1 Week 01 - Cybersecurity Concepts

1.1 Asset

Asset: anything within an enviornment that should be protected loss or disclosure could result in:

- security compromise
- loss of productivity
- reduction in profits

assets of an organization:

- information
- systems
- devices
- facilities
- personnel
- intellectual property

1.1.1 How to proectect intellectual property

copyright

- law guarantees that the creators authorship is protected against unauthorized duplication
- source code can be under copyright
- but not the idea of program; everyone could use another language to do the same thing

trademarks

- protects for words, slogans and logos to identify a company or its products / services
- it's not required to official register; just put the slogan^{TM} symbol
- for official recognition register is required
  - is the registration successfully you can use the (R) symbol

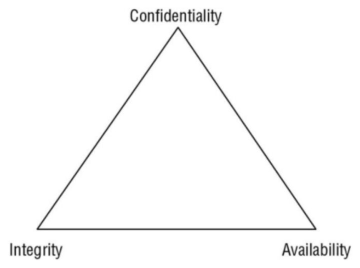
patents

- protects the intellectual property right of inventors for 20 years
- after 20 years the invention is in public domain
- invention has to be new, useful, not be obvious
- does not work well for computer software

trade secrets

- if your success is based on a idea of you, you could protect id with copyright or patetn
  - but maybe you don't want to get this information out? => trade secret
- it's one of the best technique to protect computer software (Microsoft, Cisco, Apple, ...)

1.2 CIA triad



confidentiality

- prevent unauthorized access to data
- example: encryption, access controls

integrity

- protecting the reliability and correctness of data
- example: intrusion detection systems (IDS), hash verification

availability:

- authorized subjects are granted timely and uninterrupted access to objects
- example: redundancy, maintain reliable backups, prevent data loss or destruction

nonrepudiation: ensure that the subject is not able to

- deny having performed an action
- deny that the event occurred

accountability

- being responsible or obligated for actions and results
- nonrepudiation is an essential part of accountability

1.3 Data classification

- is used to determine how much effort, money and resources are allocated to protect the data

data states:

- **data at rest**: data on media such as HDDs, SSDs, SANs, ...
- **data in transit**: data transmitted over a network
- **data in use**: data processed by an application
  - data can not be used in encrypted state

defining sensitive data:

- **Personally Identifiable Information (PII)**
- **Protected Health Information (PHI)**
- **Proprietary Data**
  - any data that helps an organization to maintain a competitive edge

Government / military classification:

1. Top secret
2. Secret
3. Confidential
4. Sensitive but unclassified
5. Unclassified

Commercial business / private sector classification:

1. Confidential / Private
2. Sensitive
3. Public

destroying sensitive data:

- **erasing**
  - normally deletes just the directory / catalog link to the data
- **clearing / overwriting**
  - preparing for reuse
  - write 1 character, write complement, write random bits
- **purging**
  - more intense form of clearing to prepare media for reuse in less secure environments
  - repeats the clearing process multiple times
- **degaussing**
  - create a strong magnetic field that erases data on some media
  - does not affect optical medias and SSDs
- **destruction**
  - destroy the media
  - crushing, shredding, ...

tracing or hiding sensitive data:

- **Steganography**
  - practice of embedding a message within a file
- **Watermarking**
  - embedding an image / patter in paper that isn't readily perceivable
  - digital watermarks is a secretly embedded marker in a digital file

1.4 Threat

threat: any potential danger to an asset (intentional or accidental)
threat actor / agent

- script kiddies
- organized crime groups
- state sponsors / governments
- hacktivists
- terrorist groups

threat intelligence: knowledge about an existing threat to assets)
threat event:

- accidental and intentional exploitation's of vulnerabilities
- can be natural or man made
  - example: fire, earthquake, flood, system failure, human error, ...

STRIDE thread model

- **S**poofing: gain access to a system through the use of a falsified identiy
- **T**ampering: any action resulting in unauthorized changer or manipulation of data
- **R**epudiation: the ability of a user / attacker to deny having performed an action
- **I**nformation disclosure: the revelation / distribution of private / confidential or controlled information to unauthorized entites
- **D**enial of service (DoS): attack to prevent authorized use of a resource
- **E**levation of privilege: a limited user account is transformed into an account with greater privileges / power / access

1.5 Vulnerability

Vulnerability: weakness in an asset

- Common Vulnerabilities and Exposures (*CVE*)

Exploit: software / sequence of commands the take advantage of a vulnerability

1.6 Risk management

Risk management:

- develop information security strategies
- reduce risk (to an acceptable level)
- identifying factors, evaluating factors, implement cost-effective solutions for mitigating / risk reduction

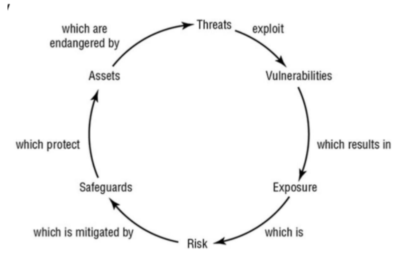
Countermeasure: any action / product that reduces risk through elimination / lessening of a thread / vulnerability
Risk analysis:

- process by which the goals of risk management arch achieved
- evaluation of all assets
- examining for risk
- evaluating each threat
- assessing the cost of countermeasures

Asset Valuation: dollar value assigned to an asset based on

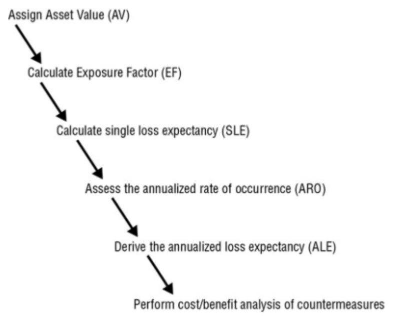
- cost
- non monetary expenses

Exposure: possibility for an asset loss because of a threat
Risk: possibility that something could happen to damage, destroy or disclose data / resources
Attack: exploitation of a vulnerability by a threat agent
Breach: occurrence of a security mechanism being bypassed



1.6.1 Quantitative risk analysis

1. **Exposure factor (EF)**: percentage of the overall asset value loss
2. **Single loss expectancy (SLE)**: cost associated with a single realized risk
   - SLE = asset value (AV) * exposure factor (EF)
3. **Annualized rate of occurrence (ARO)**: the expected frequency with which threat / risk will occur within a single year
4. **Annualized loss expectancy (ALE)**: the possible yearly cost against a specific asset
   - ALE = single loss expectancy (SLE) * annualized rate of occurrence (ARO)
5. **Annualized loss expectancy with a safeguard (ALE)**:
   - the possible yearly cost against a specific asset with an new EF and ARO
   - EF stays normally the same, ARO should be smaller (go to zero)
6. **Safeguard Costs**: assign each safeguard with a deployment value = ACS
   - ACS (Annual cost of the safeguard)
7. **Calculating safeguard cost/benefit**:
   - ALE before safeguard - ALE with safeguard - annual cost of safeguard (ACS) = value of the safeguard to the company
   - result < 0: safeguard is not a financially responsible choice
   - result > 0: this is the amount of money which could be saved when deployed (rate of occurrence is not a guarantee of occurrence)


Assign Asset Value (AV)
Calculate Exposure Factor (EF)
Calculate single loss expectancy (SLE)
Assess the annualized rate of occurrence (ARO)
Derive the annualized loss expectancy (ALE)
Perform cost/benefit analysis of countermeasures

1.6.2 risk treatments methods

Risk Mitigation: implement safeguards to eliminate vulnerabilities or block threats
Risk Assignment: placement of the cost of loss onto another entity / organization (e.g. outsourcing)
Risk Acceptance: when countermeasures cost more than the possible loss you can accept the risk
Risk Deterrence (en für Abschreckung): implementing deterrents for would-be violators of security and policy (e.g camera, motion detection)
Risk Avoidance: select alternate options / activities that have less associated risk
Risk Rejection: reject risk / ignore risk (unacceptable)
Residual risk: the risk witch is left after implementing the countermeasures

1.7 Privacy

Privacy: the right of the individual to control their personal data
pseudonymization: replacing data elements with pseudonyms

- can be reversed to make data meaningful

Anonymization: removing all relevant data so that it is impossible to identify the original subject / person

- USA PATRIOT Act
- EU GDPR

2 Week 02 - Identity and Access Management (IAM)

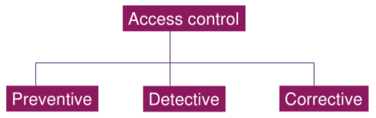
2.1 Controlling access to assets

- **access control**
  - any hardware, software, administrative policy / procedure that controls access to resources

- **Subject**
  - an active entity that access a passive object

  examples users, programs, computers, ...
- **Object**
  - a passive entity that provides information to active subjects

  example files, databases, services, storage media, ...

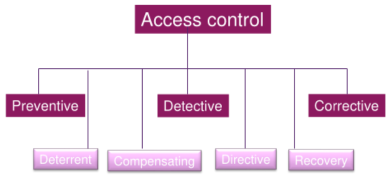


- **Preventive Access Control**
  - preventive control to stop unwanted / unauthorized activity from occurring

  example fences, locks, alarm system, ...
- **Detective Access Control**
  - attempts to discover / detect unwanted / unauthorized activity
  - can discover the activity only after it has occurred

  example security guards, motion detectors, ...
- **Corrective Access Control**
  - modifies the environment to return system to normal after an unwanted / unauthorized activity has occurred
  - try to correct problems that occurred because of a security incident

  example rebooting system, backup and restore plans, ...



- **Deterrent**
  - discourage violation of security policies
  - the subject decides itself not to take an unwanted action

  example security-awareness training, locks, security camera
- **Compensating**
  - provide various options to other existing controls to aid
- **Directive**
  - controls to force compliance with security policies

  example security policy requirements / criteria, escape route exit signs, monitoring, ...
- **Recovery**
  - extension to corrective controls
  - have more advanced abilities

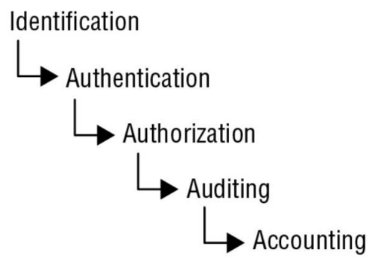
  example backups and restores, fault-tolerant drive systems
- **Physical controls**
  - items you can physically touch
  - prevent, monitor or detect contact with systems / areas

  example guards, fences, locked doors, ...
- **Technical / logical controls**
  - hardware / software used to manage access
  - provide protection for resources / systems

  example authentication methods, encryption, firewalls, ...
- **Administrative controls**
  - policies and procedures defined by organizations security policy
  - also known also management controls

  example policies, background checks, data classifications and labeling, ...

2.2 The steps of access control



2.2.1 Identification

- process of subject claiming an identity (e.g. username)

2.2.2 Authentication

- process of verifying that the claimed indent is valid (e.g. with password)
- type of authentication
  - password phrases
  - cognitive password
  - smart cards
  - tokens

  - (a)synchronous dynamic password tokens
  - onetime password generators
  - TOTP (Time-based One-time password)
  - HOTP (HMAC-based One-time password)
- authentication factors
  - type 1: **something you know**

    example password, PIN, passphrases
  - type 2: **something you have**

    example smart card, hardware token, memory card
  - type 3: **something you are / you do**

    example fingerprints, retina patterns, palm topology
  - additional factors
    - **somewhere you are** (geographic location)
    - **somewhere you aren't** (geographic location)

2.2.3 Authorization

- process ensures that the access to an object is possible with the rights and privileges
- access control models
  - **Discretionary Access Control (DAC)**
    - every object has an owner
    - owner can grant / deny access to any other subjects
    - NTFS / MS used this
  - **Role Based Access Control**
    - permissions are not assigned directly to users
    - users are placed in roles
    - privileges are assigned to roles
  - **Rule Based Access Control**
    - global rules that are applied to all subjects
    - firewall uses often this type
  - **Attribute Based Access Control**
    - uses of rules that can include multiple attributes
  - **Mandatory Access Control**
    - uses of label applied to both subjects and objects
    - user has label top-secret can access objects label top-secret
  - **Implicit deny**
    - as long as not explicit allowed it is denied
  - **Constrained Interface**
    - UI is rendered based on subjects privileges
  - **Access Control Matrix**
    - table that includes subjects, objects and assigned privileges
    - ACL are object focused and identify access granted to subjects for any object
  - **Capability Table**
    - similar to Access Control Matrix
    - Capability tables are subject focused and identify the objects that subjects can access
  - **Content-Dependent Control**
    - restrict access to data based on the content within an object
    - database view is an example for content dependent control
  - **Context-Dependent Control**
    - requires specific activity before granting users access
    - users add items to the shopping cart and then pays for it. He can't pay before something is in the cart
- Principles
  - **Need to Know**
    - grant only access to what the subject need
    - subjects has label classified, object has label classified but when subject don't need this object for it's work it shouldn't have access
  - **Least Privilege**

    * grant only this privileges they need to perform their work
    * write, create alter or delete data
  - **Separation of Duties and Responsibilities**
    * no single person should have total control over critical function / system

2.2.4 Auditing

- recording activities of a subject and its objects

2.2.5 Accounting

- linking a human to the activities of an online identity through auditing, authorization, authentication and identification

2.3 The common access control attacks

- **Access Aggregation Attack (passive attack)**
  - collect multiple non sensitive data and aggregating them to learn sensitive data
- **Password Attacks (Brute-force attack)**
  - attacks against online accounts
  - steal account database and crack the passwords offline
- **Dictionary Attacks (Brute-force attack)**
  - attempt to discover passwords by using every possible password in a predefined databases
- **Birthday Attack (Brute-force attack)**
  - finding collisions in hashes
- **Rainbow Table Attacks**
  - similar to dictionary attacks but the passwords are precomputed hashes
- **Sniffer Attacks**
  - capture traffic from the network
- **Spoofing Attacks**
  - pretending to be something / someone else
- **Social Engineering Attacks**
  - gain information over social interacting with other people
- **Shoulder surfing**
  - tries to look over the shoulder of target
  - special form of social engineering
- **Phishing**
  - Spear phishing
    * targeted to a specific group of users (e.g. employees within a specific organization)
  - Whaling
    * variant of phishing targeting high-level executives (CEOs)
  - Vishing
    * like phishing but over instant messaging and VoIP

2.4 Protection mechanisms

- Data hiding
  - store data in a logical storage part where the subject has no access => data is not visible to the subject
- security through obscurity
  - idea of not informing about an object being present and hoping the subject will not discover the object
  - does not implement any form of protection. It hopes something important is not discovered
- Encryption
  - the art and science of hiding the meaning of a communication from unintended recipients

3 TODO Week 03 - Linux

4 Week 04 - Symmetric and key exchange

4.1 Cryptography concepts

- **plaintext message**: message before coded into a form
- **ciphertext**: cryptographic algorithm was used on a plaintext message
- **cipher**: encryption algorithm

- **cryptographic key**: a (large) number
- **one-way functions**: mathematical operation that produces easily output values but it makes impossible to retrieve the input value
- **reversibility**:
- **Nonce**: Number only used once
  - can be a counter
  - is public
  - used to make sure the key is not re-used twice
- **Initialization vector (IV)**: a random bit string
  - same length as block size
  - is XORed with the message
  - used to create unique ciphertext with the same key
- **Confusion**: relationship between plain text and key is complicated
  - substitution of bytes adds confusion
- **Diffusion**: change in plaintext results in multiple changes in the ciphertext
  - small change in input => big change on the output
  - permutation of bytes adds diffusion
- **Kerckhoffs's Principle**: a cryptographic system should be secure even if everything about the system, except the key, is public knowledge
- **SP-Network**: algorithm that uses repeated substitution and permutation operations
- **One time pad**: unbreakable cipher
  - message length = key length, XOR key with message
  - perfect secrecy
  - but not practicable (1GB file needs 1GB key)

4.2 Symmetric cryptography

- symmetric key algorithms
  - uses **shared secrect** key (requireds a secure method of exchanging the key)
    - ∗ used by all parties for encryption / decryption
  - large sized keys => verry difficult to break
  - only used for **confidentiality**
  - does not implement **nonrepudiation**
  - does not implement **message integrity**

  pro extremley fast, many CPUs have AES instruction set
- **stream cipher**
  - one-time pad approximation with infinite pseudo random key stream
  - works on messages of any length

  pro fast and low memory footprint

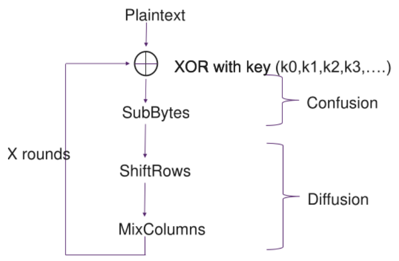
  pro can seek to any location in the stream

  con never reuse key + nonce

  con do not protect the ciphertext (no guaranteed integrity)
- **block cipher**
  - take input of a fixed size; return output of same size
  - transformation from message to cipher trought confusion / diffusion
  - normally a SP-Network
  - AES is a SP-Network
    - ∗ almost everything uses AES
- **basic substitution box**
  - input is substituted
- **basic permutation box**
  - bits will be rearranged

4.2.1 AES

- **AES**: standard built arround the **Rijndael algorithm**
  - AES = **A**dvanced **E**ncryption **S**tandard
  - SP-Network with 128-bit block size
    - ∗ 10, 12, 14 rounds
    - ∗ SubBytes, ShiftRows, MixColumns, Key Addition

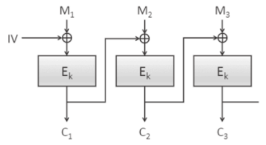


4.2.2 Mode of operations of block ciphers

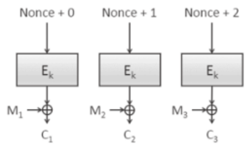
- **Electronic Code Block (ECB)**
  - just encrypt each block one after another

- ECB is not not recommended

- **Cipher Block Chaining (CBC)**
  - XOR output of each cipher block with the next input
  - not parallelizable
  - betterthan ECB



- **Counter Mode (CTR)**
  - encrypting a counter to produce a stream cipher
  - parallelizable
  - convert block cipher into a stream



4.3 Diffie-Hellman Key exchange

4.3.1 discrete logarithms problem

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

$$7^2 = 4(\text{mod} 9)$$
$$2 = dlog_{7,9}(4)$$

- to calculate the log is very difficult => brute force

4.3.2 Diffie-Hellman

1. both sites agree on parameter p (prime) and generator g (prime)
2. Alice and Bob select randomly their private values **a** and **b**
   (a) $1 < a, b < p$
3. calculate public key
   (a) $g^a modp$ bzw. $g^b modp$
4. Swap public keys over the internet
5. combining public key with own private key
   (a) $(g^b)^a \text{mod } p = g^{ab} \text{mod } p$
   (b) $(g^a)^b \text{mod } p = g^{ab} \text{mod } p$
6. Shared key = pre-master secret, used to derive session keys

- **Emphemeral mode**
  - new key exchange for evey new session (PERFECT FORWARD SECRECY)
  - Re-using a key doe's not make sense when just generate new ones
  - not necessarily every message (but pretty often)
    * if someone is sniffing the traffic and breaking the keys, he don't get any of the previous messages
    * DH provides self-healing property

4.3.3 Elliptic Curve Cryptography (ECDHE)

- **ECDHE**
  - drop in replacement for regular Diffie-Hellman
  - becoming the standard
  - elliptic curve discrete logarithm problem (ECDLP) is slight more difficult than discret logarithm
- elliptic curve needs a lot less bits for same security

5 Week 05 - Assymetric and hash functions

5.1 RSA

- two use cases
  1. encryption: encrypt with others public key => decrypt with own private key
  2. signing: encrypt with own private key => decrypt with public key
- public key (e, n), private key (d)
  - e usually small (2, 3) for efficiency
  - n is very lager number by multiply two primes number together ($p_1 \cdot p_2$)
- based on **factoring problem**:
  - hard of factoring the product of two large prime numbers => **time complexity**
- **one-way trapdoor function**:
  - no way to undo the encryption function unless you know the trapdoor

  - known the factors of **n** is the trapdoor
- Encrypt message witch public key (e,n)
  - $c = m^e \bmod n$
- Decrypt message with private key (d)
  - $m = c^d \bmod n$
- **Φ -Funktion**:
  - n positive integer, m prime
  - $m^\Phi(n) = 1 \bmod n$
  - $d = \frac{(k \cdot \Phi(n)+1)}{e}, k \in \mathbb{Z}$
- generating key pairs is time-consuming, should be done rarely
  - e is almost always 3 or 65537

5.1.1 Build parameters

1. Choose p_1 and p_2
   (a) $n = p_1 \cdot p_2$
   (b) n is public, p_1, p_2 private
2. Calculate Φ(n)
   (a) $\Phi(n) = \Phi(p_1 \cdot p_2) = \Phi(p_1) \cdot \Phi(p_2) = (p_1 - 1) \cdot (p_2 - 1)$
3. Choose exponents e and
   (a) choose e freely
   (b) $d = \Phi(n) - \lfloor \frac{\Phi(n)}{e} \rfloor \cdot$
   (c) Verification: $d = \frac{k \cdot \Phi(n)+1}{e}$

5.2 Encrypting using RSA

- is very weak for short messages
  - padding added to short messages (Optimal asymmetric Encryption padding - OAEP)
  - OAEP: pseudo random padding that introduce an IV process and hashes it
  - receiver has to do the exact same padding to match the messages
- Encryption using RSA is not very common (slow)

5.3 Signing using RSA

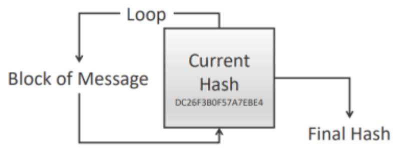
- Same process as normal encryption. Just encrypt message with own private key
- digital signature often sent as a challenge (prove its identity)
  1. client sent message to server
  2. server sign it with its private key
  3. client decrypt the signed message using public key
  4. compares original messages with message from server

1. message is first hashed
2. hash is signed and sent with message
3. the receiver hashes message, decrpyt the signature compare hashes

- **DSA**: Digital Signature Algorithm:
  - RSA in some years to slow => DSA will become standard
  - DAS only for signing

5.4 The hash function

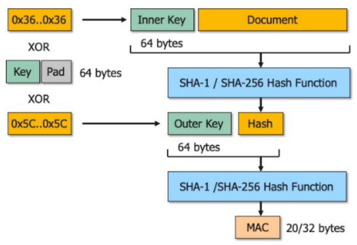
- takes message of any length, returns hash of fixed length
- hash functions iterative jumble blocks of messages after another
- output should not look like based on input
- small changes input => big changes output



- strong hash function:
  1. quick, but not too quick
  2. it has to introduce diffusion
  3. no possible to reverse
  4. given messages with its hash, no other messages hashes to the same thing
     - that is a collision
  5. we can't find tow messages with same hash
- **SHA-2 / SHA-3**
  - standard
  - shouldn't be used for passwords (they are to fast)
- **PBKDF2 (Password Based Key Derivation Function 2)**:
  - should used for passwords / login
  - useless for everything else
- **Bcrypt**
  - alternative to PBKDF2

- Usage of Hashes:
  1. Digital signature
  2. Symmetric cryptography is vulnerable to message tampering
     - Hashes let us ensure that messages hasn't changed

  - **Message Authentication Code (MAC)**
  - Hash(K|C) is append to the ciphertext C
  - if receiver hasshes key and ciphertext and its equal to the receiver messages hash, nothing changed
- **HMAC**
  - Split key in two
  - hash twice with each key



6 Week 06 - Complete cryptography systems

6.1 Public cryptography

- **Digital certificates**
  - digital signature proves access to private key (doesn't prove trust)
  - digital certificates verify ownership of a public key
- **Certificate Issuance**
  1. generate RSA / DSA private and public key
  2. creates a **Certificate Signing Request (CSR)**
     - CSR is a certificate which has to be verified
  3. CSR is sent to a **Certification Authority (CA)**
  4. CA does some identification checks
  5. CA creates and signs the certificate with its private key
- **Certificate Use**
  1. decrypt signature from server with public key from certificate
  2. in order to trust the public key you have to verify the digital signature at the bottom of the certificate
- **Chain of trust**
  - to trust server a, we have to trust the issuer of the certificate
  - to trust the issuer, we have to trust the next issuer; and so on
  - root certificates are trusted because built into OS / Browser / etc.

6.2 Complete cryptographic systems

- **Authenticated Encryption with associated data (AEAD)**
  - guarantee the confidentiality and integrity using MAC
  - encrypted message
  - message and additional data authenticated
- **AES Galois Counter Mode (GCM)**
  - it's a AEAD protocol
  - AES in CTR and Galois Message Authentication Code (GCM) as MAC / GMAC
- **ChaCha20$_{\textbf{Poly1305}}$**
  - it's a AEAD protocol
  - ChaCha20 stream cipher, Poly1305 a MAC
  - used on mobile phones, on CPUs without AES instructions ChaCha20 is faster
- Protocol Handshakes
  1. Handshake phase
     - Agree on set of cryptographic protocols
     - perform key exchange, obtain session keys, IVs, ...
     - verify authentication using public key
  2. Transport / Record Phase
     - encrypt message using cipher
     - include MAC to verify integrity
- Common Protocol Issues
  - not secured with MAC
  - no time-stamp or counter included
  - no public key authentication used
  - reuse of Nonces or IVs

7 Week 07 - TLS

7.1 Introduction to TLS

- OSI Layer 4+
- based on TCP
- responsible for
  - fragmentation into TLS PDUs

    - compression of PDUs before encryption
    - authentication of PDUs
    - encryption of PDUs
- SSL (Secure Socket Layer) previous name, now TLS (Transport Layer Security)

7.1.1 TLS Architecture

- **TLS connection**
    - transport that provides a suitable type of service
    - peer-to-peer
    - transient
    - is associated with one session
- **TLS session**
    - association between a client and server
    - defines a set of cryptographic security parameters (shared over multiple connections)
    - used to avoid expensive negotiation of new parameters
- **Connection State Definition**
    - MAC
    - key
    - IV
    - sequence number
- **Session State Definition**
    - Session identifier
    - peer certificate (may be null)
    - compression method
    - cipher spec
    - master secret (48-byte)
    - is reusable flag

7.1.2 History

- SSL 1 - TLS 1.0 shouldn't used (broken)
- TLS 1.1 - TLS 1.3 are OK
- TLS 1.3 - Clean up
    - remove unsafe / unused features
    - performance: 1-RTT and 0-RTT
    - improve security with modern techniques
    - encrypt more of the protocol, (almost all handshakes messages are encrypted)
    - backwards compatibility

7.2 TLS Protocol

- Record header
    - type (which TLS message is sent, 1 byte)
    - version (2 bytes)
    - length (2 bytes)
- application data
    - max 2^{14} bytes per segment
    - one TLS-message can be longer than one segment
- TLS-Records types:
    - **TLS Handshake** (ClientHello, ServerHello, Certificate, ServerHelloDone)
    - **TLS Change CipherSpec**
    - **TLS Alert** (Warning, Fatal, Session is closed immediately)
    - **TLS Application Data** (Transmission of encrypted payload)

7.2.1 TLS Handshake

- allows authentication
- negotiate encryption / MAC algorithms
- negotiate cryptographic keys
- 4 phases

1. Phase
    (a) sent ClientHello with random bytes R_c
    (b) server response with ServerHello and random bytes R_s
2. Phase
    (a) server normally sends now it's X.509 certificate in a **Certificate** message
    (b) optional: **ServerKeyExchange** containing the server part of DH secret
    (c) optional: **Certificate Request** to authenticate client is send from server
    (d) after the server hello phase the server sends a **ServerHelloDone** message
3. Phase
    (a) client sent it's certificate if requested in a **Certificate** message
    (b) if certificate from server received encrypt random premaster secret with public key from certificate and set it to the server in a **ClientKeyExchange** message. or send it's DH parameters
4. Phase
    (a) Client sends **ChangeCipherSpec** announcing that now the new parameters are used followed by **Finished** message
    (b) Server does the same on

it's site

5. Application data can be transmitted

- **Forward secrecy (FS)**
  - FS protects past sessions against future compromises of keys or passwords
  - over RSA public key encryption has no perfect forward secrecy
  - over DH key exchange with perfect forward secrecy
- **Computing TLS master secret**
  1. Pre master secret has to be exchanged
  2. master secret is calculated by both parties
     - for pre master secret exchange can RSA or DH be used
     - RSA, client generates 48-byte pre master key and sent it encrypted with public key of server back
     - DH: normal DH key exchange

7.2.2 TLS Alert protocol

- 2 bytes length
- first byte: level - warning or fatal (fatal immediately termination)
- second bytes: code that indicates the specific alert

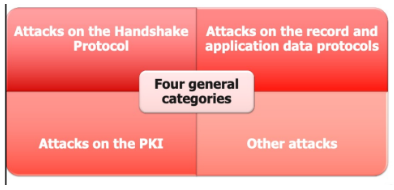
7.2.3 TLS Session Resumption

- instead of of negotiating new security parameter the previous session can be used
  1. **ClientHello** with previous **Session ID**
  2. if session is still in cache on server side the session is reused and **ChangeCipherSpec** and **Finished** is sent
  3. client response with **ChangeCipherSpec** and **Finished**
  4. If no match is found on server a full handshake is perfomred

7.2.4 Heartbeat Protocol

- runs on top of TLS Record protocol
- assures the sender is still alive
- generates activity across the connection during idle periods

7.2.5 SSL / TLS Attacks



Attacks on the Handshake Protocol
Attacks on the record and application data protocols
Four general categories
Attacks on the PKI
Other attacks

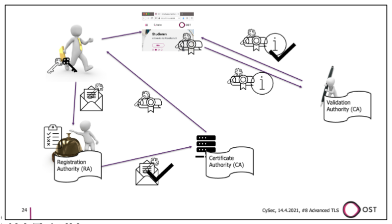
7.3 TLS 1.3

- 1-RTT
  - clients sends **ClientHello** with **KeyShare**
  - Server sends **ServerHello** with **KeyShare**, **ServerParameters** and **Authentication**
  - client sends **Authentication** and first **Application Data**
- Cipher Suites
  - Key Exchange: DHE, ECDHE
  - Authentication: RSA, ECDSA, EdDSA
  - Cipher Suites:
    - $TLS_{AES128GCMSHA256}$
    - $TLS_{AES256GCMSHA384}$
    - $TLS_{CHACHA20POLY1305SHA256}$
    - $TLS_{AES128CCMSHA256}$ (IoT)
    - $TLS_{AES128CCM8SHA256}$ (IoT)

8 Week 08 - Advanced TLS

8.1 Public Key Infrastructure (PKI)

- **Public Key Infrastructure (PKI)**
  - used to bind public key to an identity of a person / organization
  - registration process by a **Registration Authority (RA)**
  - issuance of a certificate by a **Certificate Authority (CA)**
  - CA can be validated by independent **Validation Authority (VA)**
  - artifact of binding is the **certificate**



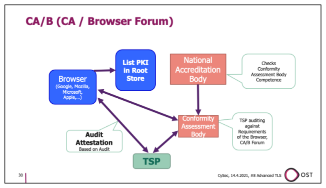
Studieren
Validation Authority (CA)
Registration Authority (RA)
Certificate Authority (CA)
24
CySec, 14.4.2021, #8 Advanced TLS
OST

8.2 X.509 Certificates

- standard for the format of public key certificates
- public key certificate = digital / electronic certificate
  - proving ownership of a public key
  - information about the key
  - identity of its owner (aka. **subject**)
  - entity that has verified the certificate's content (aka. **issuer**
- certificate quality
  - **Domain Validated (DV)**
    - check if owner has right to use the domain
  - **Organization Validated (OV)**
    - after the company name, domain name and other information through the use of public databases is checked
  - **Extended Validation (EV)**
    - only after a strict authentication producer is passed
  - **Qualified Website Authentication Certificate (QWAC)**
  - validity information
    1. **Certificate Revocation List (CRL)**
    2. **Online Certificate Status Protocol (OCSP)**

8.3 Trust Service Providers - Certificate Authorities

- **Trust Service Provider (TSP)**
  - establish trust between communicating parties
    - providing identity information
    - secure authentication
    - integrity protected communication
    - encrypted communication
- **Certificate Issuance**
  - **subscriber** prepares based on public key a **certificate signing request (CSR)**
  - the CA will register the request, verify data, and if everything OK => CSR turns into a X.509 certificate
- CA hierarchy
  - **Root CA** the last / first part of the chain of trust. Apps, Computers, Users trust the Root CA
  - **Issuing CA** issues certificate to end entities (Intermediate CA / Subordinate CA)
  - because of security / logical separation / flexibility these are roles are separated
- **CA / Browser Forum (CA / B)**


CA/B (CA / Browser Forum)
Browser (Google, Mozilla, Microsoft, Apple,...)
List PKI in Root Store
National Accreditation Body
Checks Conformity Assessment Body Competence
Conformity Assessment Body
TSP auditing against Requirements of the Browser, CA/B Forum
Audit Attestation Based on Audit
TSP
30
CySec, 14.4.2021, #8 Advanced TLS
OST

- Principle elements of conformity assessment (How to become a TSP)
  1. Stage: Document assessment
     - technical, function, organizational security measures
  2. Stage On-Site assessment
     - verify implementation of security measures (including technical and pen testing)
  3. Results report
     - identification of the TSP, service and policy content and summary
- **Certificate pinning**
  - operators limit the who is allowed to issue a certificate for the domain (pin)
  - Pin the root
  - Pin the intermediate CA
  - Pin the end entity certificate (also called the **SSH model**

9 Week 09 - Authentication Protocols

9.1 Authentication Schemes

- Authentication Schemes Classification
  1. A1. **Basic Authentication**: classical username / password pair transmitted clear

cons prone to passive sniffing

2. A2. **One Time Passwords**: transmitted clear but used only once
   - **pros** efficient means against passive sniffing attack
   - **cons** do not protect against active Main-In-the-Middle (MITM)
3. A3. **Challenge / Response**: function of password and one-time challenge
   - **pros** efficient means against passive sniffing attack
   - **cons** do not protect against active Main-In-the-Middle (MITM)
4. A4. **Anonymous Key Exchange**: exchange credentials over unauthenticated secure channel
   - **pros** algorithms like DH can be used to calculate a common secret
   - **cons** vulnerable to MITM because endpoint do not authenticate themselves
5. A5. **Zero-Knowledge Password Proofs**: does not permit offline-based password attacks
   - **pros** Brute Force impossible because the DH parameters are random
   - **cons** password database on server could be stolen
6. A6. **Server Certificate plus User Authentication**: transmitting password over unilaterally authenticated secure channel (using DH and signature)
   - **pros**
   - **cons** password database could be stolen from the server
7. A7. **Mutual Public Key Authentication**: bilateral use of public key signatures
   - **pros** with client certificates no need to store user credentials, client certificates can be stores on an smart card protected by a PIN (strong 2-FA)
   - **cons** non besides the worst case of a root CA compromise

9.1.1 A3. Challenge / Response

- the proper way of authenticating over an insecure channel
- **Challenge / Response based on MAC**
  - Hash(id + challenge + random bytes client + clients password) is the MAC
  - if both the transmitted MAC and the server side calculated MAC is equal => authenticated
- **Challenge / Response based on Digital Signatures**
  - same idea as above
  - signature of Hash(ID, R_U, R_S) is append and compared

1. Kerberos
   - **Key Distribution Center (KDC)**
     - every principal has a secret master key (derived from login password) registered with the KDC
     - all master keys are stored in the KDC database, encrypted with the KDC master key
   - **Simplified Kerberos Protocol**
     - algorithm
       (a) Alice
         i. Alice derive Master Key from Login password
         ii. Send encrypted time to the
       (b) KDC
         i. KDC decrypts time => valid?
         ii. Generate Session Key
         iii. Encrypt Session Key with Master Key from Alice
         iv. Encrypt Session Key and name (alice) with master key from

Bob (ticket)

v. Send session key and ticket to alice

(c) Alice

i. Decrypt session key

ii. Encrypt time and name with session key

iii. send encrypted time with ticket to Bob

(d) Bob

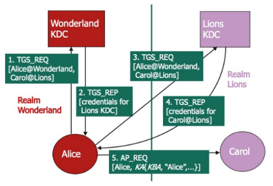
i. decrypt ticket (gets session key and name)

ii. decrpyt time (is time and name valid?)

cons principals master key is used very often, often time login credentials or credentials in temporary storage

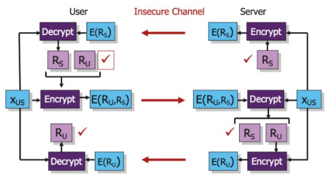
solution Long-Lived **Ticket-Granting Ticket**

- **Session Key and Ticket-Granting Ticket (TGT)**
  - the KDC creates instant of a Session key for Alice to Bob only a Session Key Alice (S_a)
  - Name and Session Key S_a are encrypted using the KDC master key = TGT
  - TGT is used to create new Session Keys between Alice and Bob
- **Inter-Realm Authentication**

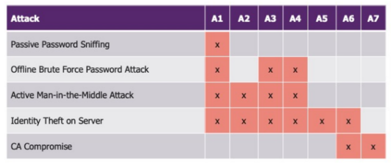

Wonderland KDC
Lions KDC
1. TGS_REQ [Alice@Wonderland, Carol@Lions]
2. TGS_REP [credentials for Lions KDC]
3. TGS_REQ [Alice@Wonderland, Carol@Lions]
4. TGS_REP [credentials for Carol@Lions]
Realm Wonderland
Realm Lions
Alice
Carol
5. AP_REQ [Alice, KA{KBA, "Alice",...}]

9.1.2 A5. Zero-Knowledge Password Proofs

- DH parameters are encrypted with the user password and then transmitted
  - resulting in a **Strong Ephemeral DH Secret** (x_{us})
- using this DH secret a mutual Challenge / Response is built



9.1.3 Summary



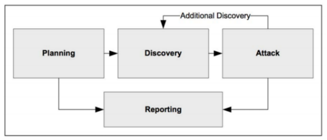
Attack	A1	A2	A3	A4	A5	A6	A7
Passive Password Sniffing	x						
Offline Brute Force Password Attack	x		x	x			
Active Man-in-the-Middle Attack	x	x	x	x			
Identity Theft on Server	x	x	x	x	x	x	
CA Compromise						x	x

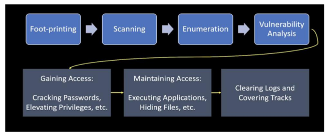
- Challenge / Response based authentication schemes are the standard

10 Week 10 - Introduction to Ethical Hacking and Penetration Testing

10.1 Introduction to Ethical Hacking

- Attack concepts
  - **Hacking**
    - ∗ exploit vulnerabilities
    - ∗ security control compromise
    - ∗ produce behaviors outside of systems original intend
  - **Ethical hacking**
    - ∗ validate, audit and report on system / software vulnerabilities
    - ∗ reporting vulnerabilities
- Hacker types
  - **Black Hat**
  - **Grey Hat**
  - **White Hat**
  - **Script Kiddie**
  - **Cyber Terrorist**
  - **State Sponsored** (Hacker employed by the government for offensive and defensive
  - **Hacktivism** (Hacker whoses activity is aimed at promoting a social / political cause)
- legal aspect of pen testing
  - pen testing contracts
  - **Statement of Work (SoW)**
    - ∗ what is allowed, time line, which activities to be perfomred
  - **Non Disclosure agreement (NDA)**
- penetration testing methodologies




Foot-printing
Scanning
Enumeration
Vulnerability Analysis
Gaining Access: Cracking Passwords, Elevating Privileges, etc.
Maintaining Access: Executing Applications, Hiding Files, etc.
Clearing Logs and Covering Tracks

10.2 Malicious code

- **Zero-Day Attack**
- source of malicious code
  - Script kiddie
  - **Drive-by download** (you don't have to click on anything)
  - **Advanced persistent threat (ATP)** (advanced technical skills and significant financial resources
    - often military units, intelligence agencies, affiliated with **government agencies**
  - **Viruses**
    - two functions: **propagation** and **destruction**

10.3 Virus propagation techniques

- **master boot record infection**
  - only small part of virus in MBR, MBR loads the whole virus from somewhere else (HDD)
- **file infection**
  - infects executable files (.com, .exe)
- **macro infection**
  - e.g. Office Macros
- **service injection**
  - inject itself into trusted runtime processes of the OS

10.4 Malware Technologies

- **Multipartite Viruses**
  - uses more than one progpagation
- **Stealth Viruses**
  - hide themselves and fool antivirus
- **Polymorphic Viruses**
  - modify their own code as they travel from system to system
- **Encrypted Viruses**
  - use cryptographic techniques to avoid detection
  - short segment **virus decryption routine**
- **Logic Bomb**
  - infect system and lie dormant until they are trigger by a condition
- **Trojan Horses**
  - appears kind but carries malicious **behind-the-scenes payload**
- **Keystroke logging**
  - logging the keys struck on a keyboard (HW or SW)
- **Ransomware**
  - infect target and encrypt files; key only known by the creator
- **Worms**
  - propagate themselves **without requiring any human intervention**
  - e.g **Stuxnet**
- **Spyware**
  - monitors your action and transmits details to remote
- **Adware**
  - display advertisements

10.5 Antivirus

- software to prevent, detect and remove malware
- not only from viruses
- Antivirus detection
  - **signature based**
  - **heuristic based** (analyze behavior of software)
  - **data integrity** (sudden changes in executable file may be an sign of malware, except for updates)

10.6 Applications Attack

- **Buffer Overflows**
  - input to large for buffer => overflow
- **Time of Check to Time of Use (TOC/TOU)**
  - timing vulnerability; program check access permission to far
- **Back Doors**
  - undocumented command sequences to bypass normal access restrictions
- **Escalation of Privileged**
  - expand access from normal user to more administrative access

- **Rootkits**
  - exploit known vulnerabilities in OS and provide the hacker the possibility to increase his access to the root level

10.7 Web Application Attack

- **Cross-Site Scripting (XSS)**
  - embed custom JS in to the website
- **Cross-Site Request Forgery (XSRF or CSRF)**
  - embed code in one website that sends a command to a second website
- **Dynamic Web Applications**
  - possible way from webserver (DMZ) to gain access to DB server (LAN)
  - firewall rules has to be corret
- **SQL Injection**
  - inject malicious SQL

10.8 Network security

- why hacking network devices
  - old protocols
  - long system life cycle
  - no malware detection
- **Attacks**
  - **Denial-of-Service (DoS)**
    * **System overload** and **Link overload**
    * **SYN flooding**
    * **Service request floods**
    * **Application level DoS** (usage of a vulnerability to cause a crash / hang / freeze / consume all resources)
    * **Permanent DoS** (installing compromised HW updated to destroy the HW)
  - **Botnets**
    * bot / zombie are controlled over the C&C (Command & Control) from the botmaster
    * **Distributed DoS (DDoS)**
  - **Man-in-the-Middle (MITM)**
    * active and passive
  - **Man-in-the-Browser**
    * Browser Plugin installed; Capture form data (username / password)
  - **Eavesdropping**
    * listen to communication traffic for the purpose of duplicating it
    * Wireshark
  - **Impersonation / Masquerading**
    * pretending to be someone / something you are not
    * unusually authentication credentials have been stolen
    * Not the same as **Spoofing** (Spoofing uses a false identity but without proof)
  - **Replay Attacks**
    * replay a captured (e.g with Wireshark) traffic to reestablish communication
  - **Modification Attacks**
    * captured packets are altered and then played against a system

10.8.1 Session hijacking

- **Session IDs**
  - used to establish a stateful connection
  - stored in cookies, passed in URL or hidden fields
- **Session hijacking with session predicting**
  - watch session IDs for pattern and tries to predict
- **Session hijacking with session fixation**
  -
- **Session hijacking with Cross-Site Scripting**
  - embed JS in website sends Session ID to attacker when client established a connection
- **Session hijacking with Cross-Site request forgery**
  - Client establish connection to server; if user clicks on a malicious link the attacker can

use the established session

- **Session hijacking with TCP/IP hijacking**
  - Normal TCP 3-way handshake
  - attacker responds quicker to the SYN/ACK then the regular client

11 Week 11 - Federation

11.1 Federations

- **Federation Identity Management**
  - agreements, standards and technologies to make **identity** and **entitlements portable** across autonomous identity domains
- **Federated Identity**
  - is the means of linking a person's electronic identity and attributes, stored across multiple distinct identity management systems
- **Federation**
  - a set of organizations agreeing on a common set of rules and standard

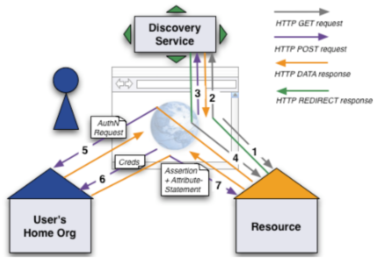
goal cooperate in inter-organizational authentication, authorization and accounting

- **Identity Provider (IdP)**
  - perform authentication
- **Service Provider (SP)**
  - offers service and performs authorization

11.2 Shibboleth

- protocol to implement

1. Phase - User connect to resource and is redirected
   - If already a valid Shibboleth session is available, you get access to **the resource**
   - if no session is active you get redirected to the **Discovery Service**. This sends you a page with all **Home Organizations**
2. Authentication Request
   - choose **Home Organization** and the Discovery Service responses with a redirect to the **session initiator** of the resource
   - the session initiator creates an authentication request for the Home Organization and sends it through the browser to the Home Organization
   - Home Organization evaluates the authentication request and present a login page
3. Authentication and Access
   - user provides his credentials; credentials are checked and an **assertion** including the users attributes is created
   - the assertion is submitted through the browser back to the resource. Resource can perform authorization checks



11.3 XML Security and SAML

- **Secure Assertion Markup Lnaguage (SAM)**
  - XML-encoded assertions about authentication, attributes and authorization
  - Allows single sing-on solution for web services

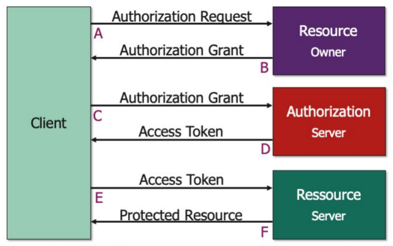
11.4 OAuth 2 Authorization Framework

goal app can access you data at some service but without knowing your password

Roles

- **Client**
  - making requests on behalf of the resource owner and with its authorization
- **Resource Owner**
  - entity capable of granting access to protected resources
- **Resource Server**
  - server hosting the protected resources
- **Authorization Server**

- issuing access tokens to the client



11.5 OpenID Connect Authentication Layer

- solves the problem of
  - one ID per site
  - managing passwords per site
  - layer on top of OAuth 2.0
- decentralized mechanism for **single sign-on**
  - access different sites using the same identity
  - website never sees your password
- used by Google, Microsoft, PayPal, ...
- access token contains:
  - user ID
  - approved scopes
  - expiration

example **SwissID**

- **Relying Party (RP)** same as Service Provider (PostFinance)
- **OpenID Provider (OP)** same as Identity Provider (SwissID)
- RP (OAuth Client) requests a signed ID Token authenticating User (Resource Owner) from OP (Authorization/Resource Server)

12 Week 12 - Email Security

12.1 S/MIME

Multipurpose Internet Mail Extension (MIME)
S/MIME type 1: multipart entity with subtype **multipart/signed**. contains two bodies:

1. content to be signed (regular mime)
2. digital signature over content part, with subtype **pkcs7-signature**

signature also possible over multipart/mixed
S/MIME type 2: MIME content carried withing an **PKCS#7 Signed Data Object (*smime-type=signed-data;)**

- pro: MIME content is not prone to changes during transfer
- contra: in order to read the mail, the receiver's mail client must support S/MIME

S/MIME - Encrypted Message Format:

- Content-Type: **application/pkcs7-mime; smime-type=envelped-data;**
- to send an encrypted message to multiple recipients encrypt content symmetric and then encrypt key with each others public key

PKCS #7 - Public Key Cryptography Standard: contains a **SignedData** block

- version, digestAlgorithms, contentInfo, certificates, crls, signerInfos (several signers possible)

SignedData contains **SingerInfo** block

- version, issuerAndSerialNumber, digestAlgorithm, authenticatedAttributes, digestEncryptionAlgorithm, encryptedDigest (signature), unauthenticatedAttributes

Storage of keys

- **Signature Key**
  - equivalent to a person's digital identity
  - must not be more than a single copy
  - best store on smart cards
- **Encryption Key**
  - backup copy protects against information loss
  - in a corporate environment deputy (stv.) should be able to access the encrypted emails if owner is absent (vacation, accident, sickness, death)

12.2 SPAM

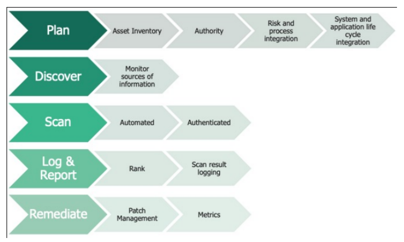
Sender Policy Framework (SPF): a SPF record indicate which servers are allowed to send e-mail on behalf of the domain
Domain Keys Identified Mail (DKIM): SMTP server hash content and encrypt it with private key; receiver can decrypt it with public key (queried over DNS) and compare hash (basically a signature)
Black Listing: blocks mails from IPs
Grey Listing: refuse deilvery of mail with unknown triplet; Greylist Triplet:

- IP address
- envelope sender address
- envelope recipient address

13 Week 13 - Detect and Respond

Technical vulnerability management is a security practice to proactively mitigate / prevent exploitation of vulnerabilities. The process involves the:

1. identification
2. classification
3. remediation (en für Sanierung / Behebung)
4. mitigation of vulnerabilities



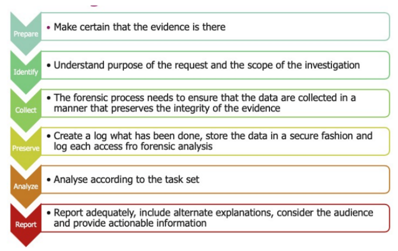
Plan is the process of Risk and process integration, integrating with asset inventory, establishment of a clear authority to review vulnerabilities and system and application life cycle integration
Discover step involves monitoring sources of information about known vulnerabilities (e.g CVSS, NIST, CERT, ...)
Scan regulary software, systems and networks for vulnerabilites and proactively address those. But scanning can cause disruptions and huge amount of data
Log & Report after scan the result should be logged so that a person can verify the activity. Each vulnerability should be ranked with skill to fix, availability to attackers, privilege gained and risk and impact if exploitation is successful
Remediate perform patches. Ideally every discovered vulnerability should be patched immediately. But availability and the impact of a patch needs to be taken in account. Special care if multiple automated patching systems are used
Security Event an occurrence consider by an organization to have potential security risk. A **Security incident** is an occurrence that actually or potentially jeopardizes the confidentiality, integrity or availability (CIA).
Security Event Management (SEM) should help to identify threats that may lead to an security incident, maintain integrity, conditionality and availability. For this the logs has to go through the following steps:

1. **Normalization** - data in common format
2. **Filtering** - assigning priorities and set a large amount aside
3. **Aggregation** - group multiples events in one bigger (where useful)

To generate alerts you can perform **Pattern matching** on logs, detect a scan (**Scan detection**), to many occurrence of a security event (**Threshold detection**) or detect correlation between multiple events which indicate a security incident (**Event correlation**).
Thread Intelligence is the knowledge established as a result of analyzing information about potential / current attacks.
Cyber Attack Kill Chain consist of Reconnaissance, Weaponization, Delivery, Exploit, install action, C&C, Action
Security Incident Management Process is a process of:

- **Preparation**: define what is a incident, answer WHO-WHEN-WHAT, response and reporting procedure, guidelines for communication
- **Detection & Analysis**: detected then magnitude (e.g. number of affected devices), severity (what it the sensitivity of the data), urgency (active problem, threat or an event-in-progress). The analysis needs to determine whether immediate actions is needed.
- **Containment, Eradication & Recovery**: most incidents require some sort of containment. Strategies for dealing with various types of incident must be planned well in advance (system isolation, disable system-level accounts, terminate active session, ...). During recovery restore systems to normal operation and harden systems where applicable
- **Post-Incident Activity**: follow-up report and lessons learned

Phases of Digital Forensics Process:



- Make certain that the evidence is there
- Understand purpose of the request and the scope of the investigation
- The forensic process needs to ensure that the data are collected in a manner that preserves the integrity of the evidence
- Create a log what has been done, store the data in a secure fashion and log each access fro forensic analysis
- Analyse according to the task set
- Report adequately, include alternate explanations, consider the audience and provide actionable information

Threat and incident management best practice

1. Cybersecurity resilience (en für Belastbarkeit): Tech-

nical vulnerability management, Security event logging, Security event management, Threat intelligence, Cyber attack protection
2. Security incident management: Security incident management framework, Security incident management process, Emergency fixes, forensic investigation

14 Week 14 - Web Application Security

User Credentials: The simplest attack to hack a system is to attack the user credentials using e.g. **Spear Phishing**, **Password breached**

- **A1: Injection**: Injection flasws such as SQL, OS, and LDAP injections (untrusted data is sent to an interpreter as a part of a command.
- **A2: Broken Authentication**: authentication / session management implemented incorrectly
- **A3: Sensitive Data Exposure**: APIs often don't protect sensitive data properly (financial, PII, PHI)
- **A4: XML external Entities (XXE)**: older / poorly configured XML processors evaluated external entity references
- **A5: Broke Access Control**: what the users is allowed to to is often not properly enforced
- **A6: Security Misconfiguration**: most commonly seen issue, insecure default configuration
- **A7: Cross-Site Scripting (XSS)**: occurs when site include untrusted data in website without proper validation / escaping
- **A8: Insecure Deserialization**: untrusted data deserialization can leads to remote code execution (do not trust data from user)
- **A9: Using Components with known vulnerabilities**: check regularly for known bugs in your library
- **A10: Insufficient Logging & Monitoring**: allows attack to maintain persistence, tamper, extract or destroy data. Most breaches are over 200 days old until detected, typically detected by external parties rather than internal processes

15 End