need to look at it and learn from it
  - How can we prevent it from happening in the future
  - Look into what happened, the outcomes, reasons why things occurred the way they did and think of recommendations
- ***Decisions***
  - Being able to factor points for and against a topic and make a decision based on it
  - Able to provide solid reasoning for your decision
- See ***what's important***
  - Identify important details in a mess of complexity
  - Don't waste time on non-important stuff, push them aside and focus on what matters
- ***Certainty***
  - You should never be certain that what you are doing is right
  - Always be open to new opinions
  - Never be satisfied with what you have done and keep improving
- ***Humans***
  - Must understand humans
  - Strike a balance between understanding humans and technology. Knowing to much about one or the other does not provide you a full picture of the scenario
  - Are complex making it easy to overlook and make mistakes and vulnerabilities

Terminology

Vulnerability	A weakness in the system
Software Bug	An error in code that can be a vulnerability
Exploit	Scripts or other methods utilising a vulnerability to compromise a system
Threat model	What you are up against

History of Hacking

We have always had the materials to make any tools in the world. The only difference is the complexity of how the tools are made.

Security through Obscurity

Security through obscurity is the reliance on secrecy or complexity to make something secure. This is a very bad idea. Why?

- The more complex a system is, the harder it is to analyse whether the system's security is actually improving when changes are made; just because it is harder to understand does not make it more secure
- Increased complexity may induce unforeseen and hidden vulnerabilities
- Complexity can always be unravelled

Kerckhoff's principles of cryptography

1. The system must be practically, if not mathematically, indecipherable;
2. **It should not require secrecy, and it should not be a problem if it falls into enemy hands;**
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will;
4. It must be applicable to telegraph communications;
5. It must be portable, and should not require several persons to handle or operate;
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

The most important principle is the second one. Good security is when the enemy knows everything you know, but even with that knowledge, they are unable to do what you can

Side-Channel Attack

A ***side-channel attack*** is a security exploit that involves collecting information about what a computing device does when it is performing cryptographic operations and using that information to reverse engineer the device's cryptography system. It is an attack based on information gained from the implementation of a computer system, rather than a weakness in the implemented algorithm itself. Timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited.

Cryptography Principles

CIA Principles

Properties you might want a system to have

- **C**onfidentiality - ensuring that only authorised people can access certain information
- **I**ntegrity - maintaining the 'trustworthiness' of a message (i.e. ensuring the message is not altered or destroyed by an unauthorised party)
- ~~**A**vailability - the message or information is in an available condition~~
- **A**uthentication - the act, process, or method of showing something to be real, true, or genuine

Simple Codes and Ciphers

A ***code*** is where each word in a message is replaces with a code word or a symbol
A ***cipher*** is where each letter in a message is replaced with a cipher letter or simple.
Codes are abstract, conceptual and completely arbitrary.

Steganography - the act of hiding the existence of the message. As soon as someone knows the method of hiding the message it become useless. E.g Greeks would tattoo messages on the shaved heads of slaves, where the message would be hidden once their hair grew back

Ciphers

- Caesar cipher - the substitution by shifting the letters by a certain number. Can be decoded via letter frequency and heuristics
- Substitution - the substitution of letters; a is represented by b;
- Transposition - letters are rearranged by some predetermined rule or key. Has the weakness of the letters being the same
- Learn Jefferson's wheel, playfair, polybius

Weakness of the Week 1: Changing our frame of mind

It's very hard to change the way you think. Your daily patterns are drilled inside your head that it is very difficult to shift between the mind frame of working or having a coffee to an emergency mindset.
We tend to have a confirmation bias, favouring information that confirms our previously existing beliefs or biases.

Case study: Halifax Explosion

The largest man-made explosion at the time of WWI.
An explosive ship came to Halifax harbour. A few days before another ship came and
As the ship is arriving (on the right side) another ship is leaving on its left side. They bump into each other, causing come barrels to fall over and leak .
French ship is set on fire. Frenchmen go on sail boats to land, yell "run for you lives"

Small amount of people leave while the rest stay to see the spectacle. People watching the blast got blinded
Made a tsunami

How would you prevent this?

- Limit what you can bring
- How many ships in one harbour
- Standards for how to enter and leave a port
- Evacuation plan
- Boat traffic control

Security by Design

We should design security so that if one component fails, the whole design does not either.
Security should also be built in from the very beginning, otherwise you will find yourself fixing breach after breach.

The equivalent of:

> *"You are no better than a barbarian trying to box. Hit him in one spot, and his hands fly there; hit him somewhere else, and his hands go there" - Demosthenes*

There are 4 primary colours for a security engineer:

1. Trust - always have a sense of scepticism with whatever you approach. Trust no one and do not build security based on someone else's perceptions of trust. By doing this, you are able to make sensible choices.
2. Secrets
3. Humans
4. Engineering - risk, complexity

Although it is impossible to achieve perfection, aim to get as close as you can to it. After all it will be the best you can get.

Defence in Depth

Defence in depth is security that is layered or compartmentalised, so that if one component breaks, the whole system doesn't either. If security is designed only with an outer shell, once it has broken through the inner working will have nothing left to be defended by. This is a single point of failure.

An example of defence in depth is the ***Bell-LaPadua Model*** (BLP). Information is often partitioned into different security levels and a person can only access information determined by their security level.
The two main security properties are:

1. A subject cannot read an object at a higher security level
2. A subject cannot write to any object at a lower security level

Apes vs Ants

Apes	Ants
• Have complex interactions between individuals by choice • Capable of breaking from a role in society • Have individuality, free will	• Individually they are not complex, but together they are • No ant acts on their own, work together as a whole • Colony is rigid and their roles are defined

Physical Security

Everything we deal with is in the physical world, but there is also a trace in the physical world. For the virtual world to exist it must run on a physical machine. If someone messes with the physical machine the virtual world can be destroyed. Hence we must consider physical and hardware security measures before considering software security measure.

Motto*: every contact leaves a trace*

Cryptographic Protocols

Vigenere Cipher

Essentially the Caesar Cipher with an additional key, where the number of letters you need to shift is dependent on the key. E.g. the key is egg (5,7,7). For your message, you will shift the first letter by 5, the second letter by 7, the third letter by 7, and repeat.

Enigma Machine

A polyalphabetic substitution cipher. Used the concept that *a purely random key sequence, containing no repetitive pattern would be unbreakable*.

System had three physical rotors. Each takes in a letter and outputs it as a different one. After the first letter has come through, the first rotor rotates one position. When the first rotor has gone through all 26 positions the second rotor will then rotate. The same will occur for the third rotor once the second rotor has gone through all 26 positions.

Weaknesses of the Enigma ciphering system:

1. No letter could be encrypted to itself. This meant that some possible solutions could quickly be eliminated because of the same letter appearing in the same place as the ciphertext and the possible plaintext.
2. Plugboard connections were reciprocal; if A was plugged to N, then N likewise became A
3. Notches in alphabet rings of rotors I to V were in different positions, which help cryptanalysts work out the wheel order
4. Operational shortcomings.
    a. Production of an early Enigma training manual containing the plaintext, ciphertext and message key
    b. Repetition of message key
    c. Using the same stereotypical expressions in messages (most began with letters ANX - German for "to")
    d. Used easily guessed keys. e.g. AAA, or BBB, or sequences that reflected the layout of the Enigma keyboard; three keys next to each other
    e. Only 3 different rotors for 3 positions in the scrambler

One Time Pad (OTP)

An encryption system, where the private key is generated randomly with length as long as or longer than the message. The key is used only once to encrypt a message that is then decrypted by the receiver using a matching one-time pad and key. Each encryption is unique and has no relation to the encryption.

So it is essentially a Vigenere cipher, where the key is as long as the message.

Weakness lies in the fact that the key must be pre-shared.

Kasiski's Test

A method of attacking polyalphabetic substitution ciphers such as the Vigenere Cipher.
The fundamental weakness of a Vigenere cipher is that if the key length is known, the ciphertext can be split apart into individual shift ciphers. So security relies on having the key length known.

Kasiski's insight:

- There are common bigrams and trigrams in the plaintext: TH, MM, RE
- From time to time, two occurrences of a bigram/trigram will be separated by an exact multiple of the key length
- This means that the two occurrences will be encrypted in the same way

This suggests:

- Find the common bigrams and trigrams in the ciphertext
- Find the distance between them
- This distance may be a multiple of the key length

By assumption, some (but **not necessarily all**) of these repeated bigrams and trigrams are separated by multiples of ***k***, the key length.

If you find every occurrence of every bigram and trigram, you'll generally find nothing useful. The art of the Kasiski attack is finding a number that divides most but not all of the separations.

Coincidence Index/Incidence of Coincidence

The coincidence index lets you determine what type of cipher was used to encrypt a message
It measures how likely it is to draw two matching letters by randomly selecting two letters from a given text.

Suppose a particular letter appears *k* times among *N* letters.
There are $\binom{N}{2} = \frac{N(N-1)}{2}$ ways we can pick two letters at random, and $\binom{K}{2} = \frac{K(K-1)}{2}$ ways we can pick the designated letter, so the probability that both letters we pick are the designated letter will be

$$\frac{\frac{K(K-1)}{2}}{\frac{N(N-1)}{2}} = \frac{K(K-1)}{N(N-1)}$$

The probability can then be normalised by multiplying it by some coefficient (typically 26 in English).

$$IC = C\left(\frac{\sum K_i(K_i-1)}{N(N-1)}\right),$$

where K_i is the number of times the *i*th symbol appears and C is the normalising coefficient

The expected average value for the IC can be computed from the relative letter frequencies fi of the source language:

$$IC_{expected} = \frac{\sum_{i=1}^{c} {f_i}^2}{1/c}$$

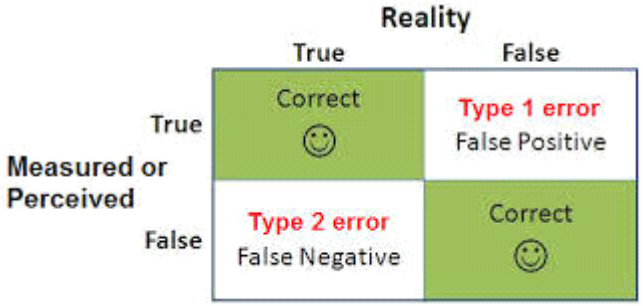
If all c letters of an alphabet were equally probable, the expected index would be 1.0. The actual monographic IC for telegraphic English text is around 1.73, reflecting the unevenness of natural-language letter distributions.

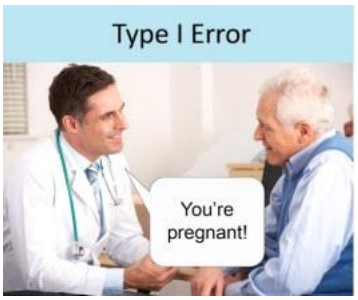
Trick:

1. Line up ciphertext with copy of itself and count the number of times the letter in the first copy it is aligned with in the second copy
2. Coincidence rate = #coincidences/length of message
3. Multiply by 26 to get an estimate of the Coincidence Index

Type I/Type II Errors

Type I: False Positive - predicting something to be ***true***, but actually turning out to be ***false***
Type II: False Negative - predicting something to be ***false***, but actually turning out to be ***true***





One of the errors will have higher consequences than the other and you will have to decide which one you want to reduce more. Note that you cannot reduce both.

Anatomy of an Attack

The anatomy of an attack can be summarised like this:

1. Someone is targeted
2. Information is gathered on the target
3. The target does something dumb
4. Your privileges rise due to the target's dumb act
5. Repeat on higher level targets until you reach the top

Technical skills aren't always required to make an attack. It is important to be able to gather information and use that information to trick people into making mistakes.

Weakness of the Week 2: Self Interest

Humans act in self-interest, doing things only if they benefit them. Initially we may not act in self-interest, but it can be very tempting and easy to do so. Intentions that start good, can turn malicious over time.

Self-interest can be good for balancing out people perspective. Like how telephone pole cannot stand straight if it only has a wire on one side, the world needs opposing parties to balance out everything.

Is-Ought Problem by David Hume

Many people make claims on what ***ought*** to be, based on statements about what ***is***.
They follow the logic that: *because this is true, this should be done*.
We evolved as meat-eating animals, therefore we ought to eat meet.
People move easily from observations to facts to judgements about values.

We should sever facts from our personal values.

Risk

Risk is invisible. We often don't know we have taken a risk until we face the consequences of the risk. Risk is always bad regardless of whether it happens or not. We only get mad and punish people when a risk goes wrong. Little do we realise that taking a risk is just as bad is its effect.

Humans are bad at assessing risk. Because the impact of risks are usually unlikely to happen, we don't get an actual accurate representation of the probability of the risk happening. Only through repetition and more experiences can we get a more accurate representation of risk. Allocate your resources appropriately across risks, but do not fixate on them.

A ***risk matrix*** can help us allocate resource to a risk, by plotting risks on graph with scales impact and likely hood. It's obvious that we want to focus on high-impact high-probability event and ignore low-impact low-probability events. The other two are a bit more tricker; high-impact low-probability and low-impact high-probability. Impact is also arbitrary depending on the individual so this may skew the risk matrix

Risk registers are tools for documenting risks and actions to manage each risk. They should be conducted by third parties to remove self-interest and many people should be involved to get as many perspectives as possible.

Compliance models can be used to check that clients satisfy certain criteria. They are good for stopping dumb things happening and set a minimum standard. It is good to have compliance, but that should not be the limit; go above and beyond, do not stop at what is set.

Dealing with Risk

There are four distinct types of approaches to risk mitigation:

1. **Prevention:** Eliminate the threat such that the undesirable event is guaranteed not to occur. Inherent saftey prevents undesired events.
   E.g. removing the bolt and bullets from a rifle to prevent it from accidentally firing.
2. **Limit:** In situations which the threat could not be eliminated, mechanisms are implemented to minimize the impact resulting from the event (fault tolerance) or reduce the likelihood of it occurring (probabilistic safety).
   E.g. a castle with concentric walls. Even if one wall is breached, only a limited portion of the castle is compromised.
3. **Passing the risk to a 3rd party:** This involves in making another person/entity the bearer of the risk.
   Generally taking out an insurance policy is used minimize or negate financial loss in an event of a failure.
4. **Wear:** Take the risk and hope for the best

Cyber Literacy

Cryptography	The science of securing data
Cryptanalysis	The science of analysing and breaking encrypted data
Cryptology	Embraces both cryptography and cryptanalysis
Ciphertext	The encrypted message
Plaintext	The non-encrypted message
Ciphers	Cryptographic algorithms, mathematical functions used in the process of encryption and

	decryption of data
Encryption	The process of converting a plaintext message into ciphertext. Specifically, it refers to algorithmic schemes that encode plaintext into a ciphertext, providing privacy
Decryption	The reverse process of encryption
Key	A secret compression. Usually, receiver of encrypted text uses a *key* to decrypt the message.

Logic puzzle

All **Martian men lie** and all **Martian women** tell the **truth**
All **Venusian men** tell the **truth** and all **Venusian women lie**.

One day Martians and Venusians come to Earth. They looks exactly the same; even men and women. How do we tell whether the alien is a man or woman, Martian or Venusian?

	Telling the men from women: ***Are you from Mars?***	Telling Martians from Venusian: ***Are you a man?***
Mars man	No	No
Mars women	Yes	No
Venus man	No	Yes
Venus women	Yes	Yes

Bits of Security

Calculating bits of security

1. Find the total number of permutations (i.e. the total number of possible passwords, keys etc.)
2. Find the nearest power of 2 number (1, 2, 4, 8, 16, 32, 64, …)
3. Convert that number into exponential for with a base of 2 (so 2 -> 2^1, 4->2^2, 8->2^3 etc)
4. Take the exponent value. That's the number of bits of security

Note: when calculating the average bits to brute force a system, it will always be one less than the total bits of security since of average it will take half the attempts to correctly guess something

Alternative: $Bits\ of\ Security = \log_2(total\ \#\ of\ permutations)$

Bits Made Easy

It went like this:

$2^{10} = 1024 \approx 1000$
$2^{20} = (2^{10})\text{^}2 \approx 1000 * 1000 = 1\,000\,000$
$2^{30} = (2^{10})\text{^}3 \approx 1000 * 1000 * 1000 = 1\,000\,000\,000$

And just like that it's easy to know how many bits are in 1 billion or even 2 billion.

Bits of Information

One question can give us two bits of information as seen in the logic puzzle above. So if ***q*** is the number of questions we ask, the amount of information we get is 2^q

Bits of Work

We measure the amount of work it takes to brute force something in ***bits***.
Exploiting the space-time trade-off can roughly halve the effective number of bits; the more space you have the less time you need to compute.

Suppose we have a key of length 3 and 62 possible characters we can encode a character to.
The number of possible options is $62^3 \approx 64^3 = (2^6)^3 = 2^{18}$
So it would take 18 bits of work to try our every combination.

You have 2^{18} attempts to find the key. If we try to find 100 keys of length 3, the maximum amount of work it takes is usually less than half the effort the actual effort. So solving 100 keys would take 100×2^{17} attempts.

Bits and Compression Examples

Ex 1: If it takes A 2^{10} bits of work to get done, and B 2^{20} bits of works done. How many bits of work will it take to do A and B one after the other?

$work = 2^{10} + 2^{20} = 2^{10}\,(1 + 2^{10}) \approx 2^{10} \times 2^{10} = 2^{20}$

So A and B can be done in 20 bits of work.

Ex 2: In the worst case, how many bits of work does it take to break a substitution cipher.

There are 26! ways to assign the letters in the alphabet to a corresponding cipher letter.

$26! \approx 4 \times 10^{26}$

$= 2^2 \times 100 \times (10^3)^8$

$= 2^2 \times 2^7 \times (2^{10})^8\ since\ 2^{10} \approx 10^3$

$= 2^{87}$

So it would take approximately 87 bits of work

Ex3: Your computer has a 16GHz processor. How many bits of work can be done in a year?

Richard says it can do $34 + 12 + 5 + 9 = 60$bits of work

$16GHz = 16 \times 10^9$

$= 2^4 \times (10^3)^3$

$= 2^4 \times (2^{10})^3$

$= 2^4 \times 2^{30}$

$= 2^{34}$ *= 34 bits*

$2^{34} \times 3600\ cycles/hour$

$= 2^{34} \times 3.6 \times 2^{10}\ cycles/hour$

$= 2^{34} \times 2^2 \times 2^{10}\ cycles/hour$

$= 2^{34+12}\ cycles/hour$

$= 2^{34+12} \times 24\ cycles/day$

$= 2^{34+12} \times 2^5 cycles/day$

$= 2^{34+12+5} \times 365\ cycles/year$

$= 2^{34+12+5} \times 2^9\ cycles/year$

$= 2^{34+12+5+9}\ cycles/year$

Public-Key Cryptography

Public-key cryptography is also known as asymmetric cryptography.

In this system, you have two keys; a public and private key. The public key is available to everyone and is used to decrypt messages that intend to be sent to you. The private key is kept secret and is used to decrypt messages encrypted by the public key. So if you want to send someone a secret message, they can give you their private key for you to encrypt your message, and they can decrypt it with their private key.

Ralph Merkle created an early version of the public-key cryptosystem known as ***Merkle's puzzles***. It worked like this:

1. A sends a bunch of puzzles to B containing an identifier and key.
2. B chooses one puzzle and solve it. The solved puzzle contains an identifier and session key for B to tell A the puzzle has been solved.
3. They communicate!

If anyone wishes to eavesdrop, they will have to figure out which puzzle was solved.

RSA

RSA is an ***asymmetric*** system, which means that a key pair will be generated; a public key and private key. Obviously you keep the private key secure and pass around the public one.

It is based on the fact that $m^{pq} = m^{qp}$.

How does it work?

Say you have a message $\boldsymbol{m}$, which is in the form of a number.

1. We have generated the public key $(\boldsymbol{k}, \boldsymbol{n})$ and private key $(\boldsymbol{j}, \boldsymbol{n})$.
2. To encrypt the message:

$e = m^k mod(n)$

3. To decrypt the message:
   $d = e^j mod(n)$
   When the operation is complete ***d*** should equal ***m***

Generating public and private keys

1. Pick two prime numbers $\boldsymbol{q}$ and $\boldsymbol{p}$
2. Calculate $\boldsymbol{n = pq}$
3. Calculate $\boldsymbol{z = (p-1)(q-1)}$
4. Choose a prime number $\boldsymbol{k}$ such that $\boldsymbol{k}$ is co-prime to $\boldsymbol{z}$
5. Numbers $\boldsymbol{n}$ and $\boldsymbol{k}$ become the public key
6. $\boldsymbol{kj = 1\ mod(z)}$. Find $\boldsymbol{j}$ and the private key will be $\boldsymbol{n}$ and $\boldsymbol{j}$

Diffie Hellman Key Exchange

DH is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols.

The simplest implementation works like this:

1. Pick two number ***p*** and ***g***, where ***p*** is prime and ***g*** is a primitive root modulo ***p***.
   We will use ***p*** as the modulus and ***g*** as our base.
2. One person chooses a secret number ***a*** and the other also chooses a secret number ***b***
3. The (secret) key will be
   $g^{ab} mod(p) = g^{ba} mod(p)$ which is essentially $\left(g^a mod(p)\right)^b mod(p) = \left(g^b mod(p)\right)^a mod(p)$

Diffie Hellman **does not provide authentication**. It only confirms that they person you are communicating with has not changed since you started communicating

Weakness of the Week 3: Corruptible

Humans are corruptible. There is now way of telling who gets corrupted and there are many reasons why; money, ideology, greed, jealousy etc.

Separation of powers is commonly used to prevent one person or body having absolute power because "*power corrupts, absolute power corrupts absolutely*"**.**

Famous insiders:

- *Kim Philby 34-63 and the Cambridge Five*
- *Gordievsky*
  - *the third lock - every contact ...*
  - *the*
- *Michael Bettany MI5 1984*
- *Aldrich Ames CIA 86-93 - 13 June 1985 named 25*
- *Robert Hanssen FBI 79-2001 - revealed a multimillion dollar eavesdropping tunnel built by the FBI under the Soviet Embassy in Washington.*

Brute Forcing

We use brute force to measure the worst-case amount of effort needed to break security.

In the case of ciphers in any languages, there are a lot of redundancies, and patterns, which let us use brute force in a more intuitive approach. e.g. there is a limited number of valid 5-letter words in English, which would be significantly less that the 26^5 possible words we get from brute force. Because of the patterns in language, we are able to significantly reduce the amount of work we need to do. There is also the context of words, which help us deduce what words may come next in a cipher.

Hashing

Hashes can be used to provide integrity and authentication for a message.

Hash functions transforms a perhaps large input with perhaps varying size to an outputs with fixed sized. A hash should have these properties:

1. Deterministic - no randomness
2. Difficult to reverse
3. Avalanche property - if the input is changed slightly, the output changes significantly
4. Unlikely that two messages will have the same hash

All hashes are bound to have collisions (since the number of input is usually larger than the possible outputs). We want to minimise collisions as much as possible.

Examples of poor hash functions (6[48]41 students hashed to a playing card):

- Clustered collisions - similar inputs having similar or colliding hashes
  e.g. people sitting next to each other having the same suit
- Non-uniform distribution
  e.g. more undergrads than postgrads, more 6441 students than 6841, birthday paradox

Desired resistance against attacks

- Preimage resistance: given the hash of M, $h(M)$, you can't find M
  essentially hashes should be difficult to reverse
- Second preimage resistance: given M, you can't find M' such that their hash of M and M' are the same; $h(M) = h(M')$ (a targeted collision attack)
- Collision resistant: you can't find any two messages, M and M' where $h(M) = h(M')$

Birthday Attacks

A ***birthday attack*** is a type of cryptographic attack that exploits the mathematics behind the birthday problem in probability theory.

The attack depends on the fact that the number of items you have to look at before a collision is surprisingly small. If there are n possible hash values, then on average you will only have to look at $\sqrt{n}$ items before finding a collision.

Examples of Hash Uses:

- Message Authentication Code (MAC) - for authenticating and providing integrity to a message
  1. Takes a message and combines it with a key
  2. Hashes the above result (MAC)
  3. New MAC is sent along with the original message.
  4. Recipient verifies message by performing MAC on the original message with a shared secret
     i. If MAC matches, message is genuine
     ii. If MAC does not match, message has be compromised
- Commitment
- Proof of word
- Passwords
- Fingerprints

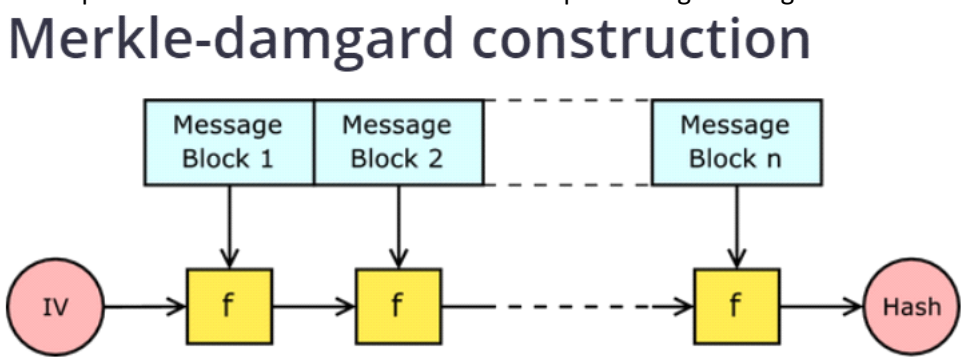
Examples of hash functions:

- MD5 - 1991 - Rivest -128 bits - VERY BROKEN
- SHA0 - 1993 - BROKEN almost immediately
- SHA1 - 1995 - NSA - 160 bits - BROKEN
- SHA2 - 2001 - NSA - 224/256/384/512 bits - NOT YET BROKEN
- SHA3 - 2015 - NSA - 224/256/384/512 bits - NOT YET BROKEN

A hash function is defined to be ***broken*** when the cost to break the function is less than what is claimed. E.g. a hash function has a 128-bit security level, so it takes 2^{128} tries to break it. If someone breaks the hash in 2^{127} tries, the hash is broken.

MD5 SHA0 SHA1 and SHA2 all use the ***Merkle-Damgard construction*** which breaks messages in to *n* blocks. It starts with an initial value (the ***initialisation vector***), which is a **fixed** value. For each message block, the compression

function *f* takes the result so far, combines it with the message block, and produces an intermediate result. The last lock is padded with zeros as needed and bits representing the length of the entire message are appended.



The MAC will essentially be the hash of the key with its data appended to it: `h(key+data)`

This construction can be subject to a ***length extension attack***, where given the hash H(X) of an unknown input X, it is easy to append another message to the already hashed message, altering the overall hash. Since the hash is technically correct, although we don't know the actual original message.

Passwords

Passwords are not very secure. Nearly one third of passwords fall into one or more of the following categories:

- First name/last name/nickname of a loved one
- Name of a pet
- Street/city/location of significance
- Birthday or special date
- Password incorporating the website name
- Any combination of the above

Be sure not to reuse any of your passwords because sooner or later a website or service you use is going to get hacked, and your password will be revealed, putting all your sites at risk.

Common methods for coming up with a password:

- ***Passphrase method*** - come up with a nonsensical yet memorable phrase compromising of about seven or so words that incorporates some number and symbols.
  e.g. the3Yellowrosessmell=goodins()mmer
- ***Abbreviated passphrase*** - come up with a unique longer phrase and choose the first letter of each word. Be sure to include capitals, numbers and symbols.
  e.g. “My dog (Jasper) always likes to eat green oranges when I forget to feed him!” would become “Md(J) al2egow1f2fh!”
- ***XKCD password*** - choose 4-5 random words building a mental picture of them all to aid memorisation and then joining them together to form a password.
  e.g. correctbatteryhorsestaple

Key stretching

Key stretching is a technique used to make typically weak passwords or passphrases, more secure against a brute-force attack by increasing the resources it takes to test each possible key:

- crypt - Robert Morris 1978
- bcrypt - 1996 - password hashing in open BSD and some Linux
- scrypt 2009

Salting Passwords

A ***salt*** is random data that is used as an additional input to a one-way function that *hashes* passwords. They are used to safeguard passwords in storage. Salts are closely related to the concept of a cryptographic ***nonce***, which is a number which is just ONCE in cryptographic communication.

A new salt is generated randomly for each password. The salt and password are concatenated and processed with a cryptographic hash function. The result is stored with the salt in the database.

Salts defend against pre-computed hash attacks such as rainbow tables. ***Rainbow tables*** are precomputed tables for reversing cryptographic hash functions.

Online and Offline Password Attacks

An online password attack would involve typing the password into the browser. Through this you are able to get locked out of the site.
An offline password attack would involve stealing a file containing you passwords, which are likely to be hashed, and trying to decrypt them locally (e.g. on your own laptop). On Linux, the /etc/shadow file typically contains the following user information

- User login name
- `"$id$salt$hashed"` : hashing algorithm id, salt and hashed password

Cryptographic Protocols

Motto: *don't roll your own*
Don't try to make you own cryptographic protocol and put it in production because it is bound to have flaws in it. We can't even implement binary search correctly :(

WEP - Wide Equivalent Privacy

How does it work?
WEP uses the stream cipher RC4 for confidentiality and the CRC-32 checksum for integrity.
Standard 64-bit WEP uses a 40 bit key which is concatenated with a 24-bit initialisation vector to form the RC4 key.

RC4 encrypts plaintext in the following way:
For each bit of the plain text, it produces on bit of keystream and XORs the two, to generate the ciphertext. The keystream is just a stream of random numbers generated by the RC4 algorithm.

An access point sends the encrypted data to the computer. Since the access point is technically streaming everywhere, anyone can see the data but they don't accept it because its not destined for them. A hacker can sniff the packets. They can figure out the key stream by XOR-ing the plaintext and ciphertext (since the position of the bits are always the same) and change the destination address to their own address. The access point will then unknowingly send the packet to the hacker.

Benford's Law (first digit law)

Benford's law is an observation about the frequency distribution of leading digits in many real-life sets of numerical data. The law states that in many naturally occurring collections of numbers, the leading significant digit is likely to be small. If the digits were distributed uniformly, they would each occur about 11.1% of the time

An ***Initialisation Vector*** (IV)

Mixing Data and Control

Buffer overflows

Buffer overflows are an attack on the stack and they occur when you are writing data to a buffer, and the data overruns the buffer's boundary and overwrites adjacent memory locations. This can cause the memory of an interrupted process to be overwritten.

An attack can be made used buffer overflows by overwriting return address with the your own malicious code. This will cause your code to be run instead of the expected process.

In the case where you don't know where in memory the addresses are, you will need to use memory leaks to find out.

A ***canary*** is a small integer, randomly chosen at program start, placed in memory just before the stack return pointer. The canary value is used to check for buffer overflows because its value would be overwritten if that were the case.

Moore's Law

Moore's law is an observation that the number of transistors in a dense integrated circuit doubles around every two years (although the period is often quoted as 18 months). The prediction has proven accurate for several decades and is linked to advancements in digital electronics.

Modern Symmetric ciphers

Data Encryption Standard (DES) is a symmetric-key algorithm that uses a key length of 56 bits with 8 bits for parity checks. Published in 1977 and standardised in 1979, it was found to be too weak and susceptible to brute force. A variant called triple DES uses DES three times and increased the effective key size a bit.

DES is a block cipher, splitting a plaintext message into 64-bit blocks. Each block is encrypted using a secret key by means of permutation and substitution. There are 16 rounds, where 16 sub keys are generated out of the 56 bit key

NIS ran a competition for a replacement code and we got Advanced Encryption System (AES) from this. It comes in different flavours each named after their key size. e.g. AES256.
AES can be subject to side channel attacks

Advanced Encryption System (AES) uses substitution-permutation network.
How it works: Each plaintext is divided in to 128 bit blocks, where cipher transformations are repeated over a number of encryption round. The number of rounds is determined by the key length

AES has yet to be broken, but is expected to be broken soon :/

Block Modes

Block cipher modes of operation are algorithms that use a block cipher to produce a ciphertext. Since block ciphers work in units of a fixed size, messages are divided into chunks. If a chunk if not big enough to be a block, padding is used to make it big enough. This can include adding null bytes, adding a single bit followed by enough zeros bits to fill out the block and others.
Let's go over some block modes:
EBC (Electronic Codebook) - messages are divided into blocks and each block is encrypted **separately**. This has the disadvantage that if two blocks are the same they will be encrypted to the same ciphertext block. This makes them susceptible to replay attacks.
CBC (Cipher Block Chaining) - each block of plaintext is XORed with the previous ciphertext block, then it is encrypted. An initialisation vector is used on the first block since it has no previous block XOR with. Since the plaintext is XORed with the ciphertext of the previous block, the decryption of a block can be correct even if the previous block was not.
CTR (Counter) - works like a stream cipher. A **nonce** is combined with a counter to produce a counter block for encryption. The use of a nonce, prevents the likelyhood of repetition.

Authentication

Many people believe authentication can be separated in to three factors:

- something you know e.g. a password
- something you have e.g. your phone
- something you are e.g. your face, fingerprints

But once you think about it all these factors are just something you know. They are essentially a secret which is represent by 1s and 0s to a computer.

Vulnerabilities

Here are a list of some vulnerabilities

- Memory corruption attacks - the attacker changes what is in memory so that the program behaves differently. The most common type is a buffer overflow.
- C formatted strings
    - %x attack - arbitrarily read information below the current memory address.
      `printf("%x\n"); // read the next item on the stack`
    - %n attack - arbitrarily write information below the current memory address

The value written is the number of bytes in the string so far.
printf("1234%n\n"); // puts 4 in the next position on the stack
When overwriting the return address of the function it is essentially an arbitrary jump

Assets

Assets are something that have value to you and they are something we want to protect. Sometimes its hard to identify you assets.

So how can we better identify assets:

- getting ***as many perspectives as possible*** - we need to look at our threat/problem from as many perspectives to fully grasp it
- develop ***sensible plans*** - critique your ideas constantly to improve them
- ***revise*** your current list of assets - assets are constantly changing you want to make sure you are aware of any changes

We also have different types of assets:

- ***tangible assets*** - physical items we can easily give value. e.g. jewellery
- ***intangible assets*** - more conceptual ideas. These aren't easy and objectively valued. It can be measure in terms of monetary value or even psychological and emotional costs

Public Key Infrastructure

PKI is a way of making sure that you are communicating with the website you think you are communicating with. A public key is available to any user that connect with the website. When communicating we, the clients, use that public key to encrypt and decrypt our messages while the website uses it private key to do the reverse.

PKI is authenticated using certificate authorities. They are the people who check the websites are who they claim they are and given them a certificate (usually found as a lock symbol on the url), which links their domain name to their public key. Your browser keeps a list of trusted certificate authorities and if it sees that a website has been authenticated by one of them it trusts the website.

PKI is great for preventing man-in-the-middle attacks because it authenticates who you are talking to and once communicating via public-private keys your communication can be expected to be secure. However a whole new areas of issues arise. Certificate authorities are paid to identify your website, not to check your website is dodgy and malicious. If you are who you say you are then you are given a certificate. There are authorities that do background checks on a company but that does increase the price of getting certified.

Secure Socket Layers (SSL) and ***Transport Layer Security (TLS)*** are both cryptographic protocols that provide authentication and data encryption between servers, machines and applications operating over a network. SSL is the predecessor to TLS.

1. A client contacts the server
2. The client and the server exchange information about the communications they intend to perform, such as the ciphers to use (SSL handshake)
3. The server transmits its certificate to the client
4. The client checks that it trusts the certification authority that issued the certificate. If it does not recognise the CA and does not get an override, the communication ends
5. The client checks for revocation information on the certificate. If the certificate is revoked or revocation information is unavailable, then the client might attempt to obtain an override. Implementations vary on how they deal with null or unreachable CRL information, but almost all will refuse to communicate with any entity using a revoked certificate
6. Both client and server send each other random data, which they can use to make calculations separately and then derive the same session keys. Three kinds of randomly generated data are sent from one side to another.
    - The "client random": This is a random string of bytes that the client sends to the server
    - The "server random": This is similar to the client random, except that the server sends it to the client
    - The "premaster secret": This is yet another string of data. In some versions of the TLS handshake, the

client generates this and sends it to the server encrypted with the public key; in other versions, the client and the server generate the premaster secret on their own, using agreed upon algorithm parameters to arrive at the same result.

7. Both parties generate 4 session keys using this data
8. All communications in the same conversation are encrypted with that set of keys

Errors

Whenever something goes wrong, we often try to find the *root cause* of the accident. There are three things we can allocate blame to:

- **humans** - we like to blame the last person to *last touch* the system as the cause of its disaster
- **the culture** - the culture of a company was not suited to running the system safely. This itself is something very vague and does not tell you specifically how to fix the problem. It's very easy to throw money at this problem and pretend that the culture has been changed because money was invested in it
- **the system** - if the system is too complex, too tightly coupled, not very cohesive, it is a disaster waiting to happen.

Human Weaknesses

- **Conflict of interest**
- **Dishonesty** - humans are prone to lying and in general we are not very good at telling the truth. In fact most people get away with lying because they thoroughly believe that what they are lying about is the absolute truth *ahemTrumpahem*
- **Limited focus and misdirection** - as humans, we aren't able to focus on many things. Instead we have a series of heuristics that we follow to find the most important things to focus on. Because of our lack of focus, we are easily subject to misdirection (especially when they distract us with gestures, eye contact, sounds, etc). To simplify, we focus on what grabs our attention.
- **Heuristics**
    - **similarity matching** - match whatever is happening now to what has happened in the past and you are very likely to act similarly to how you did before
    - **frequency gambling** - if many patterns match, you will pick the one that was most used in the past

  Why do we do the above? Because our brain likes to act in a way that requires minimal thought and the easiest way to do that is by doing what you've done before.
- **Confirmation bias** - we ignore what we don't like and only focus on evidence that support our claims
- **Satisficing** - instead of maximising something, we only do was is good enough or satisfactory
- **Group-think** - when you value your membership in a group, you are more likely to follow the consensus of the group even if you do not personally agree with it just so that you will not *lose your value* in the group

Just Culture

Just culture refers to a values-supportive model of shared accountability. It's a culture that holds organisations accountable for the systems they design and for how they respond to staff behaviour fairly and justly.

A just culture recognises that individual practitioners should not be held accountable for system failings over which they have no control. Instead look at the system as a whole and figure out what needs to be changed.

What are the properties of engineering we can apply to security engineering?

1. Learning from the past
2. Methodologies
    - Having standards as a baseline of what should be accomplished. This prevents many minor things from being overlooked
3. Redundancy and testing
    - Defence in depth
    - Dual control
4. Taking pride in what you do
5. Focus on process - understand the process rather than what you are doing
6. Review
    - Review your process

    - Review your tests
    - Have others review your work (independent, peer review). This removes bias and blindness to your own work
7. Professionalism
    - Do your job because of your duty to the profession
    - This is different to your duty to your position (doing your job for your boss)
8. Quantify things - have a way of measuring things. But be careful; not everything worth measuring is measurable and not everything measurable is worth measuring.
9. Closing the loop - most systems are *feed forward* systems with no feedback. Make sure that when something is done you have a way of knowing what the outcome of what you did.
   You want wise thinking not wishful thinking.

Systems

Properties:

1. Coherence - every unit in the system works towards something
2. Not complex - it is very easy for us to build reactively. If we encounter a problem we just patch it up instead of reviewing the whole system and taking account of the new problem. This can eventually lead to a complex system.
3. Not tightly coupled - no common method of failure and changes in one unit do not affect another.

Why do we keep humans in the loop?
Because automatic safety devices can't solve everything. Humans build them and our abilities are bound to be flawed. If we develop ASDs that can replace humans they must be able to solve every conceivable problem, which is impossible.
ASDs are good as a baseline of how to deal with problems we are likely to encounter but not a definite solution to every problem.

Security is ***end-to-end***. You want it to be secure from beginning to end. Any part that is insecure becomes vulnerable to attack.

Zero Knowledge Proofs

Privacy in the Modern Age: We're Screwed

Issues around data collection:

- ***Data linking*** - there is a network effect of linking data sources. Around 63% of the population can be uniquely identified by a combination of their information
- ***De-anonymisation*** - all anonymised data runs the risk of being reversed
- ***Massive scale*** - we are collecting an immense amount of data with the risk of creating a single point of failure
- ***Massive data breaches*** - the more data we collect the greater impact our breaches will have
- ***Fishing expedition*** - data can be looked through and used for unexpected purposes. e.g. an unnamed blue bank went through employee's history to look for reasons to fire them

Private Browsing and VPNs

Private browsing (or ***incognito mode***) is a feature on web browsers that lets you disable browsing history and web caching and cookies. This lets you browse the web without storing local data that could be retrieved at a later date. Private browsing will still leave traces on the hard drive and memory of a device. Websites can also associate visits with your IP address at their web server.

VPNs is a connection method to add security and privacy to private and public networks. A VPN server is a third party that connects to the web on your behalf. As a result it can:

- Hide your IP address
- Change your IP address
- Encrypt data transfers
- Mask your location
- Access blocked website - such as those blocked by governments

Even with private browsing and VPNs, ***fingerprinting algorithms*** can also map large amount of other non-browser related data during your browsing to you. This can include your screen size, operating system, time of access and other information.

Who's collecting data?

- Private industries
- The government
- Intelligence companies

Strategies used to anonymise data:

- Redact data
- Encrypt or hash the data
- Use pseudonyms (like a hash but non-deterministic)
- Statistical noise/'binning' - throw in excess data to throw off people's deductions or use a bound of data ranges instead of being specific
- Aggregation - combine a collective group of data to represent *one* person

All these strategies are breakable!

Anonymisation is never secure because all data has some sort of identifiable record (metadata)

What can we do about privacy:

- Start with engineers - make them understand the ethics around data
- Advocacy - educate others, complain if you witness a breach of privacy
- Design - design systems that strikes a middle ground; let people opt out but don't let no one use your services
- Story telling - privacy gains the most attention when witness its consequence on media platforms e.g 1984 and other films about data breaches