1 Network Models (OSI Layers / IP Stack)

Tabelle 1: OSI Layers

Nr	Layer	Protocols	Data Unit
7	Application	HTTP, FTP, SMTP	Data
6	Presentation	SSL/TLS, MIME	Data
5	Session	SOCKS, RPC, PAP	Data
4	Transport	TCP, UDP	Segment
3	Network	IPv4, IPv6, IPsec, ICMP	Packet/Datagram
2	Data Link		Bit/Frame
1	Physical		Bit

Tabelle 2: IP Protocol Stack

Nr	Layer	Data Unit
4	Application	HTTP, FTP, DNS
3	Transport	TCP, UDP
2	Internet	IPv4, IPv6, Routing
1	Physical Network	Ethernet, ADSL, WLan

Tabelle 3: Security at the different Layers

Layer	Advantages/Disadvantages
Link	• speed • seamless[1] security to layers above • every link must be secured separately • requires trust in link operator
Internet	• seamless security to layers above • IPSec is integrated in IPv6, also available for IPv4 • complex configuration on multi-user machines • tunnel mode encrypts only part of a route • depends on policy settings (but also a strength)
Transport	• can be added to existing applications to communicate securely • more portable and easier to configure than internet layer security • protocol specific • application must be TLS/SSL aware
Application	• extend application without involving operating system • applications understand data and can provide appropriate security • security bas to be designed separately for each application

2 TCP/UDP

images/tcp.png

images/udp.png

3 SSL/TLS Key Exchange

RSA Key Exchange (case 1) : client creates pre-master secret for session and sends it encrypted with server's public key to server. Secure against active/passive attacks, no PFS, no contributory key agreement.

RSA Key Exchange (case2) : server has RSA key that can only sign. Create temporary public/private keypair that is used to exchange a client generated key. Secure against active/passive attacks, PFS, no contributory key agreement.

Anonymous Diffie-Hellman uses Diffie-Hellman, but without authentication. Because the keys used in the exchange are not authenticated, the protocol is susceptible to Man-in-the-Middle attacks. Provides Perfect Forward Secrecy (PFS). Not secure against passive attacks. Contributory key agreement.

Fixed Diffie-Hellman embeds the server's public Diffie-Hellman parameter in the certificate, and the CA then signs the certificate. That is, the certificate contains the Diffie-Hellman public-key parameters, and those parameters never change. Secure against passive/active attacks, no PFS, contributory key agreement.

Ephemeral Diffie-Hellman uses temporary, public keys. Each instance or run of the protocol uses a different public key. The authenticity of the server's temporary key can be verified by checking the signature on the key. Secure against passive/active attacks, PFS, contributory key agreement.

images/tls.png

Phase 1 - Establish security capabilities: After phase 1 both parties know which key exchange mechanism and which cipher to use.

Phase 2 - Server authentication and key exchange: After phase 2 the client has all required values to generate the **session key**.

Phase 3 - Client authentication and key exchange: After phase 3 the client and server have authenticated each other and share **master secrete**.

Phase 4 - Finish: After phase 4 the cipher suite is changed and the connection is established. Note that in XI. $C \to S$ a handshake message containing all messages up to now is sent to the server.

4 Attack Classification Steve Kent

images/attack_classification.png

5 Reducing Risk

Avoid : Do not implement a feature and by that remove all risks that would be introduced by it.

Decrease : Add implementations/hardware/etc. to reduce a risk (i.e. implement two factor authentication instead of one factor).

Insure : Get an insurance against some of the risks.

Accept : Accept a risk.

Opportunity should be larger than the finally accepted risk.

6 Vulnerability Lifecycle

images/vuln_lifecycle.png

7 Botnet

Bot Agent : crime-ware tool installed on victims

Botnet : collection of all bot agents

Bot Master : the ciminal(s) operating the botnet

Command & Control (CnC) : botnet management system

7.1 Targets

- Buildup: efficient infection and spreading
- Persistence: prevent detection & removal
- Modularity: address changing functionality needs
- Scalability: handle large number of machines
- Adaptive: to business and technology challenges
- Anonymity: prevent identification of operator

7.2 Life Cycle

1. Computer is infected, becomes a bot agent
2. New bot agent connects to CnC, joins the botnet
3. Retrieve anti AV module
4. Secure the new bot agent

5. Listen to CnC, wait for command
6. Retrieve new payload module
7. Execute payload
8. Report result to CnC channel
9. Listen again to CnC

The command may result in the erase of all evidence and abandon client. While listening to CnC and executing the commands it's in the state "Exploitation". Before it's "Initialization" and the last one, that results in the erase of all evidence is "Termination".

7.3 Locating CnC

- fixed IP: easy to identify bot agent, easy to identify CnC, no flexibility, easy to block botnet
- fixed Domain: easy to identify bot agent, harder to identify controller, harder to shut down botnet (DNS caches), more flexibility (fluxing)
- IP Flux/Domain Flux: domain name/IP addresses change frequently, very dynamic, moving target, hard to shut down botnet
- IP Flux: constantly change IP address for a fully-qualified domain name (FQDN)
- Domain Flux: constantly change and allocate multiple FQDN
- Fast-Flux: every 3 minutes map a given FQDN to a new set of IP addresses, bot agent connects to same FQDN but this results in a connection to another CnC every 3 minutes
- Single Flux: FQDN of CnC's host has multiple IP addresses assigned (DNS A record)
- Double Flux: name servers of the CnC's FQDN as well as the FQDN of the CnC's host have multiple IP addresses assigned (DNS A and NS records)
- Single Flux/Double Flux: uses DNS protocol features: round-robin allocation of IP addresses, very short TTL values

7.3.1 Single Flux

`images/single_flux.png`

1. Clients A, B resolve 'cnc.botnet.com', both get same IP of the name server authoritative for 'botnet.com'
2. Clients A, B query '10.1.1.1' at different times and get different IPs for 'cnc.botnet.com'

7.3.2 Double Flux

`images/double-flux.png`

1. Client A, B resolve 'cnc.botnet.com', both get multiple different IPs of name server authoritative for 'botnet.com'
2. Clients A, B each query one of the NS, each NS returns multiple different IPs for 'cnc.botnet.com'

7.3.3 Domain Flux

- each bot uses a Domain Generator Algorithm (DGA) to periodically compute a list of new domain names, this is computed independently by each bot and is regenerated periodically
- bot attempts to contact the generated host, until one succeeds

7.4 CnC Topology

- Star
  - (pro) speed of control: commands and data can be transferred rapidly
  - (con) single point of failure
  - (con) bad scalability: hard to control large amount of bot agents with central CnC server
- Multi-Server
  - (pro) no single point of failure
  - (pro) geographical optimization
  - (con) advanced planning required
  - (con) coordination of servers needed (P2P-like protocol between servers)
- Hierarchical
  - (pro) no single bot agent aware of the location of the entire botnet
  - (pro) ease of resale: easily to sell part of the tree (sell subtree)
  - (con) command latency: commands must trace sever branches
- Random
  - (pro) highly resilient
  - (con) command latency
  - (con) botnet enumeration: passive monitoring of one bot agent allows enumeration of other members

8 Malware

8.1 Detection Evasion Tactics

- Create unique samples of malware at massive scale
- Protect malware from analysis
- Detect sandboxing technologies
- Test detectability before deployment
- Serial Variants: process of automatically churning out new variants of malware on massive scale
- Tactics: multiple variants of a particular malware agent are created in advance of the attack
- Process: off-the-shelf commercial software protection tools and specialized anti-antivirus manipulation tools have standardized process
- Polymorphism Techniques: manipulate the structure of the source code (of malware) by reordering and replacing common programmatic routines with equivalent ones.
- Compiler Setting Modulation: change compiler settings to get different binaries with the same semantics

8.2 Creation Steps

Original Malware : create core malicious functionality, buy or hire a coder
Crypter : encrypt malware against signature detection systems and static analysis
Protector : add anti-debugging features to malware that prevents analysis of it
Packers : make a binary file and installation kits smaller and more portable
Binders : used to embed a Trojan or other software packages
Quality Assurance : Malware created is tested against AV software to verify its quality

9 Authentication

Things you know : passwords, passphrases
Things you have : credit card, security token (RSA SecurID)
Things you are : fingerprint, voiceprint, retinal scans

Weak authentication uses only one of the above authentication criteria. **Strong authentication** uses two or more of them.

9.1 OpenID

- open standard for decentralized user authentication
- use existing account on multiple websites
- only one account with one password enough
- password given and known only to *identity provider*, which confirms identity to website

images/openid.png

9.2 OAuth

- privileges granted without sharing username and password
- user authenticates with service and grants access to another application he/she trusts

images/oauth.png

9.3 IEEE 802.1x

- client-server protocol to restrict devices from connecting to a network through publicly accessible ports
- data link layer protocol used for transporting high-level authentication protocol
- 802.1x EAP: authentication framework which supports multiple authentication methods
- limitations: MitM attacks, session hijacking

images/IEEE_802_1x.png

- Un-Controlled channel for EAPOL (Extensible Authentication Protocol over LAN) only
- Controlled channel open after authorization.

10 IPTables

- INPUT: All packets destined for the host computer
- OUTPUT: All packets originating from the host computer
- FORWARD: All packets neither destined for nor originating from the host computer, but passing through (routed by) the host computer. Used if you are using host as a router.
- ACCEPT: accept the packet
- DROP: drop the packet on the floor
- QUEUE: hand the packet off to a user-space process (rarely used)
- RETURN: stop processing in this chain and resume in the previous chain (rarely used)
- MASQUERADE: only in nat table: rewrite source or destination address with address of outgoing or incoming interface

```
# drop anything for INPUT chain as default (--policy
    same as -P)
iptables --policy INPUT DROP

# accept TCP packets with SYN flag to port 80 from
    any IP (-s) to any IP (-d)
iptables -A INPUT -p tcp -s 0/0 -d 0/0 --destination-
    port 80 --syn -j ACCEPT

# add to INPUT table: incoming packages on eth0 (-i)
    use "state" module, allow only packets belonging
    to or related to established connections
iptables -A INPUT -i eth0 -m state --state
    ESTABLISHED,RELATED -j ACCEPT
```

- `-s`: source (IP address, network address (IP address + mask), hostname)
- `-d`: destination (IP address, network address, hostname)
- `-j`: "jump to target": ACCEPT, DROP, QUEUE, RETURN
- `-i`: input interface name
- `-o`: output interface name
- `-p`: protocol: tcp, udp, icmp
- `--source-port`: number or name that is defined in `/etc/services`
- `--destination-port`: number or name as in `--source-port`
- `--tcp-flags`: ALL, NONE, SYN, ACK, FIN, RST, URG, PSH

images/firewall0.png

images/firewall1.png

11 UDP Hole Punching

images/hole_punching.png

12 IDS/IPS

- *Object* of observation
  - **packet** - analysis of packet headers and content
  - **flow** - analysis of flow parameters (IP address, port, # of packets, # of bytes, timing parameters, ...)

- *Point* of observation
  - **host** - by software running on the host, or device monitoring the host
  - **network** - by data collectors attached at strategic places in the network
- *Method* of observation
  - **signature** - comparison of observed events against database containing signature of malicious events
  - **behavior** - detection of deviation from normal state; requires knowledge of ground truth

A **flow** is a sequence of packets with the same 5-Tuple: (src IP, src port,dst IP, dst port, protocol)

Network based IDS
- easy to deploy
- one sensor can monitor multiple machines
- no context information, thus more false positives
- protects unknown machines
- performance issues

Host based IDS
- harder to deploy, because additional software on host required
- one sensor per monitored machine
- context information, thus less false positives

Signature based
- precision due to database of signatures of malicious events, thus less false positives
- not capable of detection unknown attacks, thus more false negatives
- automatic generation of signatures? still a research topic

Behavior based
- detection of unknown attacks via anomaly detection
- difficulty of establishing ground truth, what is “normal”?
- more false positives, potentially less false negatives
- area of research

There exists several possibilities to attack and IDS. One of them is **flooding / resource exhaustion**, where resources are depleted until the IDS drops packets or crashes. Another is attacking the **algorithmic complexity** of an IDS by launching a dos. This enables other attacks to remain undetected.

13 SSH

- Secure command shell
- Secure file transfer
- Data tunneling for TCP/IP applications
- Full negotiation of encryption, integrity, key exchange, compression and public key algorithm and format
- Runs on top of TCP, e.g. TCP SYN flood, TCP RST, bogus ICMP, TCP desynchronization
- Possible to do traffic analysis (amount of data, source, destination, timing, ...)
- Server authentication on first connection done by user (pinning server's key hash basically)

13.1 Protocols

SSH-CONN ection protocol for connection protocol

SSH-AUTH enication protocol for client authentication
- Authentication request, including username, method name and method-specific data in regard to authentication method to use, service name for access request
- AUTH method “None”: returns list of names of available authentication methods
- AUTH method “Public-key”: can be used to check if given user would be authorized to access account or to perform the authentication with key of user
- AUTH method “Password”: performs password test, method used for specifics can be set (PAM, passwd, LDAP, Kerberos, ...)
- User authentication: server delegates responsibility for user authentication to client host
- Host authentication: server verifies identity of client host, not user. Used for batch jobs. Better than plain text password in scripts

SSH-TRANS port protocol for server authentication, confidentiality, integrity
- Algorithm negotiation
- Session key exchange
- Session ID
- Server authentication
- Encryption, integrity, data compression

14 Block Cipher Modes

14.1 Electronic Codebook (ECB)

images/ecb.png

- (pro) simple to compute
- (con) traffic analysis yields which ciphertext blocks are equal, i.e. we know which plaintext blocks are equal
- (con) adversary may be able to guess part of plaintext: can decrypt parts of message if same ciphertext block occurs
- (con) adversary can replace blocks with other blocks

14.2 Cipher Block Chaining (CBC)

images/cbc.png

- (pro) semantic security
- (pro/con) altered ciphertext only influences two blocks

14.3 Cipher Feedback (CFB)

images/cfb.png

- (pro) semantic security
- (pro) altered ciphertext influences all following blocks
- (con) same vulnerabilities as any stream cipher

14.4 Output Feedback (OFB)

images/ofb.png

- (pro) semantic security
- (con) altered ciphertext only influences single block
- (con) same vulnerabilities as any stream cipher

14.5 Counter Mode (CTR)

images/ctr.png

- (pro) semantic security
- (con) altered ciphertext only influences single block
- (con) same vulnerabilities as any stream cipher

15 Cryptographic Hash Functions

- One-way: Given $y = H(x)$, cannot find x' such that $H(x') = y$
- Weak collision resistance: Given x, cannot find $x' \neq x$ sucht that $H(x) = H(x')$
- Strong collision resistance: Cannot find $x \neq x'$ such that $H(x) = H(x')$

16 IPSec

images/ipsec.png

17 Disclosure Debate

- Full Disclosure (FD)
  - vendor has strong incentive to release a fix
  - all affected parties get the same information
- Bug Secrecy
  - if kept secret, nothing will happen
  - likely to be found by other party that won't keep it secret

17.1 Incentives

- Economics: vulnerability information has commercial value
- Ethics: minimizing risk for society?
- Resources: has the discoverer the resources (e.g. time) to manage the discovery, will it be compensated?
- Past Experience: e.g. how did the vendor react in the past?
- Publicity: is the discoverer interested in publicity
- Legal constraints: is discoverer bound by legal constraints in how to handle the vulnerability (law, employer, ...)

17.2 Coordinated Disclosure

1. Vulnerability discovered
2. Notify vendor: alert vendor in private, give vendor reasonable time to investigate issue
3. Collaborate with vendor: technical details discussed, PoC, etc. Vendor develops patch
4. Coordinated Disclosure: Path published at agreed date, discoverer credited for finding, discoverer published finding and cites patch

If any of the steps fails from the viewpoint of the discoverer, do full discloser.

18 Security Ecosystem

- White Market: vulnerability bought and forwarded to vendor, followed by coordinated disclosure process.
- Black Market: vulnerability bought and exploited by cybercriminals and governments, no vendor notification
- Software vendors tend to be biased, not report full details and be reactive in regard to security issues and assess the risk
- Security community (MLs, blogs, tweets, etc.) is not always validated/rumoured, not a trusted source, inconsistent format, too technical for risk decisions
- Security Information Provider (SIP) Organizations *monitoring* the primary sources of security information, *validating* the content found and *publish* their findings as security advisories in a *consistent format.*

19 Email Authentication

19.1 DomainKeys Identified Mail (DKIM)

- Sender's mail server signs a hash of the email message (some header fields and content) with sender's private key
- Receiver's mail server gets sender's public key from DNS text record and verifies signature
- Guarantees message integrity, sender authenticity
- Domain-level digital signature authentication framework
- Allows organizations to take responsibility for a message
- Owner of the private/public key pair asserts validity (using DNS text record)
- Compared to S/MIME / OpenPGP: email content is not encrypted in DKIM, DKIM tries to prevent phishing

19.2 Sender Policy Framework (SPF)

- Give domain owners way to declare which SMTP machines are legitimate for their domain
- Mitigates misdirected bounces (from forged emails), most emails with forged senders are spam
- Sender's domain owner publishes which machine are authorized to use their domain in the `SMTP HELO` and `MAIL FROM`
- All SMTP hosts must be listed from which a valid mail can be sent

20 Email Spam

- Penny Stocks Spam
  - Criminal buys penny stock
  - Criminal spams about "hot" stocks
  - Recipients buy stocks from criminal, who controls the liquidity and pricing
  - Criminal profits
- Advance Fee Fraud (AFF)/Nigerian Money Scam
  - Promise of large money sums, that must be correctly transferred
  - Ask for personal details: postal address, phone/fax number, bank account details
  - After answering, a never-ending series of banking fees/lawyer expenses must be paid to initiate money transfer (which never happens)
- Work from Home / Mule Recruitment Spam
  - Receive payments on own bank account (money from illegal activities of course)
  - Keep 10
  - Get cash from bank account
  - Transfer on by Western Union (money laundering mule)
- Swiss Law
  - Sending spam (email, SMS, etc.) is illegal if the message is sent from a sender based in Switzerland
  - Revised FMG (Fernmelde-Gesetz) and UWG (Unlauteren Wettbewerb)
- US Federal Can Spam Act
  - commercial email must be identified as advertisement, and include sender's valid physical postal address
  - email must give recipients an opt-out method
  - prohibits deceptive subject lines
  - bans false or misleading header information
- European Union E-Commerce Guideline
- Austria: UWG (Unlauteren Wettbewerb), TKG (Telekommunikationsgesetz)

20.1 Anti-Spam Techniques

- Whitelisting: allow emails matching non spam criteria, e.g. sender
- Greylisting: defer initial email from same identification triple (IP sender address, email sender, recipient address), ut accept follow-up emails
- Blacklisting: reject emails matching spam criteria, e.g. realtime blacklists (RBL)
- Heuristical Content Filtering
  - look at probable spam features in email content: weird use of fonts, tiled images, strange URLs, ...
  - combine scores of all algorithms into overall score
  - mark as spam if threshold reached
- Statistical Content Filtering
  - $P(spam|words) = P(words|spam) \cdot P(spam)/P(words)$
  - Probability an email containing certain words is spam is equal to probability of those words showing up in any known spam email, times the probability that any email is spam, divided by probability of finding those words in any email
- Distributed Checksum Clearinghouse (DCC): counts number of same email messages seen in the past. High counts are indication for mass mailings. Large email hosters can calculate easily this number
- File spam filtering
  - use optical character recognition (OCR) to images, PDFs, etc. to extract text and filter the text
  - use signature-based filters (hash the files)
- Text filtering
  - normalize content before filtering: HTML comments, HTML table alignments, white text on white background, tiny letters in between actual text

- misspellings in text (defeats Naïve Bayesian filter)
- URL with redirects to another URL (defeats blacklists)

20.2 Filtering at SMTP Time

- deny while the email is handed over to the SMTP server
- deny after learning client's (sender's) IP address
- deny at HELO/EHLO
- deny sender (at MAIL FROM)
- deny recipient (at RCPT TO)
- deny begin/end of data (at DATA)
- (pro) cheap, fast, no long email body received or processed
- (con) not too much information available to decide if its really spam

20.3 DNS Blacklists

- query an IPv4 address against a DNS Blackist: check if `<ip-address>.<dns-blacklist-domain>` returns an A record, then it's blacklisted
- the same is possible with domains: `<domain>.<dns-blacklist-domain>`

20.4 Realtime Blacklists

Differences between realtime blacklists:

- Operator
- Costs
- Goal
  - hosts have done things proper SMTP servers don't do
  - insecure CGI scripts allowing open relaying
  - open proxy servers / single hop relays
  - IP addresses sent spam to trap
  - URLs found in spam emails
  - illegal third-party exploits
  - Domains confirmed by owner to not send emails
- Nomination (how to get on the list)
  - manual by admin of blacklist provider
  - users submit
  - spam trap
  - request by host
  - tested by trusted testers
  - ... often not disclosed
- Listing lifetime (how to get off the list)
  - 5s - 20min, usually under a month
  - until de-listing request
  - temporary until spam stops

21 DNS Security

A **zone** (domain) is a collection of hostnames/IP pairs all manage together.

A **name server** is a server that answers dns queries.

An **authoritative name server** is a name server that knows the answer directly, since he is responsible for this zone. Authoritative name server could be controlled by a private entity.

A **resolver** is the client part of the dns and resolves the domain names.

A **recursive name server** is a name server and a resolver at once. Those types of name server finds the results for zones it's not authoritative for.

A **stub resolver** is a small library, that forwards request to a recursive name server and is typically used by end-hosts.

DNSSEC uses public-key cryptography to provide *authenticity*, *integrity* and *backward compatibility*, but it does not provide *confidentiality* and *protection against DoS*.

22 XSS

Same origin policy defines access rights: A script can only access content and properties of a document loaded from the same origin as the document containing the script. Same origin means: protocol, hostname and port are the same, but only considering the source URL and ignoring the path. It's possible to have different origin on one page, for example by using iframes.

A **Cross Site Scripting (XSS)** is an attack where an attacker uses xss vulnerability to inject code into a web page, and executes it in the victim's browser under the web site's domain. This way he might be able to hijack user session, deface web sites, introduce worms,...

Cross Site Request Forgery (XSRF) is an attack that, where e form from one domain post a request to a different domain through an authenticated session. A malicious website could use a hidden iframe to automatically sends requests to a web service, where a user is currently logged in. E.g., reset the password. Such attacks are also known as blind **write-only attacks**, since the attacker is not able to read the users Cookie, but he could trigger the victim's browser to send it to the target web service.

Cross Site Script Inclusion (XSSI) may occur, when a website A includes a script from another one B. The loaded script is evaluated in the context of origin of A and hence allows it to do some harm.

22.1 Susceptible infected susceptible Model (SIS)

...is a biological Virus Propagation Model for a *homogeneous network* with N nodes. The three stages of worm spreding are *pre-outbreak, free spreading, clean-up*.

- $\rho(t)$ - fraction of infected nodes $\in [0,1]$
- β - infection rate of a node along each edge $\in [0,1]$
- δ - cure rate of each node $\in [0,1]$
- k - numer of outgoing edges at each node $(\approx const), k < N$

$$\frac{d\rho(t)}{dt} = \beta * k * (1 - \rho(t)) * \rho(t) - \delta * \rho(t)$$

- **Stationary Condition**, where $0 = d\rho(t)/dt$

$$\rho(t) = 1 - \frac{\delta}{\beta * k} \; or \; \rho(t) = 0$$

- **Epidemic threshold**, $\rho(t),\ t \to \infty,\ \lambda = \frac{\delta}{\beta * k}$

23 Identity and Authentication

An **identity** specifies a principal (a unique entity). e.g. individuals (person name), physical objects (computer, router, smart card, ...), logical objects (software, internet location, ...), groups of principals.

Identity theft is a crime in which impostors obtain key pieces of PII and use them for their own personal gain or to do harm.

Personally Identifying Information (PII)s are for example a social security numbers, i.e. birth date and driver's license number. *Variants: Financial-, Criminal-, Business/Commercial Identity Theft & Identity Cloning*

Authentication is the process of verifying an identity claim of an entity. It binds the principal to an identity.**Weak authentication** means checking only one authentication criteria, while **strong authentication** means checking two or more.

24 Attacks and Defence

Tabelle 4: Attacks and its Defence

Service	Attack/Defense
Anonymity, e.g., Tor, Mixnet	Traceback - *add more proxise/nodes* Collusion - *use a reputation system* Traffic analysis - *heartbeats, traffic shaping, padding,...* Logging - *reset path periodically*
Availability	SYN flooding - *SYN Cookies* Compression bomb - *restrict decompression size, limit depth* Smurf attack - mail bounce amplification -
DNS	(D)DoS - *redundancy, over provisioning* web interface - *Update firmware, use strong passwords* local host configuration, e.g., /etc/host - DNS spoofing - *bailiwick checking, ignore any records that aren't in the same domain of the query* Fast response - *source port randomization in combination with random TXID* Cache poisoning - *"* Kaminsky attack - *"*
Email	spam - *White-, Grey-, Blacklisting, Heuristical Content Filtering, Statistical Content filtering, Distributed checksum clearinghouse (DCC), File Spam-, Text filtering*
Firewall, IDS/IPS	flooding - *SYN Cookies* algorithmic complexity -
Session Mgmt	Network sniffing - *use HTTPS instead* brue force, potential DoS, lock all user accounts -
SSH	eavesdropping - *encryption in SSH-TRANS* name service & IP spoofing - *cryptographically verified server identity* connection hijacking - *cannot prevent but detected on tcp level* MitM attacks - *server authentication (first conn.)* password cracking - *rather a social problem* (D)DoS - *cannot counter* traffic analysis, possible to watch data amount, src/dest, timing - *"*
SSL	MitM attacks - *compare URL with certificate*
VPN	replay attack - *embed a unique ID / timestamp before signing*
Web Applications	SQL-injection - *sanitize all client data **on the server**, prepared statements, avoid disclosing DB error infos, run DB with reduced privileges* XSS - *Output encoding, Input validation* XSRF - *Input validation, Security Token* XSSI, direct sourcing - *DON'T* XSSI, call-back - *Security Token*