Asset

Anything within the organization that is worth protecting:
Information, Systems, Devices, Facilities, Personnel, Intellectual Property

Confidentiality:

Military / Government confidentiality:

- Top Secret: drastic effects / grave damage to national security
- Secret: significant effects / critical damage to national security
- Confidential: noticeable effects / serious damage to national security
- Sensitive but unclassified: internal use
- Unclassified: public data

Commercial / private confidentiality

- Coonfidential / private: drastic effects on the competitiveness
- Sensitive
- Public

In general classified is used as a term to describe anything but public data.

Deleting Data

There are multiple ways to delete data with massive difference in effectiveness:

- Erasing: removes only the link to the storage point, actual data remains
  Simple tools can recover this data!
- Clearing: Overwrites the data to delete it, usually with a single character
  Can not be recovered using simple tools
- Purging: Clearing, but overwrites multiple times to make recovery harder
  This method might also include others such as degaussing. It is Irrecoverable
- Degaussing: Strong magnetic field that can wipe data from an HDD
  (SSD, CD, etc not affected!)
- Destruction: Destroy the physical hardware of said data. Irrecoverable.

Tracing and Hiding sensitive Data

- Steganography: Embedding a Message within a file
- Watermarking: unique identifier, that usually can't copied, or mutated.
  Often used in Documents, movies, etc to stop counterfeits.

STRIDE Model

- Spoofing: Using a false identity to gain access to a system.
- Tampering: unauthorized changes/manipulation of data
- Repudation: The ability to deny having performed an attack,
  to others being blamed.
- Information Disclosure: unauthorized revelation of classified
  private information.
- Denial of Service: Prevent or restrict access to a service by flooding it.
- Elevation of Privilege: Gaining unauthorized privileges on a system
  For example: becoming root as regular user

Copyright

Protects these works: Literay, musical, dramatic, choreographic, graphical,sculptural,audiovisual, recordings, architectural works
Computer software is in the literary works category
Note only the source code is protected, not the idea itself.
Implicit Copyright is automatically granted if you are the creator of said work.
The explicit copyright can be obtained from the government and
lets you use the symbol: ©
This copyright lasts for 70 years after the death of the last copyright holder.

Trademark

Protects: Slogans, Logos, words used to identify something
Implicit Trademark is automatically granted if you use the ™ symbol
Explicit Trademark has to be obtained from the government
and lets you use the ® symbol.

Patents

Patents protect intellectual Property for 20 years.
For a patent a product must be useful,new,not be obvious
After the 20 years the patents enters into the public domain.
Patents are the worst thing ever.

Trade Secrets

Copyright, Patents and Trademarks require you to disclose what is protected
Because of this, many companies simply hide this information from the public
This means, while you are not protected by the law, as long as this information stays within your company, it will be protected forever.
Likely the "best way" to protect computer software, aka proprietary trash

CIA triad

The primary goal of Security infrastructure:

- Confidientiality: prevention of unauthorized access to data
  no matter if the data is in transit, in storage or in process
  Usually done with encryption and access control
- Integrity: prevention of unauthorized alteration of data
  Data integrity: data is complete, consistent and accurate
  System integrity: System only does what it was intended to do
  Usually done with hash verification, intrusion detection
- Availability: Systems should be accessible at all times
  Usuall done with Redundancy, backups

Nonrepudation & Accountability

Nonrepudation ensures that every action can be traced to the actual source
This prevents attackers from covering their actions.
Accountability ensures that every being is responsible for their actions.
E.G. Nonrepudation ensures Accountability.
Usually done with certificates, session identifiers, logs, etc

Data Classification

- Data in Use, Transit, Rest
  used by application, tranit via x, rest = storage, HDD, SSD
- Personally Identifiable Information PII
  information to identify a person, name, social security number etc
- Protected Health Information PHI
  Health Information recorded in any form.
- Proprietary(Trash) Data
  microtroll windoof, appul

Threats

- Threat: A potential danger to an asset
- Threat intelligence: Knowledge about emerging or existing threats
- Thrat event: Accidental or malicious exploitation of a vulnerability

Vulnerability

- Common Vulnerabilities and Exposures (CVE)
  Industry wide standard identification number for Vulnerabilities
- Common Vulnerability Scoring System (CVSS)
  Uses the CIA triad to score vulnerabilities on severity

Risk

Assessment of the possibility that a threat will exploit a vulnerability

Realized Risk

A threat actor has now taken advantage of a vulnerability
The whole point of security is to stop exactly this.

Risk Management

- Identifying vulnerabilities
- evaluating importance of data and countermeasure cost
- Implementing cost effective countermeasures

Risk management is the balance of threat/risk and usability
In other words, only implement measures that are necessary
Some data is not worth protecting, and some systems can be replaced easily.
Others are sensitive, or fundamental, these must be preserved at all costs.

Risk Analysis

- Evaluation, assessment, assigment of value to assets
- Examining environment for risks
- Evaluating the likelihood of a threat event occurring
- Assessing the cost of countermeasures
- present cost/benefit report to upper management

Asset Valuation

The representation of an asset in currency
This can include things such as repair costs, maintenance costs,etc

Exposure

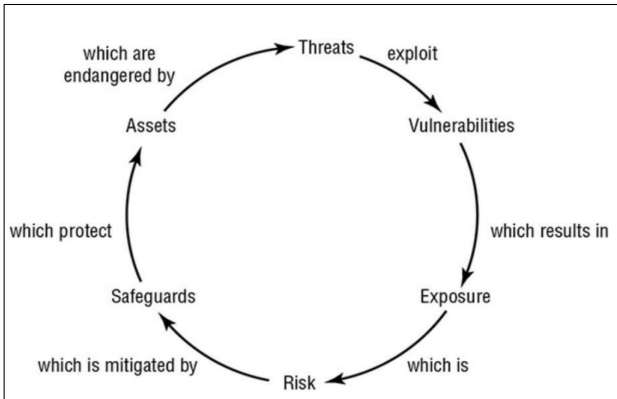
Possibility of an asset loss due to a threat
exposure factor (EF) checks how serious that loss would be

Attack

The intentional try to exploit a vulnerability

Breach

The circumvention of security measures by a threat actor.
A breach combined with an attack, can result in penetration



Quantitative and Qualitative Risk Analysis

–Quantitative Risk Analysis:

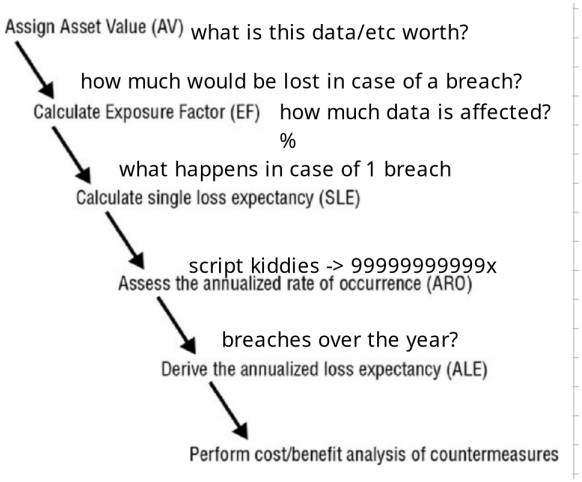
assignment of real dollar figures to loss of assets

–Qualitative Risk Analysis

The subjective / intangible worth value to the loss of assets

Both methodologies are necessary for complete risk analisys!

Quantitative Risk Analysis



– Exposure Factor (EF): percentage of loss by realized risk

– Single Loss Expectancy (SLE): Cost of single realized risk

SLE = EF * Asset Value (AV)

– Annualized Rate of Occurrence (ARO):

expected frequency of a risk within a year

– Annualized Loss Expectancy (ALE):

Expected yearly cost of Losses due to realized risks

ALE = SLE * ARO

If you implemented a safeguard, you have to recalculate the ARO.

The entire idea of of security is to reduce the ARO!!

The EF usually remains the same

– Safeguard Costs

First, compile a list of safeguards against each threat.

Assign each safeguard a deployment value -> Annual Cost of Safeguard (ACS)

– Safeguard Cost/Benefit

ALE without safeguard - ALE with safeguard - ACS = Value of safeguard

if the value of safeguard is below 0, then it is financially irresponsible

Note that this only takes in the financial damage since this is Quantitative!

Dealing with Risk

– Risk Mitigation Implementation of safeguards in order to eliminate vulnerabilities or block threats

– Risk Assignment / Risk Transferring

purchasing insurance or outsourcing. Transfer the cost of risk to other entity

– Risk Acceptance Doing nothing as cost/benefit would be low

– Risk Deterrence auditing, cameras, security guards, warnings, etc

– Risk Avoidance Not using the system associated with the risk

– Risk Rejection Simply ignore risk without cost/benefit analysis!

– Residual Risk A countermeasure might not fully eliminate a risk

This is the remaining risk that we have decided to accept.

Privacy

– GDPR
  - Companies have inform authorities in case of serious data breaches
  - Individuals have the right to demand their data from companies
  - Individuals have the right to be forgotten (deletion of data)
  - EU tries to enforce this globally
  - enforces pseudonomization

– Patriot act
  - blanked authorization of surveillance of an individual with 1 warrant
  - ISP have to provide data
  - easier wiretapping

– pseudonomization replacement of data with aliases

This makes it harder to identify a person from said data

– anonymization Complete obfuscation of an identity

Access Control

– Subject The Requesting party

Note, this can be a person, a program, a process, etc.

– Object The requested "object"

Databases, programs, files, etc

– Access Control The management of the relationship

Between Subject and Object

– Preventive Access Control

Stops unwanted access to Objects

– Detective Access Control

Detects unwanted access to Objects after it has occurred

– Corrective Access Control

Restores the last "correct" state after unwanted access

– Deterrent Access Control

discourages unwanted access to Objects

– Directive Access Control

Directive issued by company. -> Don't click on links.

– Compensating Access Control

Controls used in addition/ as a replacement. Can be any control measure

– Recovery Access Control

More advanced version of corrective control

– Physical Control

Control of items that one can physically touch

– Technical / logical Control

Hardware or Software mechanism for access control

– Administrative Control

Policies and Procedures created by a Company

Steps of Access Control

1. Identification

Providing an identity to enforce Nonrepudation.

is usually public information.

username, tokens, fingerprints, facial recognition

2. Authentication

Often used in combination with Identification.

Verifies the identity. Usually a password, or token.

3. Authorization

Controls what a user is allowed to do and what they aren't

There are different ways of controlling this.(Check next page)

4. Auditing

Record all actions by all Subjects in order to hold them accountable

5. Accounting (Accountability)

This would mean taking actions against a person,

that has acted in bad faith/maliciously

Keep in mind that you need a strong system of identification,

and auditing

in order to actually win in a court of law against the bad actor!

Authentication Factors

– Type 1: Something you know

– Type 2: Something you have, ex. a Device, smartcard, etc

– Type 3: Something you are/do, ex. Fingerprint, face, etc.

Type 1 is the weakest form of authentication and type 3 the strongest!

– Location

This is a form that is used in combination with others.
Ex: A certain IP might be a requirement to log into one of your systems
Or your bank might block a transaction if it isn't from your country of residence.

Multifactor Authentication

Multifactor Authentication simply uses more than one type to authenticate
This might be a password(Type1) and a token(Type2)
The strongest form of authentication if therefore all 3 types together!

Passwords

Passwords are never stored in plain text,
they might be stolen easily with 1 breach
Instead, they are saved with a hash-algorithm,
makes them nearly useless to threat actors
Unless they also have the access to said algorithm.

Types of passwords:

– Plain Password
  use numbers, characters and special symbols with a length of at least 10.
  10 characters -> 928 years of cracking!!
  Don't use personal information such as names etc.
– PassPhrases
  Passphrases are chained words that are easy to remember
  It is important to have a longer passphrase than a regular password!
  Still use special symbols. Also try to include random upper-lower case spelling
  Or leadspeak s1nc3 that 1s qu1te 3ff3ct1v3!!
– Cognitive Passwords
  A series of personal questions. -> what is the name of your first pet?
  Best way to do this is letting users create both the question and the answer!
– SmartCards
  Card used for identification / authentication. Often integrates key encryption
  Usually temper resistant
  One downside, loss of card might give a threat a window of exploitation!
– Tokens
  Generated by a special Device.
  Regular password generators: any device can take that place. ex. smartphone
– Synchronous Dynamic Token
  Time-based One-Time Password (TOTP)
  Device generates Token every x seconds.
– Asynchronous Dynamic Token
  HMAC-based One-Time Password (HOTP)
  Device generates one time token based on algorithm. Stays until used!

Access Control Models (Authorization)

– Discretionary Access Control (DAC)
  Every object has an owner, and that owner can to grant or deny permissions
  NTFS from windoof uses this
– Role Based Access Control
  Permissions are assigned to roles not users. Users are assigned to roles.
  Users who are in a role with said privileges can use them.
– Rule Based Access Control
  Global rules that are applied to all subjects
  Good example is a firewall, which applies said rules equally to all subjects
– Attribute Based Access Control
  Similar to Rule based Control but with additional attributes
  This could give one subject more rights than another
– Mandatory Access Control
  Use of labels applied to both subject and Object
  If user has the same label as a file, then user has access to it.

Authorization Mechanism

– Implicit Deny
  Deny everything that hasn't been specifically allowed
  Most used!
– Constrained Interference
  Applications might hide functionality based on the privileges of a user
– Access Control Matrix
  This writes Objects,Subjects and privileges into a table
  If Subject tries to access an object the table for said object is checked
– Capability Table
  This is the same as the Access Control Matrix but with a subject focus
  In this table the subject and all accessible Objects are written down
– Content-Dependent Control
  Constrict Access to the data within an Object
  In a database a user might be able to check table 1 but not table 2.
  While the object is the entire database!
– Context-Dependent Control
  Give a subject access depending on what the subject does
  Ex. The checkout button in an online shop only works, if you have something in the shopping cart.
– Need to Know
  Subjects should only have access to Objects they need to do their job.
– Least Privilege
  Subjects should only have the privileges they need to do their job.
– Separation of Duties and Responsibilities
  No single person should have total control over the entire System!

Common Access Control Attacks

– Access Aggregation Attacks (Passive Attacks)
  This is the collection of nonsensitive data, that combined could give a threat actor the opportunity to launch a proper attack.
  Ex. IP address, open ports, Operating System -> specific exploit
– Password Attacks (Brute Force)
  Spam random sequences until you get the right one
– Dictionary Attacks (Brute Force)
  Try passwords from a list of passwords, example leaked password list.
  Can also be done with list of common passwords, or slightly changed previous passwords (One-Upped-Passwords -> 1 character changed)
– Birthday Attack (Brute Force)
  Try to get the same hash as the password with a different sequence
  Can be mitigated by using better hashing algorithms. SHA-3 instead of md5
  Note the attacker needs access to the hash in order for this to work!
– Rainbow Table Attacks
  Combines the Birthday attack with a table of precomputed hashes.
  This is then used to compare to a password hash list.
– Sniffer Attacks
  Threat actor analyzes data sent over network with a sniffer tool.
  A good example for this is wireshark
  Can be mitigated by using encryption and One-Time passwords
  encryption makes the data useless and One-Time passwords are as well
– Spoofing Attacks
  Pretending to be something/someone else. Ex. pretending to be router.
– Social Engineering Attacks
  Gaining and then misusing trust of someone.
  Indian accent you get refund if you buy me 2 cards from target
– Shoulder Surfing (Social Engineering)
  Reading information on a screen from a persons back.
– Phishing (Social Engineering)
  Trick a Person to click on a fake link to log in, giving the attacker all the credentials to log-in
– Spear Phishing (Social Engineering)
  Targeted Phishing at a group. Ex. Employees at company x.
– Whaling (Social Engineering)
  "Phishing für grosse Fisch" -> CEOs etc
– Vishing (Social Engineering)
  Phishing via VOIP or instant messaging

Protection Mechanism

– Layering (defense in depth)
  Multiple Controls in Layers, if one fails, there are still the other ones
– Abstraction
  Combining Objects into groups in order to simplify permission management

– Data Hiding
Storing Objects in compartments that can't be seen / accessed by an unauthorized subject
– Security through Obscurity
Not informing a subject about an object, and hoping it will stay hidden
– Encryption
Turning data into gibberish via algorithms.

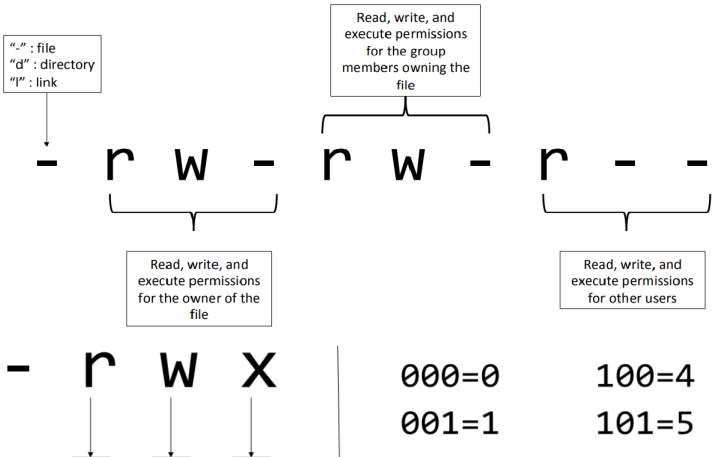
Red-Team
Offensive Cyber-Security: simulate attacks
– Think outside the box
Find new ways and tools and attack systems to show the flaws
– Deep Knowledge of Systems
Deep Knowledge about systems, flaws, exploits, methodologies,etc
always up-to-date with technology
– Software Development
Learn how to develop your own tools.
– Penetration testing
Identify vulnerabilities and potential threats
– Social Engineering

Blue-Team
Defensive Cyber-Security: prevent attacks
– Organized and detail-oriented
Prevent gaps by thinking about EVERYTHING
– Cybersecurity Analysis and threat profile
Assess the security of an organization. Create Risk/Threat Profiles.
– Hardening Techniques
Reduce the attack surface hackers might exploit
– Knowledge about detection Software
Be familiar with software that recognizes unauthorized actions
low skill application would be rkhunter.
– Security Information, Event Management (SIEM)
Software that allows real-time analysis of security events

Yellow-Team
Builds defensive software against attackers

Purple-Team
Improves organization security posture

Linux
Adding user: usermod -m username -s /path/to/shell
Change shell: chsh -s /path/to/shell
Change password: passwd username
Add user to group: usermod -a -G groupname username
Change file permission: chmod permission file
Change file owner: chown file user/group
Check IP address: ip addr / ip -c -brie a
DNS query: dig domain (dig shitgaem.online)
– File System Permissions
r = read, w = write , x = execute
Every single file has these attributes.
These attributes are also duplicated for 3 different types of users.
1. owner, 2. group of owner, 3. other
This means the actual permission would look like this:



Manually set IP address (abando): edit /etc/network/interfaces
Configure SSH: edit /etc/ssh/sshd_config
For birbs sake, use a nonstandard port for ssh :)
Check current shells: ps
Check current processes: htop
grep read lines
grep 'Warning' /var/log/rkhunter.log
Read Lines: awk 'sshd.*invalid user/ {print $11}' auth.log
bit-by-bit copy: dd if=<media/partition> of<image_files>
mount - o ro,noexec,loop evidence_01 /mnt/investigation
mount with read only and no execution
Check recently changed files ls -lasrt

PID
The unique identifier the kernel gives each process
This shows both background and foreground applications

logs
Logs capture every single action on linux, this can be used to detect bad actors.
However, logs can also be spoofed, which means you always have to be sure, that everything is being logged, and that no logs have been tampered with.
notable logging systems/files: syslog, rsyslog,var/log ,auth.log
The shred command with -f and -n force deletes log files.
Mainly used like this: shred -f -n 15 /var/log/auth.log*
This shreds 15 lines from every log file with the name auth.log(something)
other things to look out for:
– set-UID Rogue Files
– Directories with .something Hidden...
– Regular files in the /dev directory
– Recently modified files ls -lasrt

Schedules Tasks
Schedules tasks can be written either in cron.d or with systemd

IP-Tables
name one reason not to use ufw...
Flush all rules: iptables -F
Block Input: iptables -P INPUT DROP
Block Output: iptables -P OUTPUT DROP
Block Forward: iptables -P FORWARD DROP
Allow Port: iptables -A INPUT/OUTPUT -p port ...other shit...
Show rules: iptables -L -n -v – line-numbers

Sticky Bit
This is a single bit in front of rwx. -> 1777 sticky set, 0777 sticky not set
The interpretation of this bit depends on the file type
For directories, it means that any files within that folder
May only be renamed or deleted by the owner.
For files this bit is deprecated!

Security Enhanced Linux (SELinux)
This is a module created by the NSA that implements types,
which mark files based on the type of a subject.
Ex. a top-secret process can create a file with chmod 777,
but a confidential process still can't open it.
This is called MLS in SELinux and is related to Multi Category Security (MCS)

Snort
Detection software like rkhunter

NetCat
This can be used for anything dealing with TCP and UDP.
You can also use it to control compromised systems...

Reverse Shell
The idea is, since the starting connection comes from the victim,
Not only do we not have NAT and firewall problems, the connection
also looks more legit than when we connect.
This gives a hacker some sort of legitimacy on that system.

Scapy
Tool used to send, sniff, dissect and forge IP packets.
You can probe, scan and attack networks
You can attack signature for IDS/IPS systems

Encryption Terms

- Plain Text: unencrypted message
- Ciphertext: encrypted message
- Cipher: Algorithm used to encrypt
- Cryptographic key: Just a number to decrypt a message
  The range is defined by the algorithm. 0 to 2^n
  A key with 128 bits would have a range of: 0 to 2^{128}
  It is critical to keep the keys secret!
- One-Way Function:
  mathematical function that produces output in a way that
  input can't be retrieved.
  There is no TRUE One-Way-Function
  Cryptography works on the believe that it can't be broken RIGHT NOW
  However, this does not mean it will stay so forever, see already broken ciphers
- Reversability: The option of encryption....
- Nonce: A Public,unique One-Time-Use Number
  Makes sure a key is not re-used twice!
- Initialization Vector (IV): A random bit string
  Same length as the block size and is 'XORed with the message'
  IVs are used to create a unique ciphertext with the same key
- Confusion:
  This is the case when encryption is so complicated,
  that merely reforming the string doesn't reveal the message
  Aka bruteforce doesn't work anymore.
- Diffusion:
  A change in the plaintext will result in multiple changes in the ciphertext.
- The Kerckhoff's Principle
  This means everything about the system is public but the key
  It therefore requires the system to be secure even under these circumstances
  The idea is that public algorithms may hasten the improvements on them
- Permutation Swapping Bytes around
- Byte Substitution Replacing bytes with others
- SP-Networks algorithm that uses repeated Permutations and Substitutions
  Permutations and Substitutions are combined to a round
  Rounds are then repeated many times

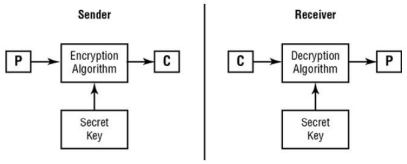
Caesar Cipher or ROT3

One of the earliest encryption systems
Simply shifts a chracter by 3 A to D, B to E...

One-Time-Pad

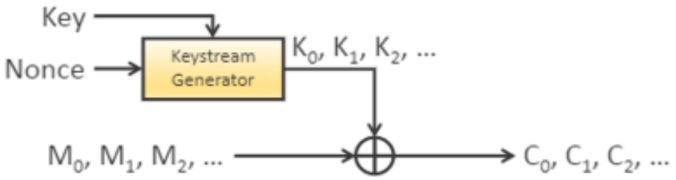
Create a key with the same length as the message
XOR each message bit with each key bit
This Cipher is UNBREAKABLE!
However it is not practical.. 1GB file 1GB key...
No proper way to transmit, store a key
Using a key twice == Cipher broken

Symmetric Cryptography



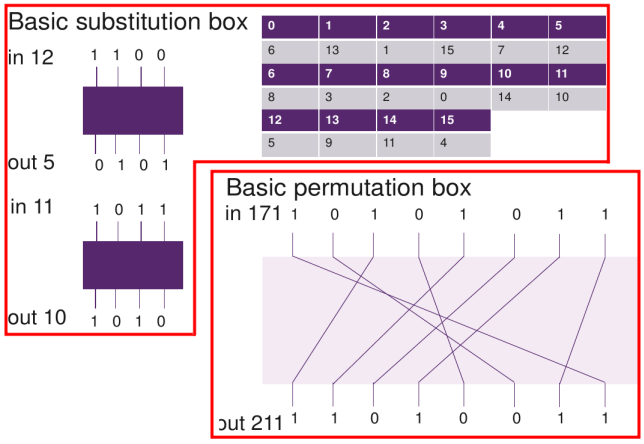
- Same key for encrypting and decrypting
- Shared key for all parties involved!
  If one leaks the key, the cipher is broken!
- Doesn't confirm identity!
  Anyone who has the key can pretend do be another

Stream Ciphers



+ Encryption of long continuous streams, of possible unknown length
+ Extremely fast with low memory footprint, ideal for low power devices
+ If designed well, it can seek to any location in the stream
- The keystream must appear statistically random
- You must never reuse a key + nonce
- Stream ciphers do not protect the ciphertext (no guaranteed integrity)

Substitution / Permutation box



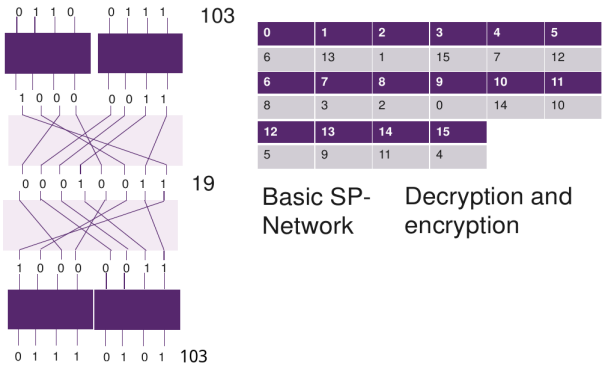
Block Cipher

Takes in an input of a fixed size and returns an output of the same size

- Diffusion and Confusion
- SP-Network

Advanced Ecryption Standard (AES) is a Block Cipher

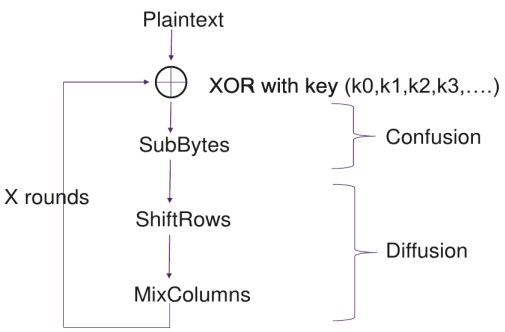
Here is a Block Cipher works:



AES

Built around the Rijndael algorithm
Superceedes the DES as a standard

- SP-Network with 128-bit block size
  - » Key length 128,192,256 bit
  - » 10, 12 or 14 rounds
  - » Each Round: Substitute Bytes, ShiftRows, MixColumns, KeyAddition



Block Cypher with random input length

Obviously we want to encrypt more than just one block

How do we do that?

– Electronic Code Block (ECB)

– Cipher Block Chaining (CBC)

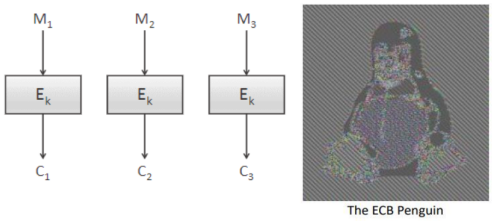
– Counter Mode (CTR)

Electronic Code Block (ECB)

Just encrypt block after block.

However, this might give away the bigger picture

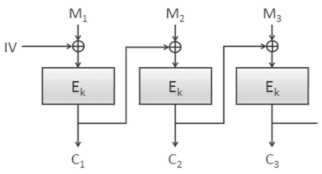
Aka the pattern of the data is still visible!!



Cipher Block Chaining (CBC)

XOR each output with the next block.

not parallelizable, but more secure than ECB



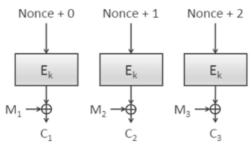
Counter Mode (CTR)

Encrypt a counter (Nonce) to produce a stream cypher

Encrypted Nonce is then XORed with the plain text

parallelizable!!

Standard for all AES ciphers!



Cons and Pros of symmetric Cryptography

– Key distribution

Keys have to be shared securely, anyone who has the key can encrypt and decrypt all messages (sent by that key)

– No Nonrepudation

Because everyone who has the key can encrypt and decrypt, there is no guarantee that this message is from a trusted source.

– No message Integrity

If the message gets damaged, then there is no recovery inbuilt.

\+ Speed

often 1000 to 10000 times faster than asymmetric algorithms

Lots of processors have an AES intruction set.

Alternatives: Chacha20 cipher

Diffie-Hellman

With this method the problem of sharing a key over the internet is solved

We can now do so without any worries of giving a malicious third party access

Every TLS handshake is in some way powered by this.

We are not actually exchanging a key, only some mathematical part of it!!

Discrete Logarithm

A logarithm that is implicit when using mod.

$a^b (mod\, n) = c == b = \log_{a,n}(c)$

These are harder to calculate than regular ones!

Which is why they are used in Diffie-Hellman!

- 3^x mod 7 = 1, what is x?

This leaves us having to brute force the answer

- Brute force:
  - 3^1 (mod7) = 3 (mod 7) = 3
  - 3^2 (mod7) = 9 (mod 7) = 2
  - 3^3 (mod7) = 27(mod 7) = 6
  - 3^4 (mod7) = 81 (mod 7) = 4
  - 3^5 (mod7) = 243 (mod 7) = 5
  - 3^6 (mod7) = 729 (mod 7) = 1

What if mod 7 was mod some 2000 bit number

Primitve Root

A number g is a primitive root of p when:

$\bigvee_{k=0}^{p} g^k\, mod\, p$ = Distinct from each other

In other words, every single result from the modulo must be different!

Diffie-Hellman Example

1. Agree on Parameters

Alice and Bob agree on a large prime p and a second prime / primitive root g

p is usually at least 2048 or 4096 bits

2. Select Private Numbers

Alice picks the random number a

Bob picks the random number b

» private numbers are between 1 and p

» If p is 2048 bits, then you are guessing a number with 2048 bits, have fun :)

» They NEVER tell each other the private number

3. Alice and Bob each calculate a Public Key

» Alice calculates key: $g^a\, mod\, p$

» Bob calculates key: $g^b\, mod\, p$

Because we are using a discrete logarithm, it is mathematically infeasible to get the private numbers by calculation.

4. Alice and Bob exchange the Public Keys

These are simple the calculated versions of keys.

5. Alice and Bob calculate the shared key

for both this is: $g^{ab}\, mod\, p$

The shared key is therefore the same for both parties

6. Calculate Master Secret

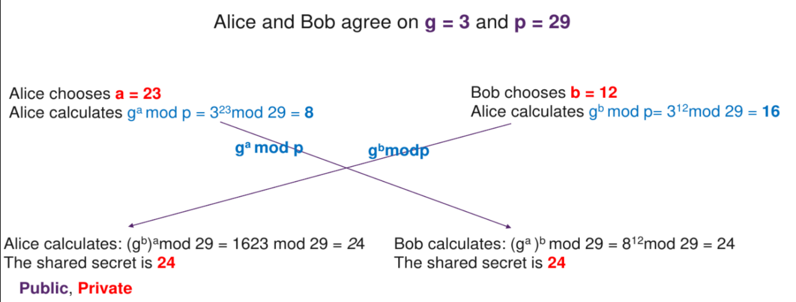
The shared key is also called the Pre-Master

This is because the shared key is quite big and

not often used to encrypt directly

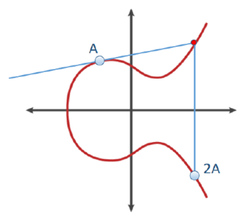
It is instead used to control sessions after it has been hashed

The hashed shared key is then called the Master Secret



Eliptic Curve Cryptography

- Elliptic curves are a drop-in replacement for the mathematics underpinning regular Diffie-Hellman.
  - If you look into your browser, you see something like ECDHE
  - Elliptic curve Diffie Hellman is now becoming the standard
- Elliptic curve is a two dimension curve
  - y2=x3+ax+b
  - The private key is a number
  - The public key is composed of two numbers(X,Y) number



- Elliptic curves are much stronger than traditional public-key schemes for the same key length.

Symmetric	Diffie-Hellman and RSA	Elliptic Curve
56	512	112
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

Ephemeral Mode

For Diffie-Hellman this means calculating a new key for every session
This is also called Perfect Forwarding Secrecy
The reason for this is that calculating this key costs close to no power
So why not just create a new one for every session to increase security?
This is usually done every browser refresh etc.
It is also automatically done after a certain time, again to improve security

RSA Rivest-Shamir-Adleman

- Public Key cryptosystem
- widely used for secure data transmission
- Most common method for public cryptography
- Offers Nonrepudation!!
- Reverseable Keys

  Both the public key and the private key can be used to encrypt or decrypt.
  It just has to be the inverse at the other end.
  Encrypt pub-key -> decrypt priv-key | encrypt priv-key -> decrypt pub-key

» The reverseable keys lead to 2 operating Modes

1. Encrypt so only the receiver can read

If I want to send a message to the server that only said server can read, then I can encrypt my message with the servers public key.
The server then uses its private key to decrypt. Not even you can decrypt it :P

2. Nonrepudation Mode

By encrypting with the private key, everyone who has the public key can read.
However, it guarantees that said message is from you, and no one else!

Prime Factorization

Every non-prime number has a prime factorization
This means every non-prime number can be created by multiplying x primes
Prime factorization of 30 -> 5 * 3 * 2
Calculating the prime factorization is EXTREMELY HARD for big numbers
In other words, it is not feasible to calculate it with a current Computer
Which is why it is used by the RSA algorithm

RSA Functionality

Public keys: e n
Prime factors: p1,p2
Private key: d
Message: m

e is almost always 3 or 65537
n is a random Prime factorization of 4096 bits

$$d = \frac{k * \phi(n) + 1}{e}$$

where k is a random integer
p1,p2,d must be private!!

- Encrypting Public: $c = m^e \mod n$
- Decrypting Private: $m = c^d \mod n$
- Encrypting Private: $c = m^d \mod n$
- Decrypting Public: $m = c^e \mod n$
- Combined $m^{ed} \mod n = m$

The PHI Function

- Euler has studied the distribution of prime numbers and invented the PHI function
  - ϕ(n)
  - It outputs how many integers are less than or equal to n that do not share any common factor with n.

ϕ(8)= 4   1 2 3 4 5 6 7 8   ϕ(7)= 6   1 2 3 4 5 6 7

In red, the numbers that are relatively prime with n.
coprime

counts for prime:
PHI = prime - 1
Φ(21377)= 21376

- Euler theorem

$m^{\phi(n)} = 1 \mod n$ — $m^{e*d} = m \mod n$

$1^k = 1$
$m^{k*\phi(n)} = 1 \mod n$
$m*m^{k*\phi(n)} = m \mod n$
$m^{k*\phi(n)+1} = m \mod n$
where k is an integer

$\Rightarrow ed = (k*\phi(n)+1)$
$d = (k*\phi(n)+1)/e$

- It is computationally infeasible to calculate a private from a public key.
- This is achieved through intractable mathematical problems.
- In order to calculate ϕ(n) easily, you have to know p1 and p2 where p1*p2=n

Using RSA

1. Choose two very large Primes

- Alice randomly generates a prime number p1
  - p1=

1838476548476453627840298364538745366746352928736539382279438292033874693937373798201992394739099928334411737377921100388376547382653873638268837628763887439987367922029874635373829288393829476555281818330900309887654337893393875648382719485736

- then a second randomly generated number p2
  - p2=

38475647288888276199100375628193726591747629461976654112345622998674756292756278292847390019384769992000029282772734777629984110009385769392938593985909300290000093410009258110000094558666688332295668549285800000000000003333000556600000000000072

- Alice calculates n
  - n=p1xp2

1948475948398729829989937500022229918855587630022911134949494498739995577575799222200000000000445500000000000399988587757777756663399299229292929292929299299477657675488568887887873000010002948575873847848398999999999285478567575837829000000111
1111184477566382947663492947875849828794888000029284858758485484858458847283857362887562094857391111100928485669001929838475854949999938857466646464646283783837378373828878827383897838778877782828348838883888873788877487588375858588585557849865 48

- Alice hides p1 and p2 (p1 and p2 are secret)
- n is public

2. Calculate PHI

$\phi(n) = \phi(a) * \phi(n)$
a and b are the prime factorization primes!
if number is prime, then: $\phi(a) = a - 1$
Ex. n = 77, a = 11, b = 7 -> $\phi(77) = \phi(11) * \phi(7) = 10 * 6 = 60$

3. Choose k and e to calculate d

$$d = \frac{k * \phi(n) + 1}{e}$$ with k being an integer

Ex. n=55, e=7, k=4, $\phi(n) = 40$ $d = \frac{4 * \phi(55) + 1}{7} = 23$

4. Encrypt and Decrypt...

RSA quirks

- Very weak with short messages

  To mitigate this, padding is added
  Optimal assymetric Encryption Padding (OAEP) is used
  pseudo random padding that introduces an IV then hashes it
  Server has to create same padding to check if it matches up
- Not common to encrypt with RSA!

  TLS used RSA before but no longer.
  RSA is used more for signing! Something that Diffe can't!!
- RSA is 1000x slower than symmetric crypto systems!!
- Message Integrity
  - » Message is first hashed to shorten it
  - » Said hash is sent alongside the message
  - » Receiver hashes the message as well on arrival, to check whether the 2 hashes match

  Simply decrypting the hash is sadly not possible.

Digital Signature Challenges

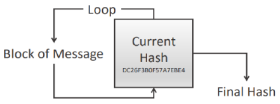
Before communicating with a server, we want said server to prove its identity, we do this by sending the server a message that the server then has to sign with its private key.
This challenge is also part of Transport Layer Security (TLS)

Digital Signature Algorithm (DSA)

RSA will soon become unusable due to the ever growing messages.
In order to solve this the DSA has been developed, which is much faster!
The downside is that it can't be used to encrypt, but we have Diffie for that
It also supports Elliptic Curves as it has similar mathematics as Diffie

Hash Function

- Fixed Length
  A hash function always has a fixed length. Meaning that if we have a 128 bit hash function, then it will always return 128 bits
  no matter how many inputs we enter
  - Iterative
    We iteratively hash over the message
    by block until we get to the end
    Then we take the current hash as the final hash.



- Indistinguishable from Noise
  The output of a hash function should not be interpretable.
- Diffusion
  Best way is to make small changes in the message
  in a big change in the hash.
- Balance between speed and security
  A good hash function needs to be quick enough to use
  But also needs to offer proper protection!!
- Needs to be infeasible to Revert
  Just like with encryption, there is no way to guarantee irreversibility
  We can however make it infeasible to do this with current computers!!
- few Collisions
  The best would be to have 0 collisions, but that is impossible to guarantee
  Instead we simply push the probability to a very, very small number!
- Used for Integrity
  Often a message gets corrupted, or even attacked by a threat actor
  In both cases we need a way to check integrity.
  We do this by hashing the message and sending the hash alongside it
  The receiver can then hash the message as well and compare the hashes.

Current Hashing Algorithms

- SHA-2 256/512
  Current standard!
- SHA-3
  Same quality as SHA-2, designed as a backup in case SHA-2 is broken

Name	Output Length	Rounds	Security	
MD5	128-bit	4	Broken	Cryptography
SHA-1	160	80	Recent collision found, not trusted	
SHA-2	224, 256, 384, 512	64, 80	Some theories, currently considered safe	
SHA-3 (Keccak)	224, 256, 384, 512 SHAKE128, 256	24	Secure, but relatively untested, strength variable	
PBKDF2	Varies		Iterates another hash function	Password Storage
bcrypt	184-bit		GPUs struggle to crack it	

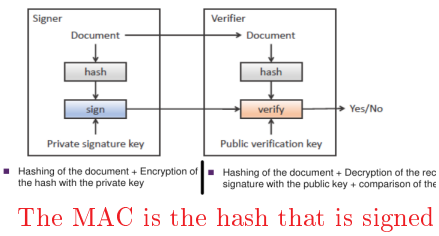
Hashes and Passwords

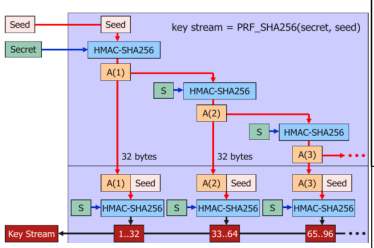
These hashes alone are not good enough for passwords!
Hacker will simly use hashed passwords to compare to your hash table
Solution: Password-Based Key Derivation Function 2 (PBKDF2)
This hashes the password with SHA-2 5000 times.
It is therefore 5000times slower, but also much more secure!
Inside password managers, the iterations go up to 100 000 times!
The alternative is called 'Blowfish'

HMAC and the length extension attack

using only 1 hash might give attackers the opportunity to
add more data to the end of the hash, as long as the hash is still "random"
With HMAC we split the message in half and hash each side with both
The public and the private key. This prohibits length extension attacks.
MAC means Message Authentication Code, not Multimedia Access Control lmao

Message Authentication Code (MAC)





The MAC is the hash that is signed

Certification Terms

- Public Key Infrastructure
  The entire Processof requesting, receiving and holding a certificate
  This means the Environment: hardware, software, policies, roles, procedures.
- Certificate Signing Request (CSR)
  A request to a CA to sign your certificate
- Certification Authority (CA)
  CA is the signing third party organization: letsencrypt, Google, Digicert, etc.
- Root Certificate
  Public key of a CA that will be used to decrypt and check the
  Certificates of websites. Note the CA must match!
- Trust Service Provider (TSP)
  An organization that provides Certificates, validation, etc
  letsencrypt, digicert etc.
- Trust Services (TS)
  All services of a TSP -> CA,VA,RA
- Validation Authority (VA)
  TSPs Validation part
- Registration Authority (RA)
  TSPs Registration part

Digital Certificate

A digital certificate is nothing more than data that includes:

- Information about the key
- The identity of the owner (subject)
- Issuer information

With public and private keys, we can only guarantee that the message
comes from PC x, but not that said PC is actually trustworthy!
Or that said PC is actually from the person it is claiming to be!!
This is where digital certificates come into play
They are signed by a third party and are therefore considered to be trustworthy
The third party is usually a Public Key infrastructure

Qualities of Certificates

There are different levels of certificate quality,
the higher, the more trust you can have in a certificate:

- 1. Domain Validated (DV) – 2.23.140.1.2.1
  It is issued after the owner hash shown proof that they
  have the right to use their domain. (automatic check)
  letsencrypt only offers this level! 80% of internet
- 2. Organization Validated (OV) –2.23.140.1.2.2
  This is issued after the CA has validated the company name,
  domain name and other information through public databases
  Used by bigger companies and Emails
- 3. Extended Validation (EV) – 2.23.140.1.1
  This requires strict authentication procedure
  Used by banks, Governments, etc
- 4. Qualified Website Authentication Certificate (QWAC)
  Not supported by browsers
  mixture of OV and EV, wanted standard

Certificate Issuance

1. Create a private/public key for RSA or DSA
2. Create a CSR and send it to a RA
   in Base64-PEM format
3. The RA does identification checks
   These vary heavily depending on the type of CA and
   the amount of trust you want to have from a client
   letsencrypt is free and easy to use, but offers only minimal identification!
   Digicert and others offer in person identification and liability
   However these also cost money and are more time consuming to create
4. The CA signs the public key, you now have a certificate
   Now every time someone visits your website, they see the
   certificate signed by the CA with their private key
5. The VA checks the validity of the certificate

Certificate Use

When you navigate to a website, you get the certificate of said website during a TLS handshake, after that you need to decrypt the certificate with public key of the issuer CA. This key is stored within your browser or OS
Then you can use this decrypted certificate, which in the end is just a public key To decrypt all messages sent by the server, which are now guaranteed to be from said server

This is called Chain of Trust!

If you don't have the Signing CAs public key stored, then the connection is rejected!!

X509 Certificate

```
Certificate ::= SEQUENCE {
    tbsCertificate TBSCertificate,  signatureAlgorithm AlgorithmIdentifier, signature BIT STRING }
tbsCertificate ::= SEQUENCE {
    version
    serialNumber,
    signature,
    issuer
    validity
    subject
    subjectPublicKeyInfo
    issuerUniqueID: OPTIONAL,
      -- If present, version MUST be v2 or v3
    subjectUniqueID: OPTIONAL,
      -- If present, version MUST be v2 or v3
    Extensions: OPTIONAL
      -- If present, version MUST be v3 }

AlgorithmIdentifier ::= SEQUENCE {
    algorithm OBJECT IDENTIFIER,
    parameters ANY DEFINED BY algorithm  OPTIONAL }

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension
Extension ::= SEQUENCE {
    extnID OBJECT IDENTIFIER,
    critical BOOLEAN DEFAULT FALSE,
    extnValue OCTET STRING
        contains the DER encoding of an ASN.1 value corresponding to the extension type
        identified by extnID }
```

- Certificate extensions are data fields completing the X.509 certificate.
- Depending on the use case of the certificate, extensions are mandatory: e.g. root CA certificate, signing certificate, TLS profile certificate, etc.

– Extension Requirements:

	Root CA	Issuing CA
TBS	✓	✓
Authority Key Identifier	✓	SHOULD
Subject Key Identifier	✓	SHOULD
Key Usage	✓	✓
	keyCertSign, CRLSign, Offline CRLSign	Depending on the certificate type*

– Serial Number

Positive Integer unique for each certificate issued by CA

– Signature Algorithm

Algorithm used by CA to sign the certificate

x509 Validity Checks

– Method 1: Certification Revocation List (CRL)

Most browsers perform this check
CRL Distribution point included in the certificate

– Method 2: Online Certification Status Protocol (OCSP)

The Authorative Information Access (AIA) field provides information about the CA.

Other Terms:

– Certificate Policies

Link to governance rules of the CA

– Authority Key Identifier (AKI)

Key identifier of the CA

– Subject Alternative Name (SAN)

various values associated with the certificate owner

– Subject Key Identifier (SKI)

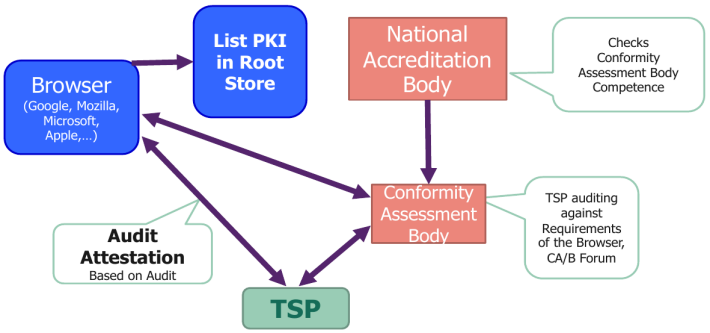
Hash value of the certificate

Certificate Pinning

When a website pins a CAs root, or intermediate CA, it forces the browser to only use this specific certificate
This prevents modified RootCAs from connecting to this website.
The problem with this is that, when the pinned CA is compromised, there is no check the browser can do to stay secure.
For this reason, the pinned CAs are usually only valid for 60 days.
Pinnable CAs: root CA, Intermediate CA, End Certificate (SSH)

Root Store and CA/B Browser Forum

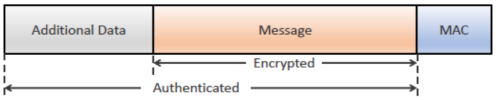
The root store defined what TSP/CA we trust.
But who decides what CA is in said store??
While every browser has their own Root Store Policy,
There is a forum that tries to unify this:



Authenticated Encryption with associated data (AEAD)

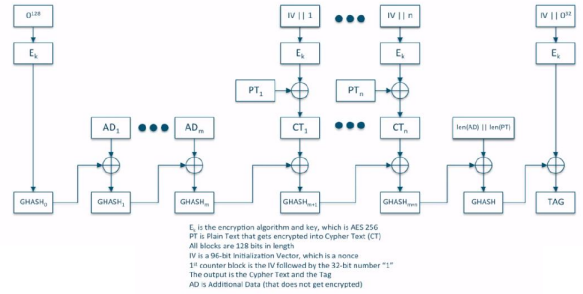
– is an AEAD protocol

In order to guarantee confidentiality as well as integrity,
we add a Message authentication code at the end of our message



One such MAC is the Galois Counter Mode (GCM)

It calculates a MAC over the message and the additional data



ChaCha20_Poly1305

– is an AEAD protocol

– allowed on TLS 1.3

The only one next to AES.

– needs a Nonce as ChaCha20 is a Stream Cipher

– Used for Android

It is used mostly for speed purposes as ChaCha20 is faster
when the CPU doesn't have an AES instruction set.

Protocol Handshakes

1. Handshake phase
   - » Check Certificate with root CA certificate
   - » Agree on a set of cryptographic protocols: AES 123, DSA, RSA, etc
   - » Perform key exchange to obtain session keys and other values like IVs
   - » Verify authenticity using the public key
2. Transport/Record phase

when sending encrypt packets with the agreed method
always include a MAC for integrity!

Common Protocol Issues

– Cipertexts that aren't secured with MAC

This means the message can be altered in transport

– Messages without a timestamp

This means we can copy and resend this message
from another PC!!

– Protocols without public key -> without authenticity

These are vulnerable to man in the middle attacks

– Reuse of Nonce

might lead to a broken cipher...

Dos

- Use cryptographically strong random numbers. In python this is os.random and os.urandom, not math.random (for example to generate RSA keys, DH private keys)
- Use "recipe" layers where possible. They take the algorithm and protocol decisions out of your hands
- Use ephemeral session keys for communication, and long-term public-keys for authentication
- Nonces are numbers only used once
- IVs must be unpredictable

Don'ts!

- Implementing your own algorithms is a bad idea!
- Never use hard-coded keys, convenience comes at a cost
- Don't use ECB mode. Prefer CBC, but better yet CTR or GCM
- Don't use small public key sizes. At least 2048 bits! Elliptic curve is preferable: P-256 or X25519

Network Cryptography Tips

- For general end-to-end communication, consider using TLS
- TLS supports client and server authentication using two certificates –useful for many setups
- Restrict what cipher suites are allowed to avoid protocol downgrades
- If you don't use TLS, always use AEAD or another authenticated encryption mechanism

Storage and DB Tips

- Use password derivation functions not hashing functions for password storage. E.g. PBKDF2
- Encrypt data where possible using keys that you change fairly often
- Encrypt the keys, adding a layer of indirection for an attacker. (Key encryption key)

Transport Layer Security (TLS)

– Confidentiality

– Integrity

– Authentication

– Fragmentation

– Compression

– Primary protocol for HTTPS!!

However, it can be used by everywhere!

– Other:

Versions: SSL1.0, SSL2.0, SSL3.0, TLS1.0, TLS1.1, TLS1.2, TLS1.3
DO NOT USE SSL! In fact USE TLS 1.3!!
» currently at version 1.3
Previously called Secure Socket Layer (SSL)

TLS Connection

» peer to peer

» transient, they are not stored

» each connection is associated with exactly 1 session

– Server and client random number

– Server and client MAC private key

– Server and client encryption private key

– Initialization Vectors

For block ciphers IV are maintained for each key.

– Sequence numbers

Same as with TCP
When change cipher spec is used, the sequence number is reset to 0
Sequence numbers may not exceed $2^{64} - 1$

TLS Session

» Defines the cryptographic parameters used

» Sessions are used to avoid expensive negotiation of security parameters on every new connection

– Session Identifier

an arbitrary bite sequence chosen by the server to identify
an active or resumable session state
Can also be a previous session to resume!!!

– Peer certificate

certificate of the peer, may be null

– Compression method

The algorithm used to compress data prior to encryption

– Cipher spec

Specifies: encryption algorithm, hash algorithm, hash size, Signing algorithm

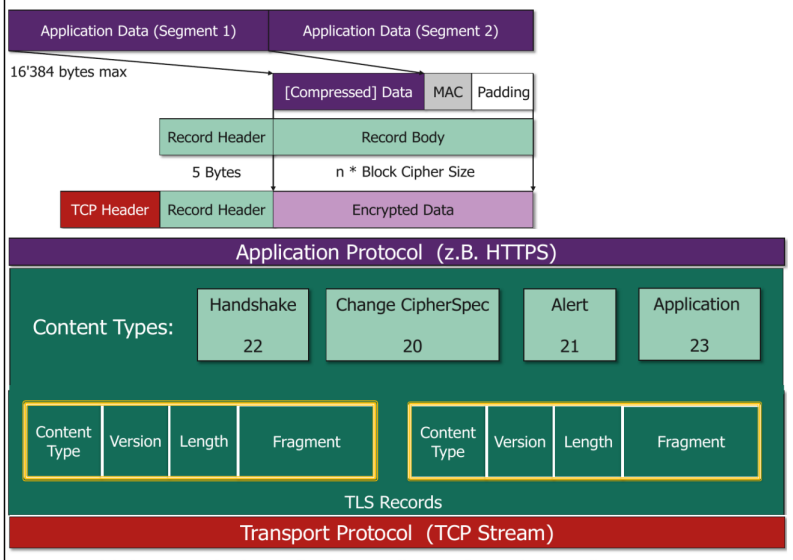
– Master secret

shared key that is used for encryption -> Diffie-Hellman

– Is resumable -> can be stored, unlike connections

a flag that indicates that this session can be resumed

TLS and TCP

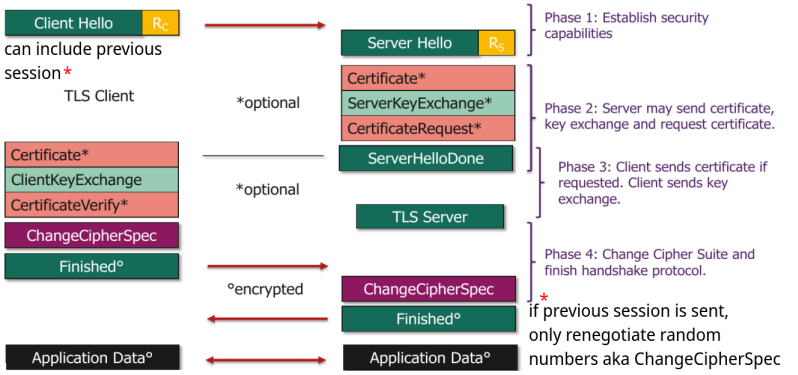


- Handshake Protocol: ClientHello, ServerHello, Certificate, ServerHelloDone
- Change Cipher Spec Protocol
- Alert Protocol: Warning, Fatal (Session is closed immediately)
- Application Protocol: Transmission of (encrypted) payload

TLS handshake

The handshake allows server and client to negotiate
all necessary algorithms and specifications

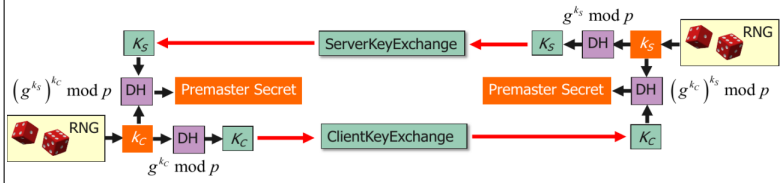
Note that this example is for older versions of TLS, NOT TLS3.0!



Rs and Rc are the random numbers

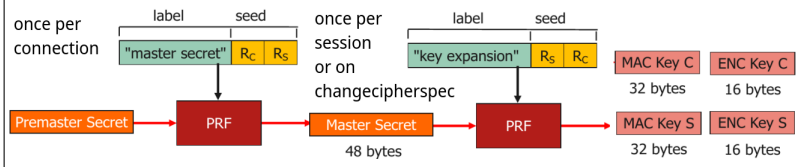
Ephemeral Mode in TLS

We have already learned that we calculate a new key for every session
This now applies to TLS, which conveniently works with sessions as well
This means we now calculate a new key for every session we have with the server



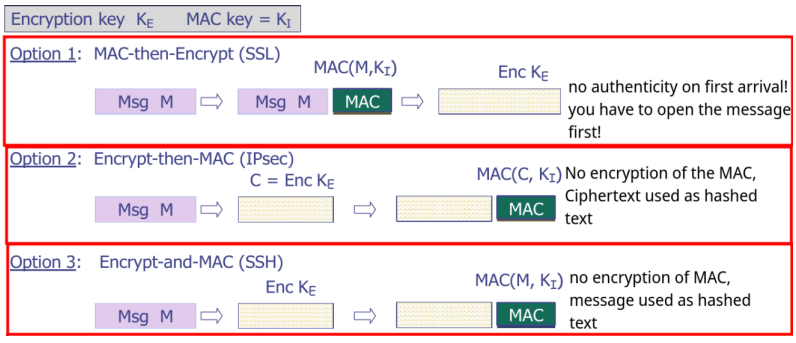
Master secret to Signing and encryption key

The master secret is used to calculate all the other keys,
this makes it possible that the server and client only need
to share keys once, after that they can simply
send a new random number for a new set of keys!! Super simple!!



Different Combinations in TLS

With Diffie-Hellman you can encrypt only encrypt/decrypt
With RSA/DSA you can sign or encrypt/decrypt
The most usual combination is to use RSA/DSA and Diffie-Hellman!
However, what do you do first? Encrypt or sign?
You can choose this depending on your needs.
If you want 100% anonymity then use encrypt first
If you want authenticity then use signing first.



TLS allowed Combinations:

TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256

Key Exchange / Authentication
Encryption / Authenticated Encryption
Message Authentication Code / Pseudo Random Function

TLS alerts:

Consists of 2 bytes: severity and code

- Severity
  - There are 2 severity levels, warning and fatal
  - on fatal the connection terminates immediately
  - Warning simply shows the alert code
- Code
  - Specific alert code.

Heartbeat Protocol

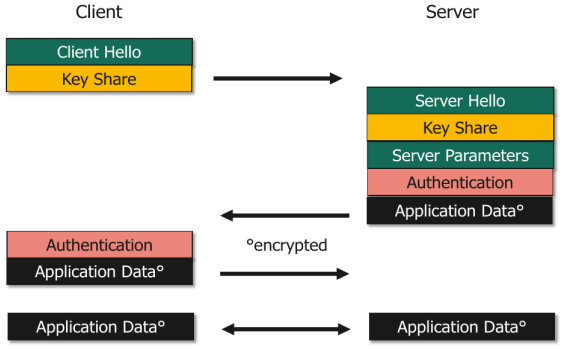
- Keep-alive Protocol
  - assures the server that the client is still alive
- Runs on top of TLS
  - negotiated on handshake

» Heartbleed (Heartbeat exploit) «

This is done by requesting a bigger size package than the heartbeat actually is
This can trick the server into sending data that it wasn't supposed to send.
If done often enough and with lots of luck, the server might send data,
that is critical, such as private keys and password hash-tables!

TLS 1.3

- Removal of broken features
  - Removed broken ciphers like MD5, SHA-1, Kerberos, etc
  - Removed compression and renegotiation
  - Remove static RSA/DH
- Allows 1-RTT and 0-RTT
- Improved security by using modern techniques
- Privacy encrypt more of the protocol
  - Almost all handshake messages
- Continuity » backwards compatibility

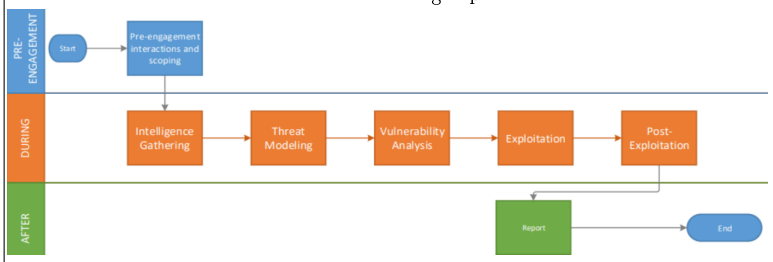


Key Exchange: DHE, ECDHE (p256, p384, p521, x25519, x448)
Authentication: RSA, ECDSA (p256, p384, p521), EdDSA (Ed25519, Ed448)
Cipher Suites: TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_SHA256
TLS_AES_128_CCM_SHA256 (IoT)
TLS_AES_128_CCM_8_SHA256 (IoT)

Authenticated Encryption (AEAD)
Hashed Key Derivation Function (HKDF)

Penetration Testing

Penetration testing is done to find and eliminate vulnerabilities
Usually companies hire a specialist to do this work
For legal reasons it is very important that a rigorous contract is signed
The reason for this, is that attacking a system is illegal in many countries
So you need explicit permission from the company to do so.
The contracts are usually called Statements of Work
They include: Activities to be performed, pen testing timeline
Scope, and Location of work
Usually you also have to sign a Non-Disclosure agreement
This makes sure the vulnerabilities do not get public!



Zero-Day attacks

This is an exploitation of a vulnerability that has been present since
the creation of the program/service. However, it has not been found yet.
The time frame of hackers figuring out the vulnerability and the developers
patching said vulnerability, is called the window of vulnerability
» Zero-Day is a big problem, because system administrators often wait
with applying updates to systems, which leaves a bigger window for attackers

Malicious Code

Code that is written to exploit vulnerabilities on networks, OSes, etc

- Viruses,Worms,Trojans
  - These are the most typical forms of malicious code.
  - Most of them are only received through misuse of a computer
  - Viruses usually have 2 functions, Propagation and Destruction

Sources of Malicious Code

- Script Kiddie
  - downloads crap and spams it into the internet
- Drive-by Download
  - This exploits a vulnerability where the browser downloads
  - something in the background without your knowledge
- Advanced Persistent Threat (APT)
  - These are often state sponsored actors with large amount of resources.
  - They often use Zero-Day attacks against very specific/high value targets
  - An example is the RUAG Stuxnet attack.

Virus Propagation Techniques

- Master Boot Record Infection
  - Very nasty, although the MBR is not big enough to host the virus
  - It is perfect to launch it without a users knowledge
  - It simply points to a location somewhere on a disk,
  - where the virus is stored
- File Infection
  - These viruses change existing files to include their own code.
  - Often executables are used, although there are many other variants.
  - Most standard injectors are easily detected with antivirus software
- Companion Virus
  - A Virus that has a similar name to legitimate system services.
- Service Injection
  - Instead of regular file injection, these viruses inject themselves
  - into critical system services in order to escape detection from
  - antivirus software. Can be nasty to remove
- Macro Infection
  - This is an infection inside an application that has a scripting language
  - For example Excel has the visual basic crap language inside it.
  - A virus can be written in that language and be configured to
  - launch at startup. Which means click on file = pwned
  - Although this specific variant was fixed, there are
  - always new ways in which this happens again and again

Malware Variants

Note: Malware can have multiple variants at once!

- Multipartite Viruses
  - Viruses that use more than 1 propagation technique
- Stealth Viruses
  - These are viruses that use tricks to fool the OS and
  - antimalware software that everything is working fine.
  - An example is an MBR infection virus that returns a
  - clean MBR if the antimalware software checks the MBR
  - but on boot it selects the infected MBR
- Polymorphic Viruses
  - These viruses change their own code on each infection
  - This is similar to how an actual virus gets mutations
  - The effect is that the signature of the virus changes!!
  - This makes it hard for antimalware software to detect it!
- Encrypted Viruses
  - Uses encryption to hide itself
  - On each system, a new key is used in order to change appearance
  - The Virus decryption routine handles encryption and decryption
- Logic Bombs
  - This malware stays dormant until a certain time,
  - or certain prerequisites are active.
- Trojan Horse
  - Software that appears kind, but carries a malicious payload behind the scenes
  - It is often used in combination with a keylogger to get critical information
- Key stroke logger
  - Logs every single keypress the victim does on a system
  - Can also be in form of hardware!
- Ransomware
  - Malicious Code that encrypts the entire drive of a victim and
  - prompts them to pay a ransom for their files
  - If the user doesn't pay in time, the files are deleted.
  - Examples: Petya, wannacry, cryptlocker
- Worms
  - Malware that propagates without any human interaction/intervention
  - Often used in combination with other types.
- Adware
  - Often targeted at browsers. Launches and shows random adds and websites
  - Were often bundled with legitimate software in the 2010s.
- Spyware
  - Reports (all) information on your system to an attacker
- CryptoMiner
  - Software that mines cryptocurrency for an attacker in the background

Antivirus / Antimalware

Most antimalware software uses multiple methods!

- bluSignature based
  - Checks for specific signatures (hashes) from a regularly updated database
- Heuristics based
  - Checks for specific malicious behavior
- Data integrity based
  - Checks for suspicious file changes on the system
  - Usually this is a change to an executable without updating it.

Application Attacks

- Buffer Overflow Attack
  - Send more data than the server anticipated.
  - Unless the server is modfied to drop the rest of the data,
  - it is written as a block to memory. In that additional data,
  - there might be instructions, commands that the attacker wants to run.
  - input field : 5 char -> input = "hellonetcat 192.168.1.1 ..." (reverse shell)
- Time of Check to Time of Use (TOCTTOU)
  - If an admin revokes a users permission while said user is online,
  - then the change won't happen until said user loggs off.
  - This user can now simply choose to not log off and keep all permissions.
  - A simple way to fix this is, is to force log off said user.

- Back Doors
  an undocumented way of accessing a system
  This can either be done via malicious software,
  or intended backdoors by developers.
  The developer backdoors are often only used in testing, and removed later
  The removing part can of course simply be forgotten :)
- Escalation of Privilege
  Use exploits to elevate the priviliges on a victims system
  often done with a rootkit, but does not necessarily have to be root!
  For example, an elevation from user to moderator would also suffice.
- Rootkit
  Uses various exploits to gain root access on a system.

Web Application Attacks

- Cross Site Scripting
  A browser opens scripts with the <script>script-name</script> notation
  Normally this is used to get input from users and to spawn other things
  However, these scripts can also be integrated into a link directly
  In this case an attacker might send you a link to a legitimate website,
  however, inside that link is also a script that gets loaded,
  is not from that website
  This means that the hacker could get your login information,
  if you enter that on this website
- Cross Site Forgery (XSRF or CSRF)
  This sends a command to another website that is currently open.
  For example, if you are currently logged in into your bank account,
  and you have that website open, then clicking on such a malicious link
  might steal funds from you without you noticing!!
- SQL Injection Attacks
  form username: password: in username you write:
  dashie';DELETE * from acc;
  you end the first command and then enter another, both commands:
  SELECT * FROM acc WHERE name = 'dashie'; DELETE * FROM acc;
  bye bye table :). Luckily this is easy to combat
  Use prepared statements, limit privileges and perform input validation

Network Attacks

- Denial of Service (DoS)
  Simple resource consumption attack that tries to use all bandwidth.
  This makes the server unable to provide the service to other users.
- SYN Flooding
  Spam a server with SYN floods. Aka pretend to open 999999 TCP connections
  The server responds to these SYN requests, but you never answer,
  instead you send new SYN requests
  Without protection against this, the server can't open new connections!
- Service Request Floods
  Similar to the SYN flood with the difference being actual open connections
  This means you actually respond and open the connection, only to then
  use this connection to spam application requests.
- Application level DoS
  Exploits a vulnerability in a software to spam a server with bad packages.
  These bad packages might overload the server, preventing access.
- Permanent DoS
  Crippling the hardware itself so the service can't be accessed.
  A good example if installing faulty firmware (Phlashing attack)
- Botnets
  This is a network of bots(systems) that have been compromised.
  It is then used to perform a combined attack on a victim,
  or simply to add more systems to the botnet.
- Distributed DoS
  A combined targeted DoS against a service
  Much harder to combat than single DoS, often done via Botnets
- Man-in-the-Middle Attack
  Forcing traffic to your own system instead of router etc.
  This allows package manipulation, eavesdropping etc.
- Man-in-the-Browser Attack
  Trojan installed via plugin, used to steal credentials,
  inject scripts and hijack authentication sessions
- Eavesdropping
  Monitoring the traffic of a system to the net.
  Hard to detect as it has no symptoms.
- Impersonation/Masquerading
  Pretending to be someone else in order to gain access to a system
  requires authentication credentials to be stolen!
- Replay Attacks
  Gaining access to a session by sending copies of older messages.
  Possible after eavesdropping a session.
- Modification Attacks
  Capture and then modify packages.
  It is essentially a man-in-the-middle attack.

Session IDs and attacks

- Session IDs
  Session IDs are a quick way of knowing who the server is dealing with
  It allows for storage of tokens, cookies etc.
  And it reduces the amount of times we have to enter passwords
- Hijacking with Session ID Predicting
  The attacker tries to bruteforce guess the Session ID.
  Attacker sends thousands of guesses based on previous sessions.
- Session Hijacking with Cross-Site Scripting
  Get the session ID with a cross-site script
- Session Hijacking with Cross-Site Request Forgery
  Get the session with a malicious command
- TCP/IP Hijacking
  During a TCP handshake, respond to quicker to the server than the victim.
  or force the victim to reset the handshake you start the session instead.

Ethical Hacking and regular Hacking

- Ethical Hacking
  System auditing, reporting and testing
  Vulnerability reporting
- Hacking
  Security control compromise
  Produce behaviors in a software outside of original intent
  Exploitation of vulnerabilities

Types of Hax0rz11!

- Black Hat
  malicious/destructive hacker
- Grey Hat
  possessing black hat skills, but focus on both offense and defense
- White Hat
  Defense focused hacker
- Cyber Terrorist
  Advancing Ideology with cyber attacks

Email Security S/MIME

Multipurpose Internet Mail Extension MIME

- Application Layer Protocol

```
From: trinity@matrix.org
To: neo@matrix.org
MIME-Version: 1.0
Content-Type: multipart/mixed;
  boundary=boundary1
[
--boundary1
Content-Type: text/plain; charset=us-ascii
[
Dear Neo, please study the attached Word document.
--boundary1
Content-Type: application/msword; name="Matrix.doc"
Content-Transfer-Encoding: base64
[
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfH
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbTrfv=
--boundary1--
```

– S/MIME signing format 1

```
Content-Type: multipart/signed;
  protocol="application/pkcs7-signature";
  micalg=sha1; boundary=boundary1

--boundary1
Content-Type: text/plain

This is a clear-signed message.

--boundary1
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfH
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbTrfv=
```

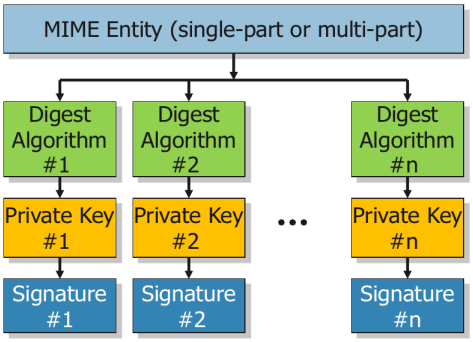
MIME entity to be signed

OR

```
Content-Type: multipart/mixed; boundary=boundary2

   ... multipart message with various MIME-types ...
```

– S/MIME supports Multiple Signatures



– S/MIME Signing format 2

```
Content-Type: application/pkcs7-mime;
  smime-type=signed-data;
  name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfH
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbTrfv=
```

MIME content carried within PKCS#7 Signed Data Object

\+ This format is not affected by encoding enforced by mail providers

– In order to read this format, the mail client must support S/MIME

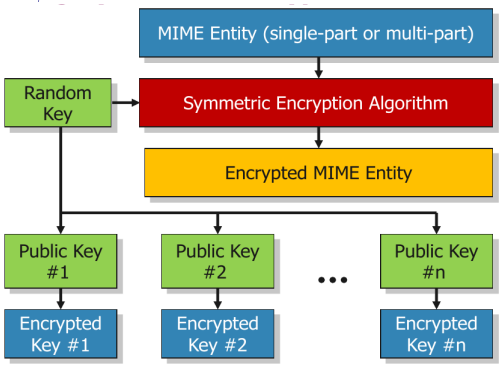
– S/MIME Encrypted format:

```
Content-Type: application/pkcs7-mime;
  smime-type=enveloped-data;
  name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfH
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbTrfv=
```

ASN.1 structure for the **EnvelopedData** content type — enveloped = encrypted

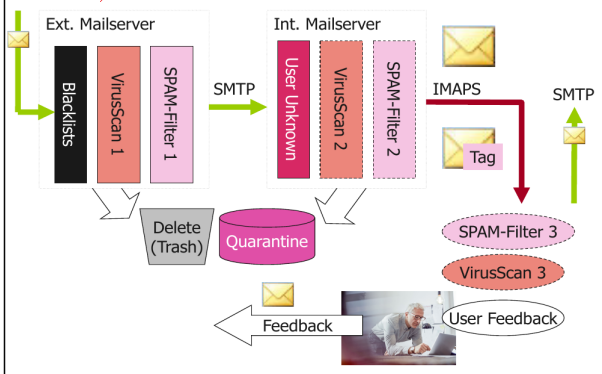
– S/MIME Multi Encryption



Just like before, if you sign before you encrypt you have authenticity, if you encrypt before you sign you have anonymity.

For more info, check the encryption and signing part of this doc.

SPAM :)

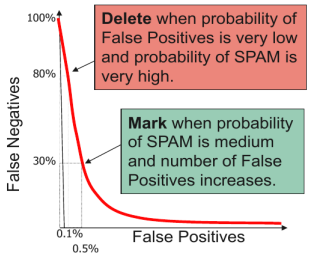


Spam Assassin as an Example

- Framework for combining SPAM detection techniques
- SpamAssassin examines many parts of the email
  - Fine details of the headers
  - Regexps in the body text
  - Tests evaluate things like "HTML tag percentage"
- Extract features from the email
  - Look up how many times we have already seen a certain feature in emails that have been declared as SPAM.
  - Create probability for that feature
  - Combine all the probabilities for all the features into an overall probability
- A self-learning Bayes network is continually trained on the incoming emails
- Uses SPAM Signatures

– False Negatives vs False Positives:

- Different probability thresholds can be used to trade off False Positive rate (Accuracy) versus False Negative rate (Completeness)
- Delete only when probability of being SPAM is high
  - Lot of missed SPAM
  - Few false positives
- Mark when probability of being SPAM is medium
  - Few false negatives
  - Many false positives



First doesn't cover all spam, but guarantees that no legitimate mail is missed

Second blocks nearly all spam, but also blocks some legitimate mail!

– Sender Policy Framework (SPF)

TXT records in DNS

Mail client does DNS query for the mail sender.

- SPF response may be
  - No published SPF records
  - Pass
  - Fail
  - Sender cannot be confirmed as legitimate or illegitimate
  - Lookup error
  - The check is neutral or incomplete, which should be treated as if there was no SPF record
- Example SPF DNS record
  - strongsec.net TXT "v=spf1 a mx a:lists.strongswan.org -all"

DKIM Domain Keys Identified Mail

```
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
    d=gmail.com; s=20161025;
    h=mime-version:from:date:message-id:subject:to;
    bh=p62SWWykr87qyDGT/9jfMLkh15E0x/U4F82UKrGMcSw=;
    b=XmDyWfhv8bHaHg2hJw5+wqPNLDNYgOGNgLP8rx2L65qQj+5FfuByMoJGPM2v239RBvr
    Kqf7IcPYKPO/5oyVYEtPVbbiAyOPK4B3R0RiA1LqsQD7IVU5xEjn3B8QIHOys9TQQNPQM
    YZfinh1Viv++FqeF65CdMfOU2Qak1d+KXIfldZ8mz7xXbTwRPmIp+zv2t22ao9krnZ3JH
    sLAZud15zSGwQ+Rd+OXYwtBkZ3BoChGSPFBVJ0LjhJULOYHyznI7iIi51MCBMR9TQPkRJ
    sWCpdukAK+Vo7Rv0NI1zWro5NundKYj1KqqkZVwiaLhflj7Ebvu/NVaHr1cSL9pQAasA==

20161025._domainkey.gmail.com  TXT "k=rsa;
    p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAviPGBk4ZB64UfSqWyAicdR7l
    odhytae+EYRQVtKDhM+lmXjEqRtP/pDT3sBhazkmA48n2k5NJUyMEoO8nc2r6sUA+/Dom5
    jRBZp6qDKJOwjJ5R/OpHamlRG+YRJQqRtqEgSiJWG7h7efGYWmh4URhFM9k9+rmG/CwCgw
    x7Et+c8OMlngaL104/bPmfpjdEyLWyNimk761CX6KymzYiRDNzlMOJOJ7OzFaS4PFbVLn0
    m5mf0HVNtBpPwWuCNvaFVflUYxEyblbB6h/oWOPGbzoSgtRA47SHV53SwZjIsVpbq4LxUW
    9IxAEwYzGcSgZ4n5Q8X8TndowsDUzoccPFGhdwIDAQAB"
```

Sending mail server signs mail hash

Receiving mail server checks signature by fetching public key via DNS

Other Methods

– Blacklists

Filter based on a list of bad actors

\+ Very fast with no false positives

– List must be updated constantly, new senders not blocked

– Greylisting

Check Sender IP, sender mail address and recipient address

Test this against database. If it is unknown return failure to sender.

\+ blocks ALL spam

– also blocks legitimate mails that are unknown

– senders with multiple IP can't send mail

– Check can slow down mails to minutes or hours

Authentication

– A1 Plain Password

Regular Passwords, easy to sniff if they aren't encrypted.

– A2 One time Passwords

Can't be sniffed and used, but Man in the middle attacks are possible

– A3 Challange Response Protocol

Server sends a random value as 'challenge'.

The user then hashes his password and turns it into a MAC.

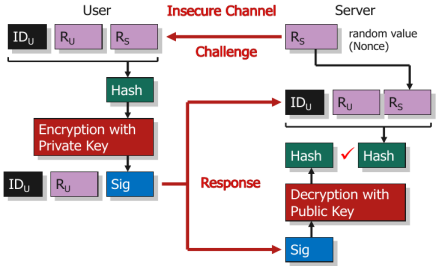
The user then sends the MAC alongside his own Random number and his ID.

The server calculates the MAC as well and compares it.

The server has the password saved as well, aka it just hashes that

This also means that a brute force is possible if the password table is stolen

For better protection you send a signed MAC instead, user authenticity!



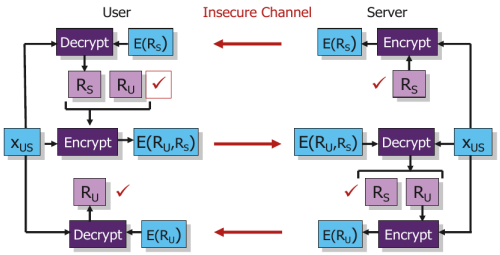
– A4 Anonymous Key Exchange

Essentially encrypted password

– A5 Zero Knowledge Passwords

Challenge Responses with encrypted random numbers.

Server also sends the MAC back to user to double check!



– A6 Certificate-based Server Authentication

Server authenticates itself, then encrypted password used.

– A7 Mutual Public Key Authentication

You authenticate yourself and the server authenticates itself

Recommended if you want a certain machine to have access.

Often combined with encrypted passwords.

Attack	A1	A2	A3	A4	A5	A6	A7
Passive Password Sniffing	x						
Offline Brute Force Password Attack	x		x	x			
Active Man-in-the-Middle Attack	x	x	x	x			
Identity Theft on Server	x	x	x	x	x	x	
CA Compromise						x	x

Kerberos

– Realm == Domain

– Key Distribution Center (KDC)

The Server that handles key distribution and Kerberos tickets

– Kerberos Tickets

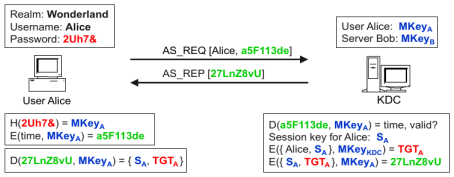
Mediates the communication between principles.

– Principles (Users)

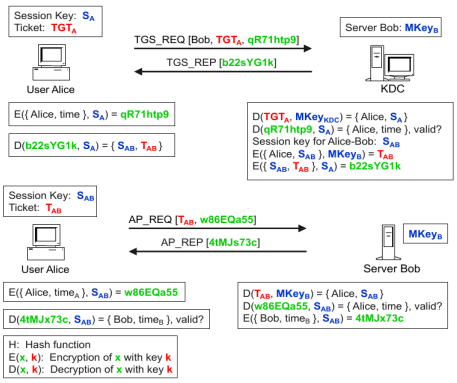
Every user in the network is called a principle

– Shared key = Principles master key

Key that is assigned to a principle. Connects to the KDC



Shared secrets like Diffie-Hellman and time-stamped.



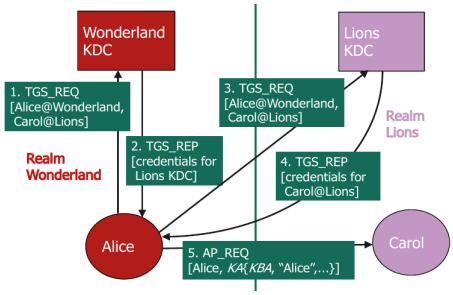
TGT: user authenticated for a certain amount of time.
If the user is timed out, user has to re-authenticate with KDC
AS = Authentication Response
TGS = Ticket Granting Service
AP = Access?

– KDC Slaves

In order to save the KDC from overload, we create some read-only KDC slave server, that can create tickets and authenticate users

The only thing they can't do is create new passwords for users etc.

– Inter-Realm Authentication



Federation

The linking of 2 domains alongside their login systems.

Example: kerberos inter-realm authentication.

Formal Definition: a set of oranizations agreeing on a common set of rules and standards.

– Shibboleth

» based on SAML, opensource, EDU-SWITCH, most used protocol

» used for WLAN, E-Learning, Libraries, etc

– W3C - XML Signature/Encryption

xmlns:ds = signature, xmlns:enc = encryption

– Secure Assertion Markup Language (SAML)

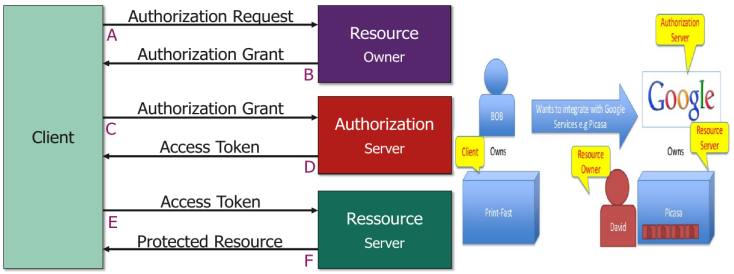
Parts: Assertions and Protocols, Bindings, Profiles, Metadata, Authentication-Context, Conformance Requirements, Security/Privacy, Glossary

used with xml : xmlns:saml / xmlns:samlp

– OAuth 2 Authorization Framework

» Used to connect services to accounts, ex. use Picasa resources from google

» based on https, security handled by TLS, used by big companies



– Open-ID Connect Authentication Layer

» on top of O-Auth 2.0, solves multiple IDs for multiple services

» used by many big companies, implementation up to service

» SwissID also uses this.

- OAuth requests that you specify a "scope": List of access methods that the app needs permission to use
- To enable user identification: Specify "openid" as a requested scope
- Send request to server (identity provider): Server requests user ID and handles authentication
- Get back an access token: If authentication is successful, the token contains:
  - user ID
  - approved scopes
  - expiration
  - etc

  } same as with OAuth requests for authorization

Technical Vulnerability Management

The idetification, classification, remedation and mitigation of vulnerabilities

- Plan
  Identify Assets that could be exposed to risk
  Establish who or what will conduct the vulnerability tests
  Establish how the tests will flow into software development
- Discover
  Monitor sources about known vulnerabilities that might apply to your system
- Scan
  » run automated tests based on your previous results
  » hire professionals to do penetration testing
  » Compare these results back-to-back
  Note, certain tests might disrupt the system!
  And, there might be false positives! Especially with the automated tests
- Log & Report
  Log results based on severity...
- Remediate
  » Ideally, every discovered vulnerability should be patched immediately
  » However, performing a proper cost analysis as explained in the first 2 pages is often preferred as not all vulnerabilities are worth fixing.

Security evens and Security Incidents

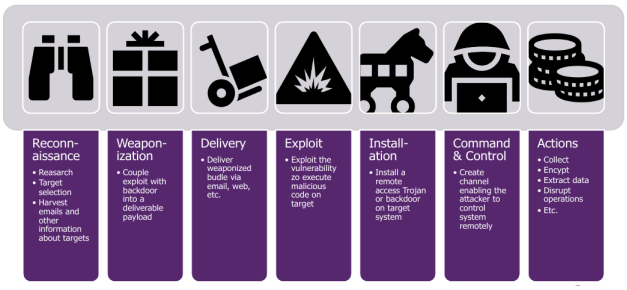
- Security Event
  A security event is just a simple logging mechanism for noteworthy changes: password changes, certificate renewal, new certificates, new root CAs etc
  These can be ignored if they were intended.
  They need to be configured, as you don't want your logs spammed with trash
- Security Incident
  a Security Incident is a detected breach!! Take action!!
- Scanning for Security Incidents
  » Pattern matching: find a collection of events with similar cause
  » Scan detection: attacks often begin with port scans etc!
  » Threshold: if x events with y port scans, then ALERT
  » Event correlation: assign the attack to a known attack variant

PLAN-ASSESS-SIMPLIFY-DEPLOY

Best practises for security event management

- PLAN
  » What systems should be monitored?
  » What events are important, which can be discarded?
  » Where should we store the logs, and for how long do we store it?
  » How will the performance of this system be monitored?
- ASSESS
  Understand your priorities
- SIMPLIFY
  Order the priorities and simplify them
- DEPLOY
  Turn your priorities into action

Cyber Attack Kill Chain



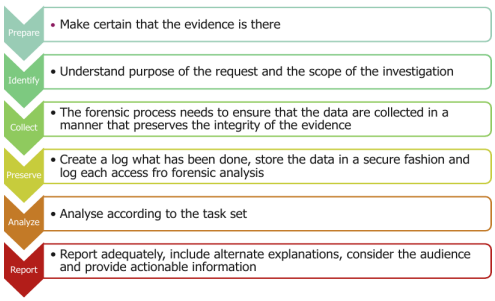
How to deal with each step:

- Reconnaissance
  » use firewalls, whitelists, Network segmentation
  » simple port scans can also be deterred by using high random ports
- Weaponization
  » patching a vulnerability before it is exploited....
- Delivery
  » Firewalls, antimalware software, good policies
  Intrusion Prevention Software (IPS), Web Application Firewall (WAF)
- Exploit
  » Host-based Intrusion Detection System (HIDS), Backups, frequent patching
- Installation
  » antimalware software, HIDS, proper security events -> privilege escalation
- Command & Control
  Firewall, Network-based Intrusion Detection System (NIDS)
- Actions
  you are fucked, restore the backup.

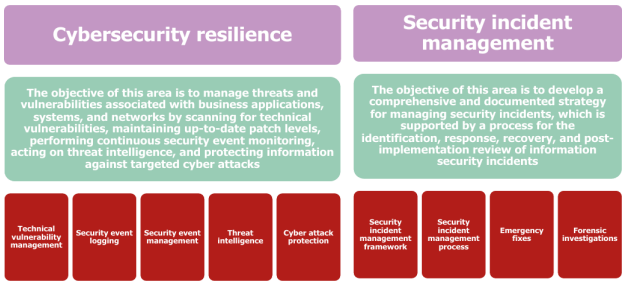
Incident Management Process

- PREPARE
  » WHO-WHEN-WHAT
  » develop Internal and External reporting procedures
  » Guidelines for interacting with external parties (PLAYBOOKS)
  » Define service by Incident Response Team (IRT)
  » Establish and maintain accurate notification mechanism
  » Establish which incidents take precedence
- ANALYSE
  Magnitude? Severity? Urgency?
- CONTAINMENT
  Depending on the previous analysis, take action and contain the damage or the attacker.
- RECOVERY
  Restore the system to regular state and patch it if necessary

Digital Forensics



Threat Incident management best practises



OWASP TOP 10

Exploit: How easy is it to exploit it?
Impact: If exploited, how harsh are the consequences?
The top 10 vulnerabilities, redefined each year

- A1: Broken Access Control
  Attackers have access to accounts they shouldn't have
  Mitigations: Proper access control..., input validation occurred: 318k, Exploit: 6.9, Impact: 5.9
- A2: Cryptographic Failures
  Missing or broken Ciphers like SSL, MD5, etc
  Problem right now: TLS 1.2 with ineffective configuration
  Mitigation: Use proper crypto tools like TLS 1.3, verify your hardware
  Make sure encryption is used correctly
  Occurred: 234k, Exploit: 7.3, Impact: 6.8

– A3: Injection

Why is this still happening?

Just do input validation or use prepared statements ffs!

Output sanitation is also useful, simply encode the dangerous characters!

More info in the first 2 pages...

Occurred: 274k, Exploit: 7.3, Impact: 7.2

– A4: Insecure Design

Use Threat models, better frameworks, lots of testing

most costly problem to fix later, so do it now!

Occurred: 262k, Exploit: 6.5, Impact: 6.8

– A5: Security Misconfiguration

Outdated and vulnerable software in production, default admin accs

weak passwords, bad ports, leaking info with error messages

Mitigation: Use software to detect misconfiguration and act accordingly.

Harden Systems, deploy in testing first, update systems

Follow Recommendations!

Occurred: 208k, Exploit: 8.1, Impact: 6.6

– A6: Vulnerable/Outdated Components

Nothing to say other than don't use old versions for critical systems..

Occurred: 132k, Exploit: 7.4, Impact: 6.5

– A7: Identification/Authentication Failures

These are login breaches.

Mitigation: use better hash, use one time passwords

Make sure tokens etc are correctly configured

Make sure no part of the authentication is in plaint text

Perhaps internal leak?

Occurred: 262k, Exploit: 6.5, Impact: 6.8

– A8: Software and Data Integrity Design

Corruption or malicious change of files

Mitigation: Raid 0, backups, validity checks, no plain text packages

Occurred: 47.9k, Exploit: 6.9, Impact: 7.9

– A9: Security Logging and Monitoring Failures

Useless errors that only say: oops error sorry

This could be an attacker but who knows, the error won't tell you

In other words, the attacker is protected, because you aren't informed!

Mitigation: Offer more information to the user, at the very least on request.

Occurred: 53.6k, Exploit: 6.9, Impact: 5.0

– A10: Server Side Request Forgery

Server loads URL without checking, making it possible to load malicious stuff

Including commands to execute on website, like banks, hence forgery

Mitigations: deny by default firewall, Sanitize URLs, whitelist for URLs

disable https redirections, DON'T use Blacklists!

Occurred: 9.5k, Exploit: 8.2, Impact: 6.7