TSIT01 Datasäkerhetsmetoder

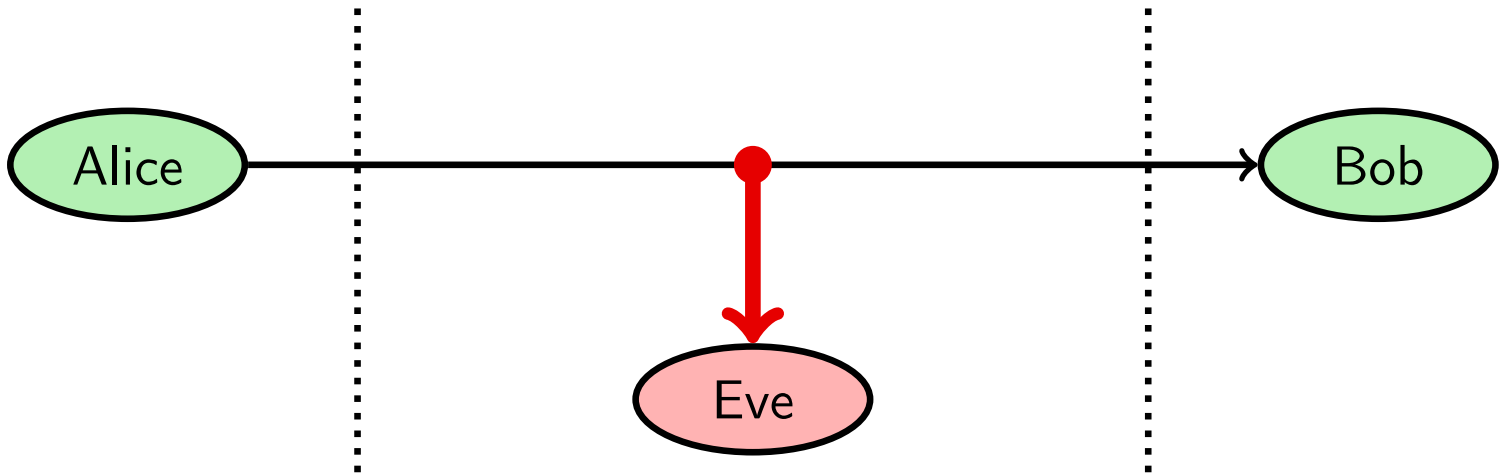
Föreläsning 8: Nätverkssäkerhet

Jonathan Jogenfors

Information Coding Group
Department of Electrical Engineering

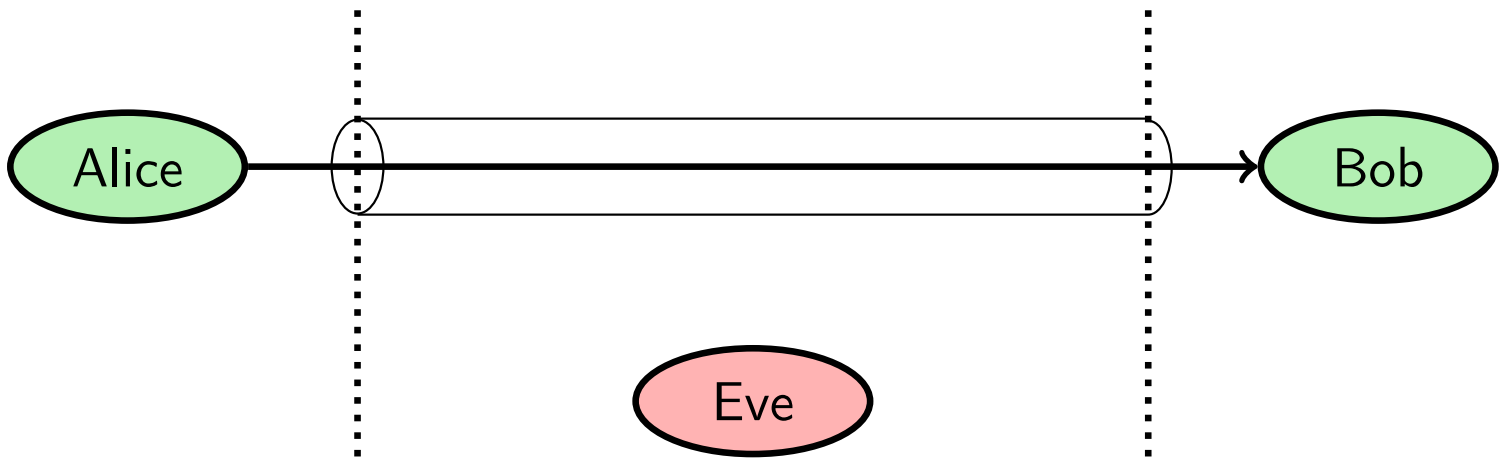
Communication and network security: Threat model

- Passive attacks: Eavesdropping, Wiretapping, Sniffing, and Traffic analysis



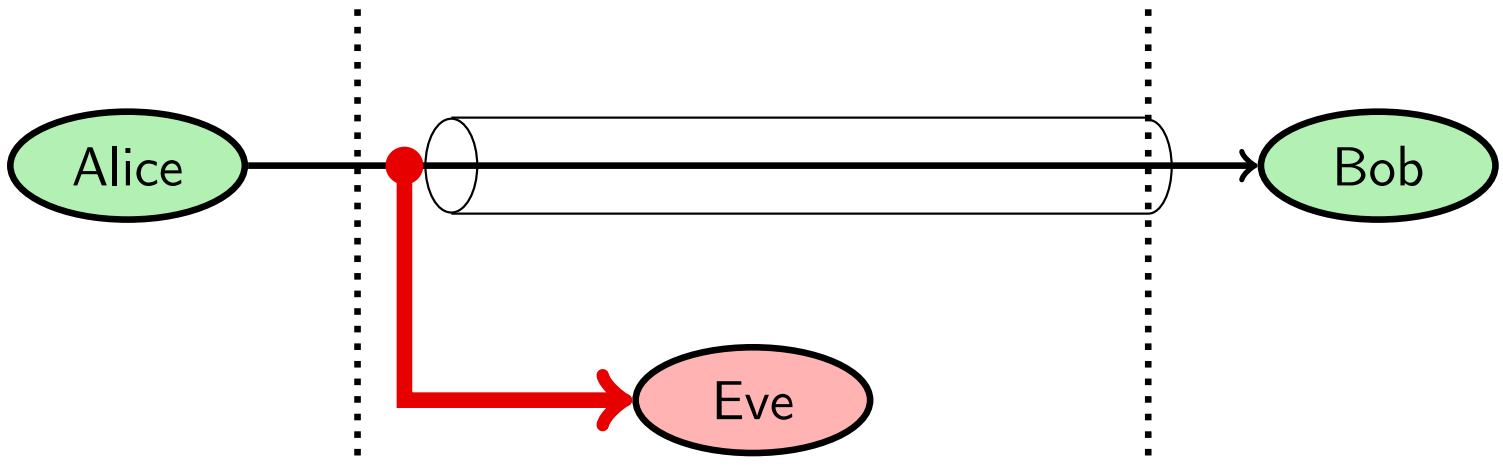
Communication security: Secure tunnels

- Typically provide Confidentiality, data Integrity, and data origin authentication
- End points may be machines or services on the local computer



Communication security: Secure tunnels

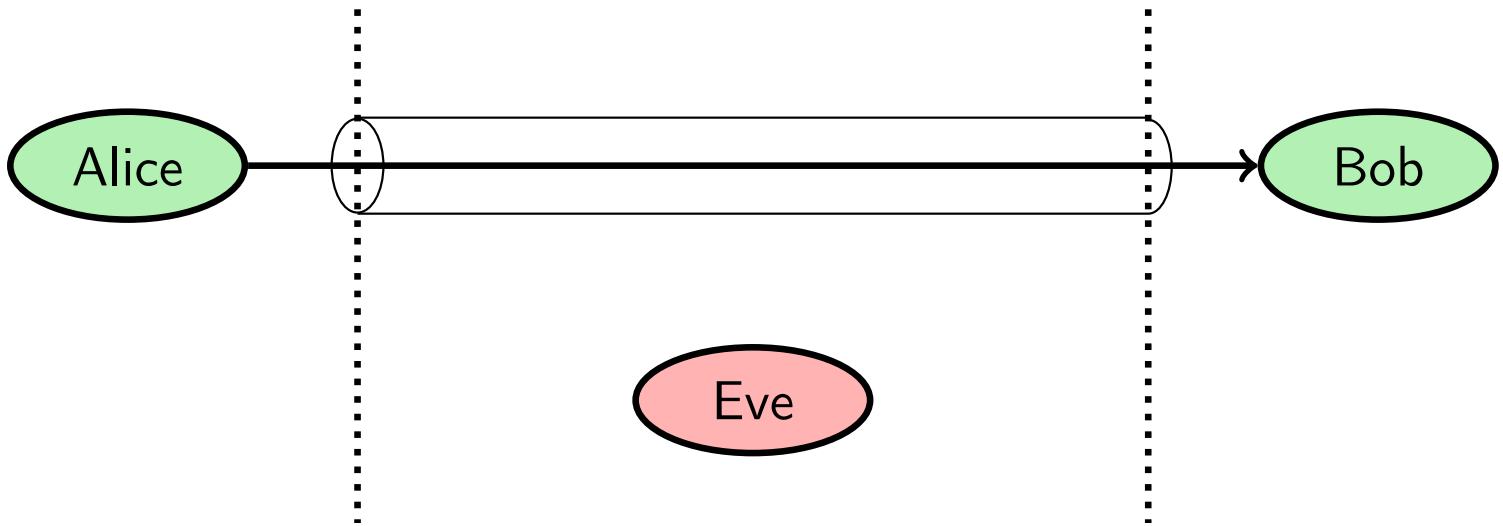
- Typically provide Confidentiality, data Integrity, and data origin authentication
- End points may be machines or services on the local computer
- The placement is important to achieve security



Communication security: Secure tunnels

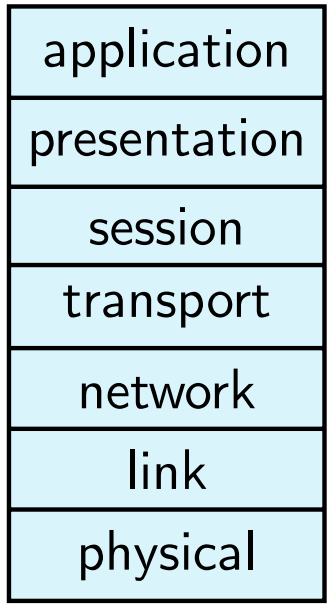
Steps to set up a tunnel

1. Authenticated key establishment ($\rightarrow$asymmetric key)
2. Key derivation ($\rightarrow$symmetric key)
3. Traffic protection through symmetric cryptography

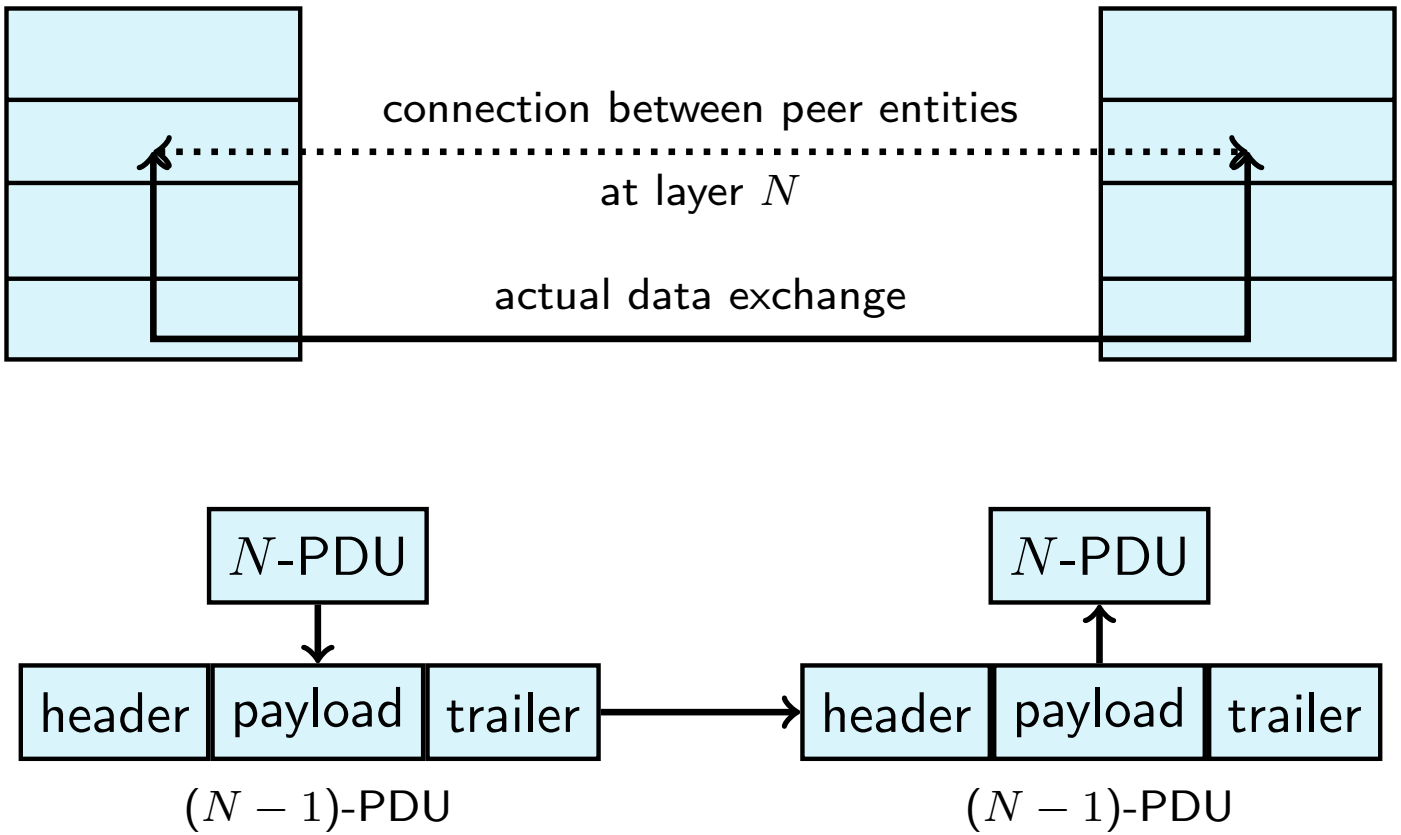


Layered model of network protocols

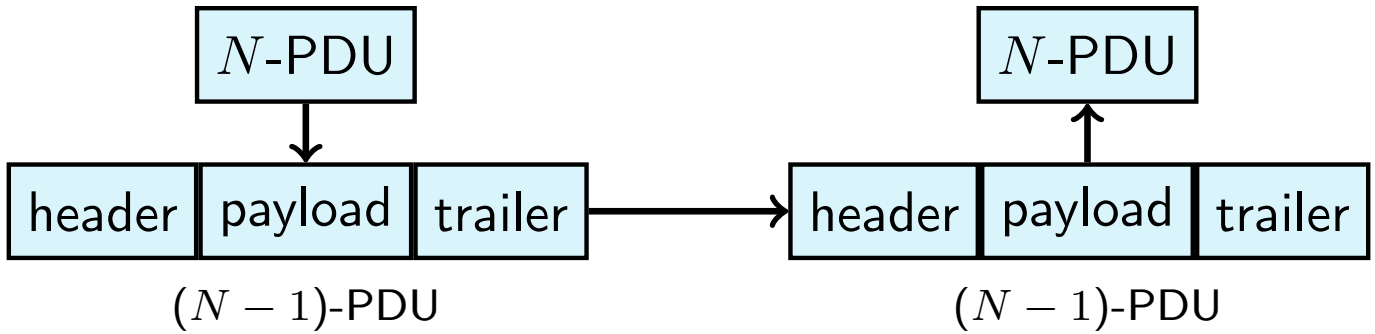
- *The* abstraction of network structure, the ISO/OSI seven-layer model
- Security services at the top can be tailored for specific applications, but each application then needs a separate service
- Security services at the bottom can protect the upper layers transparently, but may not meet all requirements of specific applications



Briefly on the layered model



Briefly on the layered model

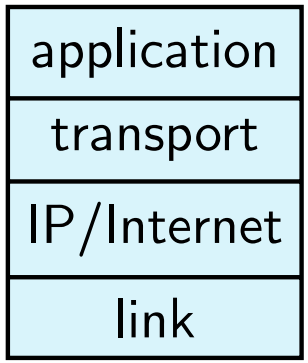


There are two options for security services in the $N-1$ layer

- The upper layer can be *aware* of the security services at the lower layer
- The lower layer security services can be *transparent*

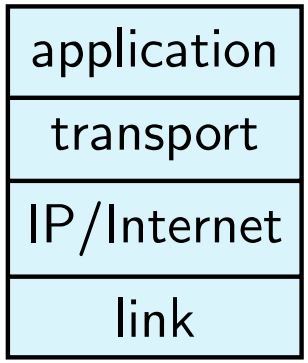
The internet protocol stack

- The application layer has Telnet, FTP, HTTP, SMTP, SET, ...
- The transport layer has the protocols TCP and UDP, and applications connect to *ports* in this layer
- The internet layer has the IP protocol, nodes are identified through their *IP address*
- The link (and the physical) layer are specific to the tech used
- There are security services both in the transport and Internet layers



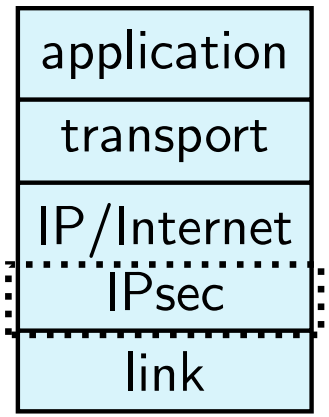
The IP protocol

- The IP protocol is stateless and does not keep track of connections
- Each packet is independent
- No guaranteed delivery
- Order is not preserved
- No security mechanism



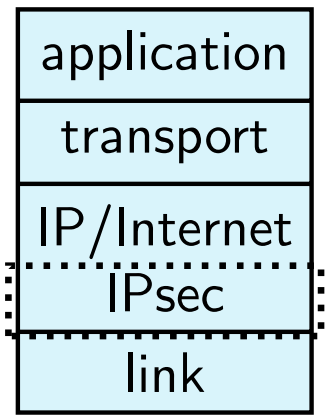
IPsec, IP protocol security

- Security architecture is in RFC 4301
- Optional for IPv4, mandatory for IPv6
- Two major security mechanisms: Authentication Header and Encapsulating Security Payload (ESP)
- Authentication Header does not give Confidentiality; it was used to avoid export restrictions in the 90s

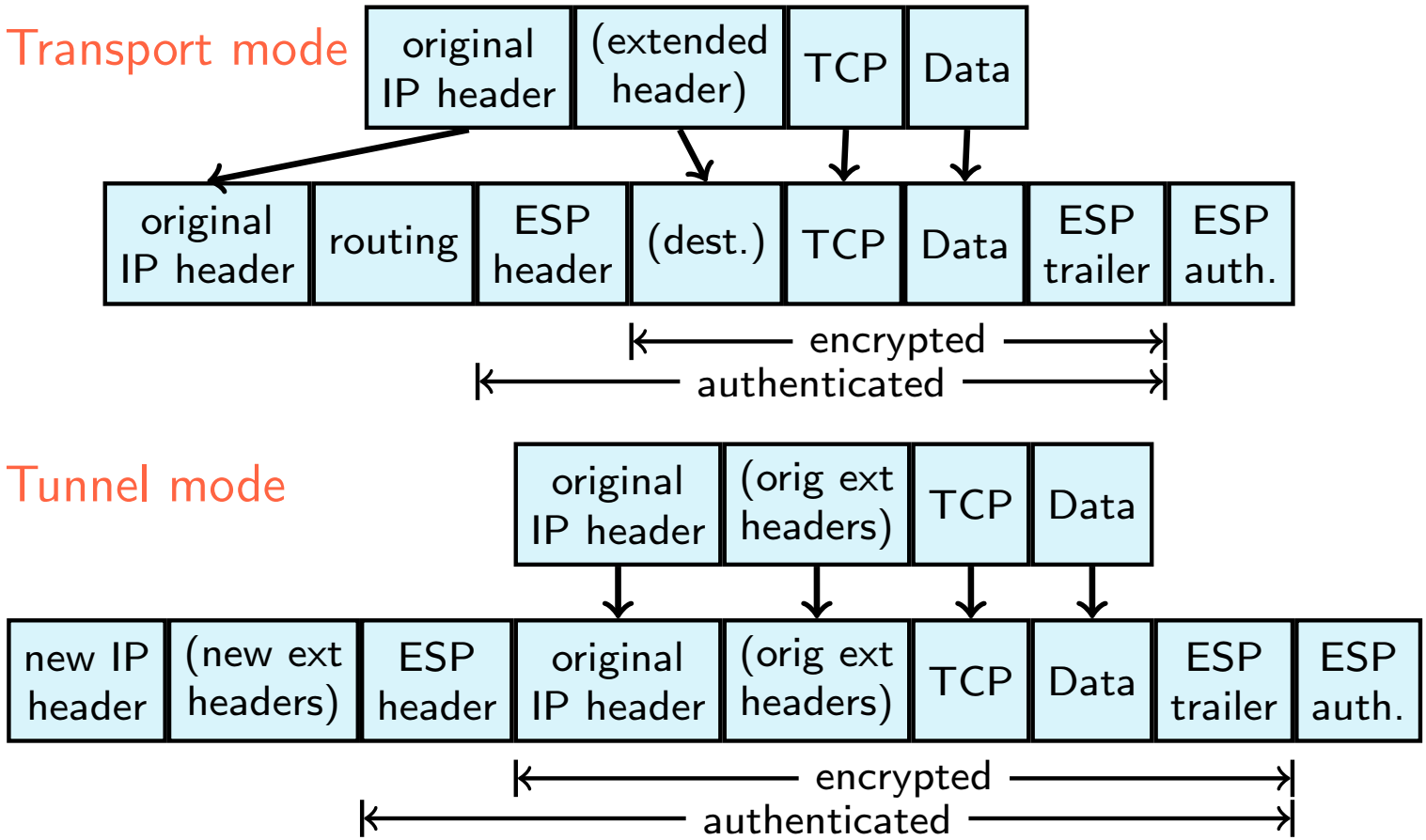


IPsec, Encapsulating Security Payloads

- ESP provides Confidentiality, data Integrity, data origin authentication, some replay protection, and limited traffic flow Confidentiality
- ESP can be run in two modes: Transport mode and Tunnel mode
- For transport mode, both nodes need to be IPsec-aware
- Tunnel mode, on the other hand, is transparent: IP-within-IPsec

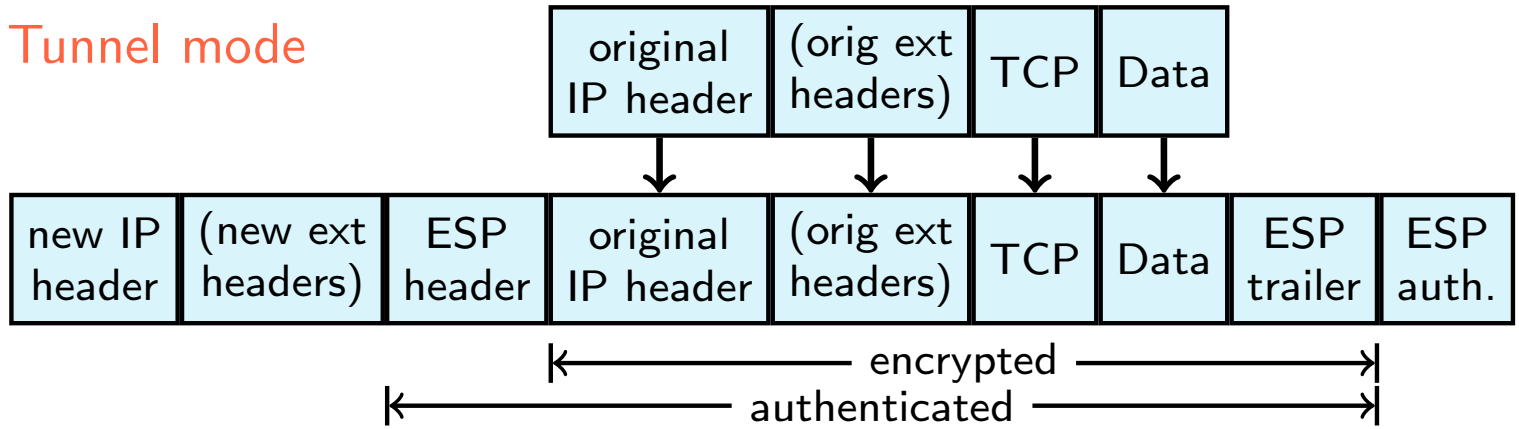


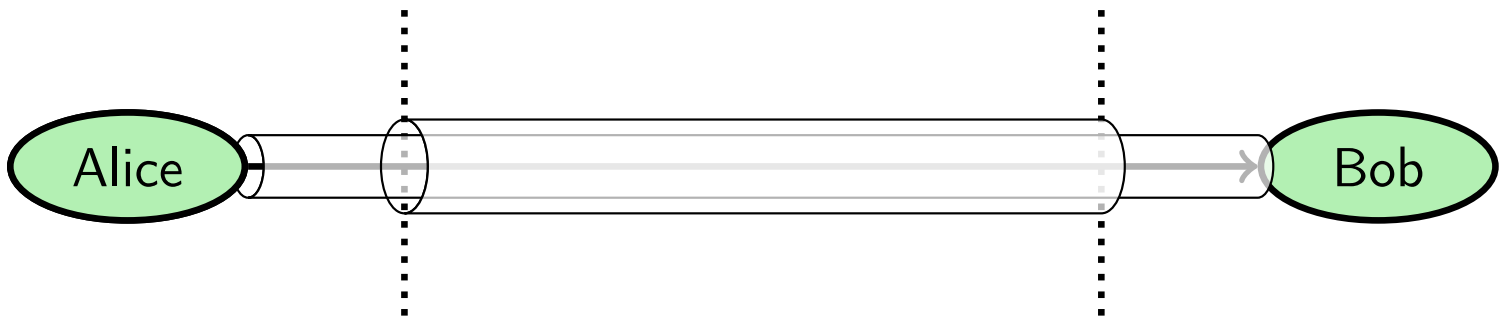
IPsec, Encapsulating Security Payloads



IPsec, Encapsulating Security Payloads

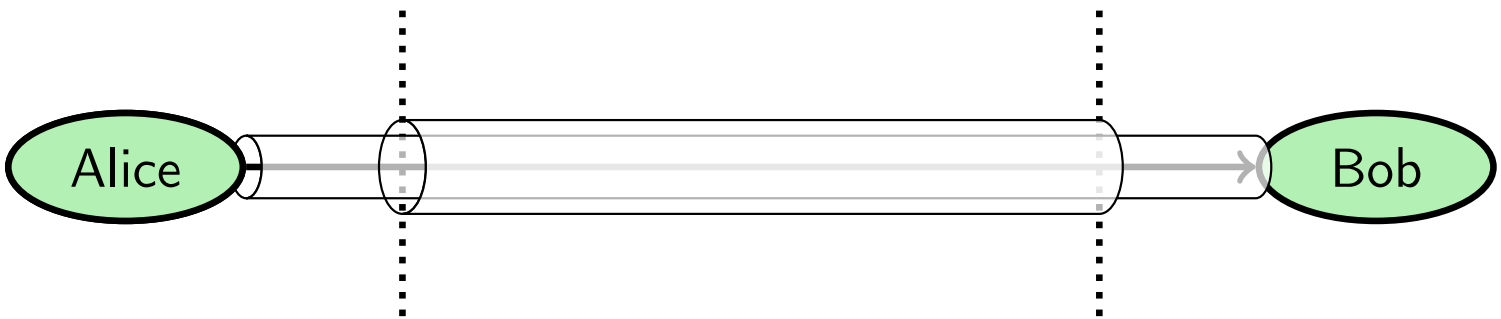
Tunnel mode





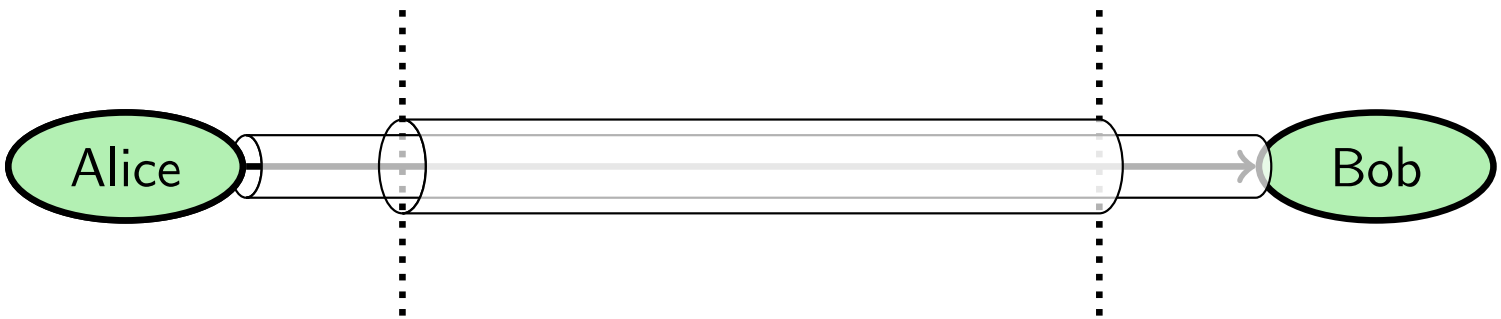
ESP headers, and Security Association

- The ESP header contains the ID of a Security Association (SA), and a sequence number
- The SA is a common state for communication from one node to another (so IPsec is not stateless)
- Usually created in pairs



ESP headers, and Security Association

- State includes source, destination, security protocol, cryptographic algorithms and keys, key lifetimes, IVs, sequence numbers and anti-replay windows
- SAs can be combined in multiple nesting levels

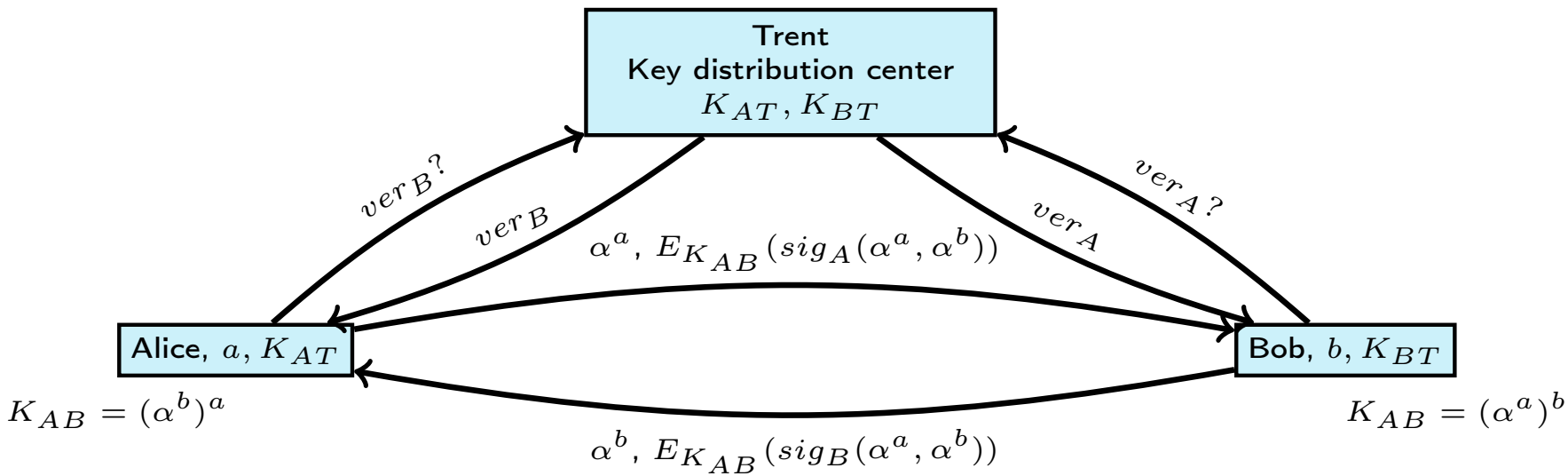


Establishing SAs through Internet Key Exchange (IKE, ISAKMP)

- Internet Key Exchange (v2) is in RFC 4306
- Uses two phases
- First phase:
  - establishes a shared session key through Diffie-Hellman key exchange
  - the result is an SA for IKE itself

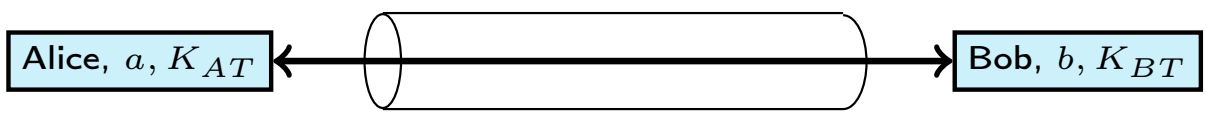
Establishing SAs through Internet Key Exchange (IKE, ISAKMP)

- First phase:
  - establishes a shared session key through Diffie-Hellman key exchange
  - the result is an SA for IKE itself
  - to achieve authentication, a pre-shared secret or trusted public keys are needed



Establishing SAs through Internet Key Exchange (IKE, ISAKMP)

- First phase:
  - establishes a shared session key through Diffie-Hellman key exchange
  - the result is an SA for IKE itself
  - to achieve authentication, a pre-shared secret or trusted public keys are needed
- Second phase establishes IPsec Security Association(s) in an encrypted session, that uses the key formed in the first phase



IPsec problems: DOS

- In the first phase, an attacker might send bogus responses
- The second phase would never complete
- The bogus keys will need storage

- Mitigation: use several responses
- Storing all responses for each connection will waste resources
- Solution: store them until negotiation completes
- Use the resulting SA

IPsec problems: DOS II

- Another DOS possibility is to flood a node with initiation requests from forged IPs
- This would make the node waste resources on calculating responses and storing session data

- Solution to this is to do no state storage first, but respond with a "cookie", expecting it to return from the initiator IP
- If the IP is bogus, the DOSer does not get the cookie, and cannot return it
- Cookie format is not standardized, the RFC suggests to hash IP, the request nonce, and a local secret that is changed regularly

IPsec and NAT

- Network Address Translation was invented to cope with the lack of available IPv4 addresses
- Nowadays it is sometimes seen as a security feature that internal IPs are not directly addressable
- This creates problems, especially for IKE since authentication is by IP address
- NAT-T ("IPsec passthrough") solves this by adding an extra header, triggering rewriting rules at the recipient after packet authenticity has been checked

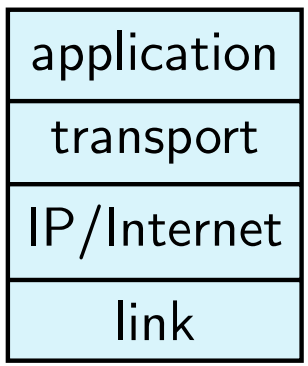
Pros and cons of IPsec

- IPsec provides security transparently
- Upper layers need not be aware that lower layers are more complicated to provide security

- Cannot be tuned for specific applications
- IPsec provides host-to-host (gateway-to-gateway) security, not user-to-user or application-to-application security
- IP is stateless and unreliable by construction, but IPsec is stateful
- IPsec packets need to be ordered, while IP should not be concerned with packet order or dropped packets

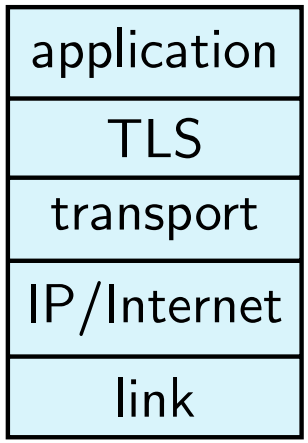
SSL/TLS

- Placed between "normal" TCP and application
- Handshake phase uses asymmetric encryption and certificates to exchange the session key
- The server (but not the client) is authenticated (by its certificate)
- Session key is for a symmetric algorithm
- Many different algorithms can be used, the set is not standardized



SSL/TLS

- Placed between "normal" TCP and application
- Handshake phase uses asymmetric encryption and certificates to exchange the session key
- The server (but not the client) is authenticated (by its certificate)
- Session key is for a symmetric algorithm
- Many different algorithms can be used, the set is not standardized

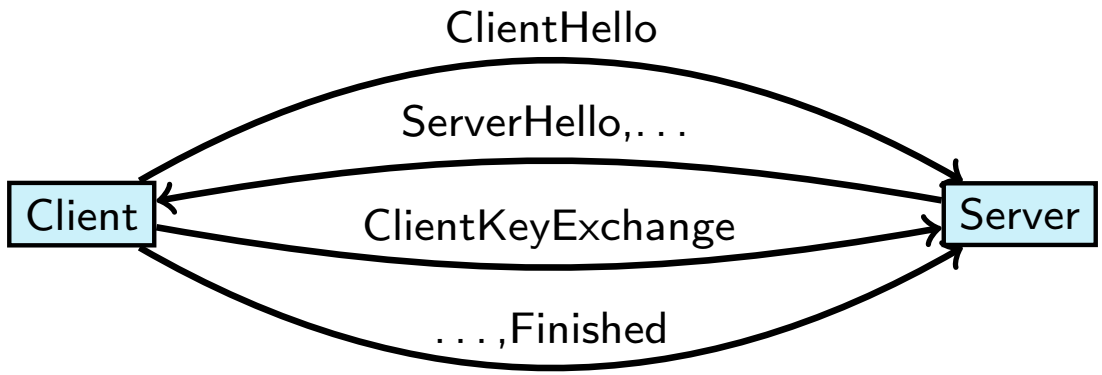


TLS Handshake Layer

ClientHello: highest TLS protocol version, random number, suggested public key systems + symmetric key systems + hash functions + compression algorithms

ServerHello, Certificate, ServerHelloDone: chosen protocol version, a (different) random number, system choices, public key

ClientKeyExchange: PreMasterSecret, encrypted with the server's public key

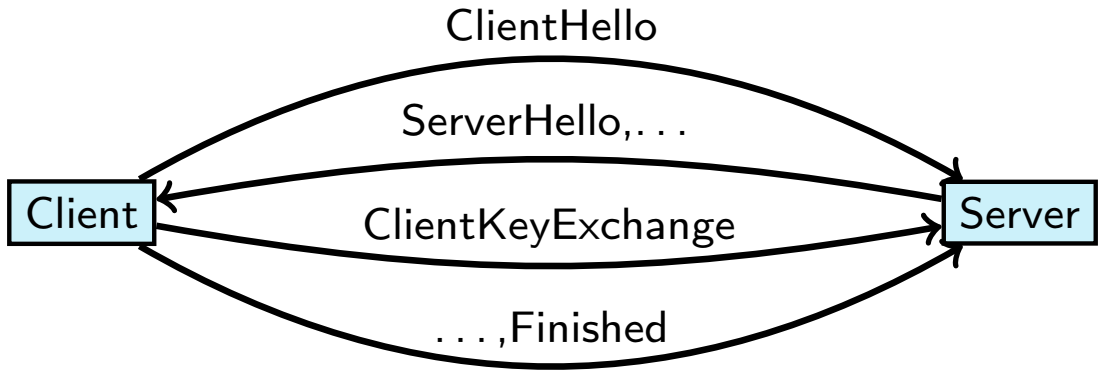


TLS Handshake Layer

(Master secret): creation of master secret using a pseudorandom function, with the PreMasterSecret as seed

(Session keys): session keys are created using the master secret, different keys for the two directions of communication

ChangeCipherSpec, Finished authenticated and encrypted, containing a MAC for the previous handshake messages



Pros and cons of TLS/SSL

- TLS provides user-to-user or application-to-application security
- Useful even in specific applications
- Sits above TCP, so can use benefits of stateful TCP packet handling

- Applications need to be security-aware
- They must explicitly ask for security
- Changes are not that large, use TLS connect instead of TCP connect
- The security state sits closer to the application/user, and may be more vulnerable

Current state of SSL/TLS

- 1994: SSLv1, never published.

Current state of SSL/TLS

- 1994: SSLv1, never published.

  > *The actual history of SSL was that SSL 1.0 was so bad that Alan Schiffman and myself broke it in ten minutes when Marc Andressen presented it at the MIT meeting.*

Current state of SSL/TLS

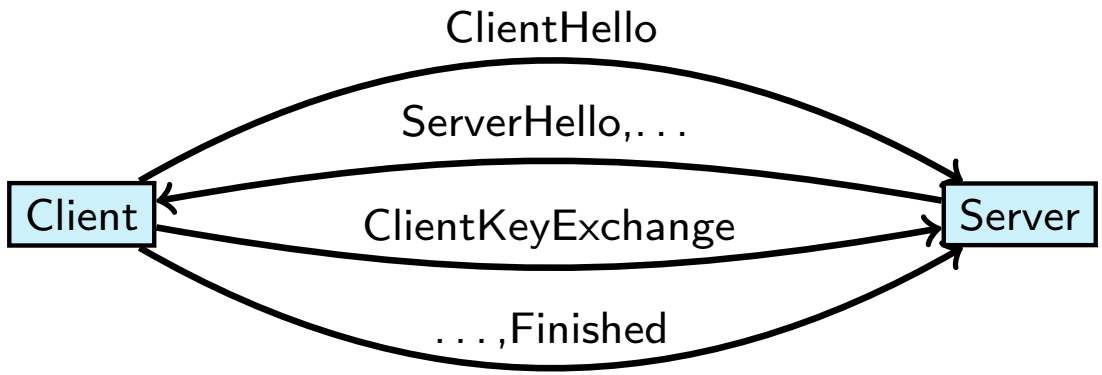
- 1994: SSLv1, never published.
- 1994: SSLv2: better but was found to contain many flaws.
- 1996: SSLv3: Complete protocol redesign
- 1999: TLS 1.0. Minor changes from SSLv3
- 2006: TLS 1.1: Protection against padding and CBC attacks
- 2008: TLS 1.2: Adding SHA256, and more
- 2015: TLS 1.3 (draft): Removing old ECC curves and MD5, and more

SSLv3 is now known to be insecure

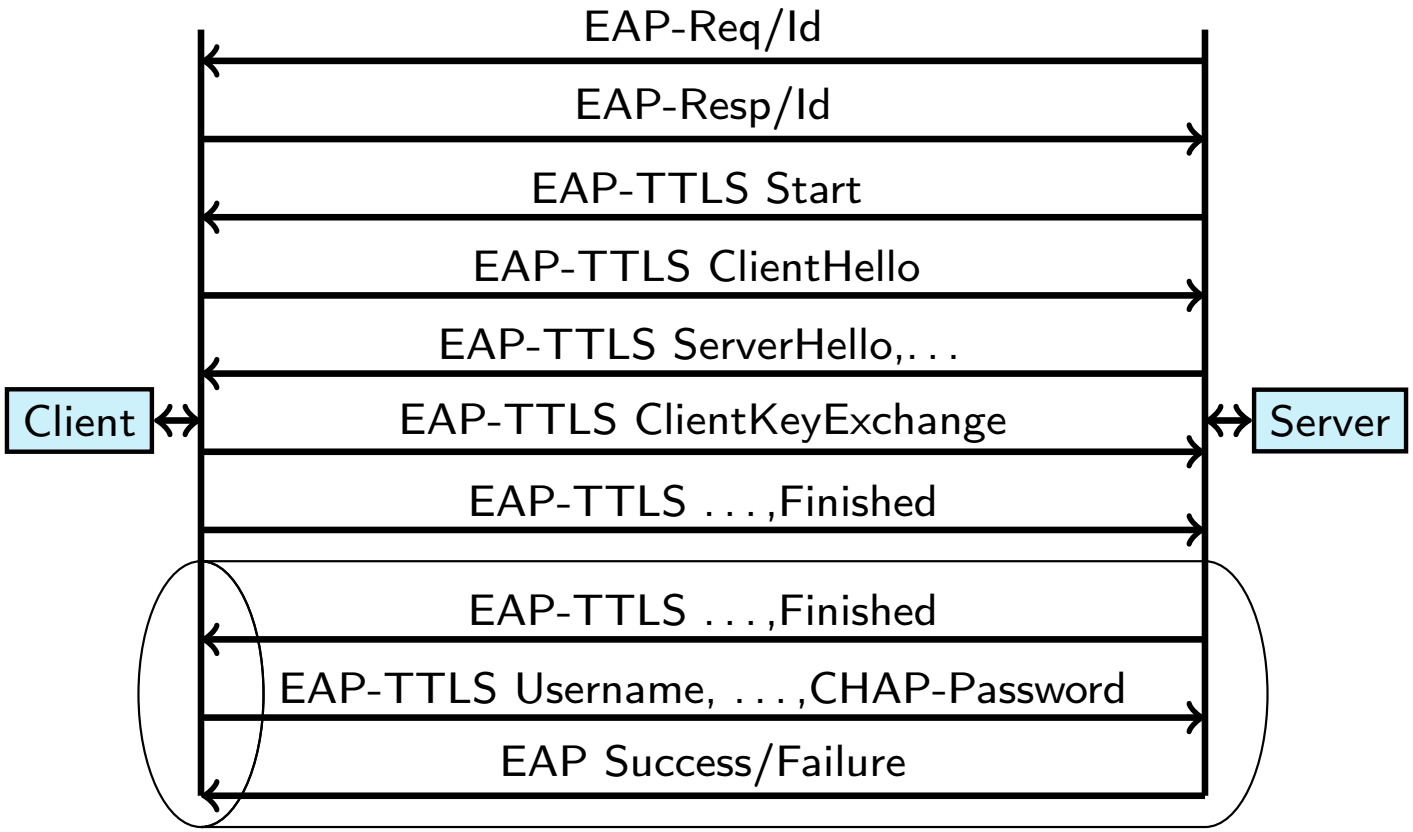
- On the 14th of October 2014, the POODLE attack on SSLv3 was published by Google
- SSLv3 is now 20 years old!
- Recommendation: Switch to TLS!
- Problem: Many systems have SSLv3 as fallback for legacy systems.
- POODLE attack allows an attacker to downgrade a TLS handshake to use SSLv3
- Recommendation: Disable SSLv3 completely!

Extensible Authentication Protocol

TLS

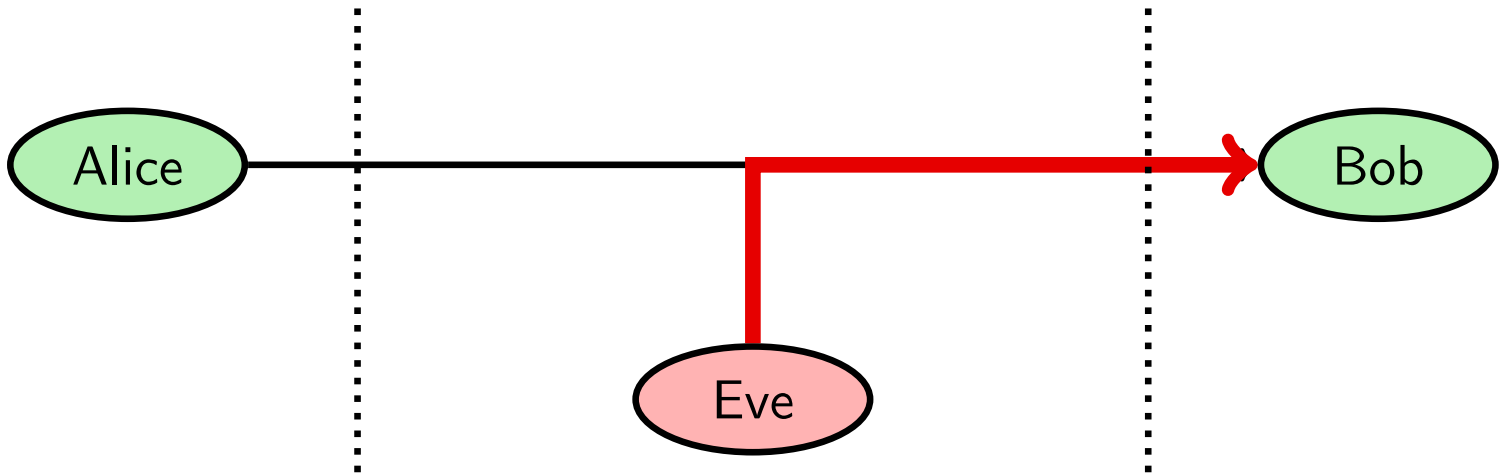


Extensible Authentication Protocol



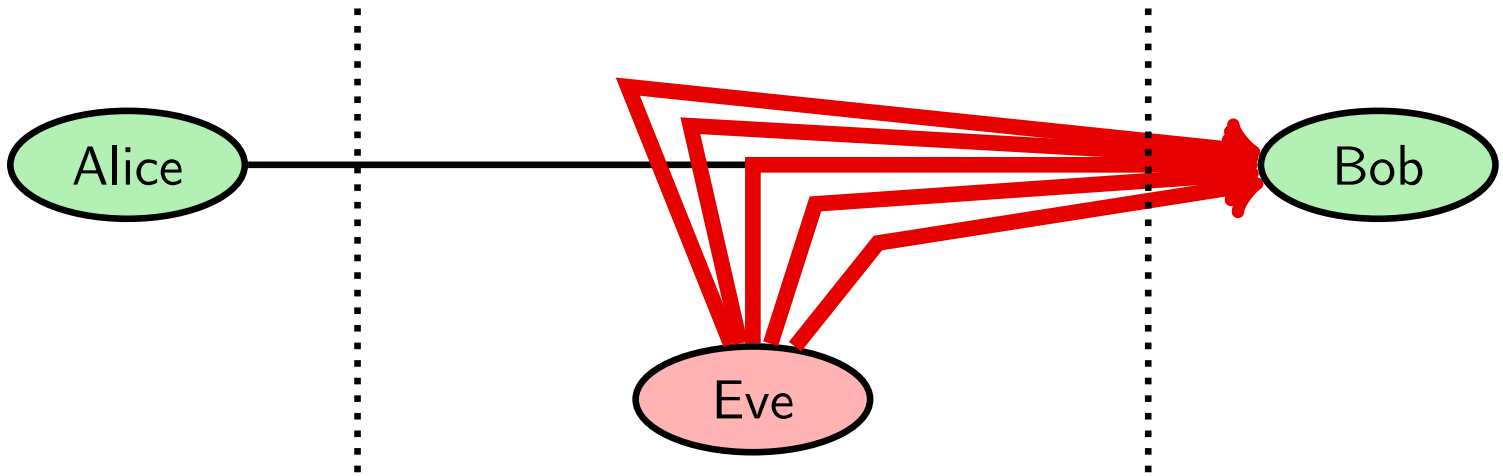
Communication and network security: Threat model

- Passive attacks: Eavesdropping, Wiretapping, Sniffing, and Traffic analysis
- Active attacks: Spoofing,



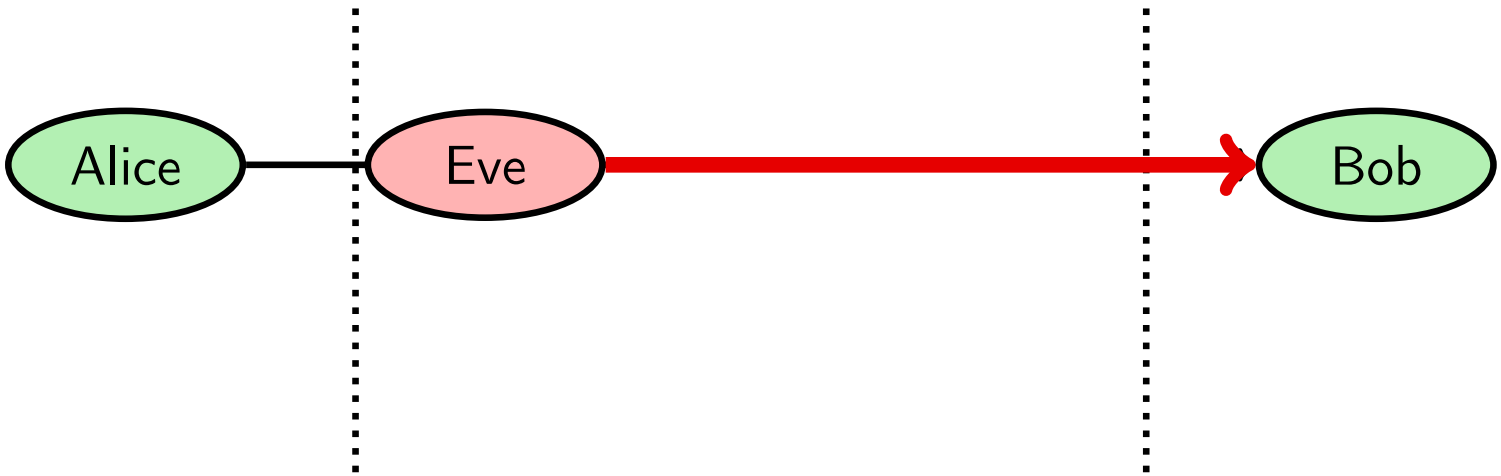
Communication and network security: Threat model

- Passive attacks: Eavesdropping, Wiretapping, Sniffing, and Traffic analysis
- Active attacks: Spoofing, Flooding,



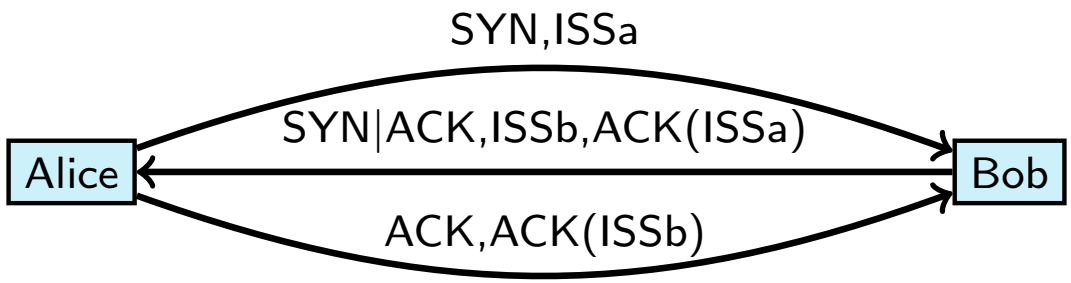
Communication and network security: Threat model

- Passive attacks: Eavesdropping, Wiretapping, Sniffing, and Traffic analysis
- Active attacks: Spoofing, Flooding, Squatting



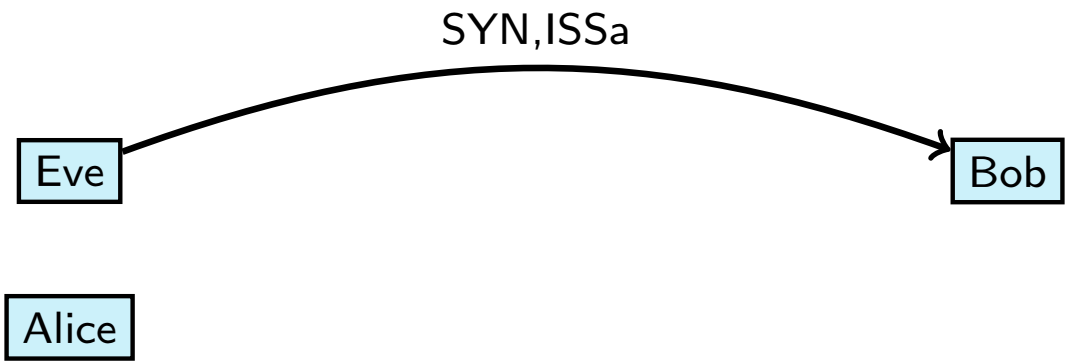
TCP session hijacking

- The below exchange starts a TCP session
- The acknowledgements are simple: ACK(ISSa)=ISSa+1



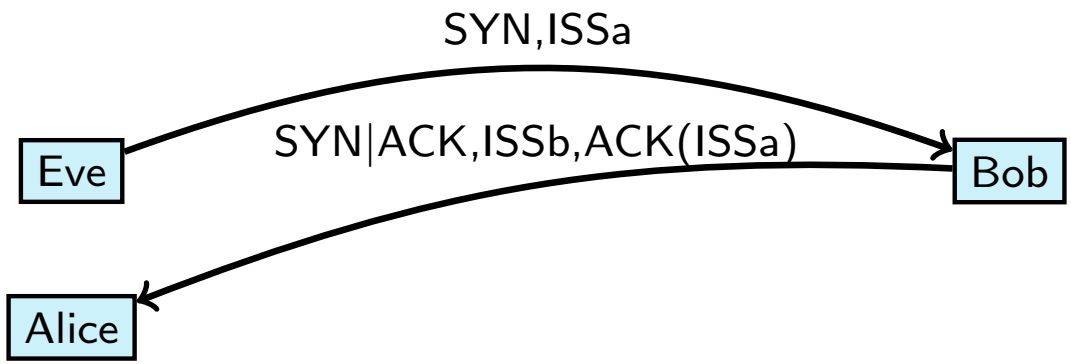
TCP session hijacking

- The below exchange starts a TCP session
- The acknowledgements are simple: ACK(ISSa)=ISSa+1
- Eve sends SYN,ISSa with Alice's response address



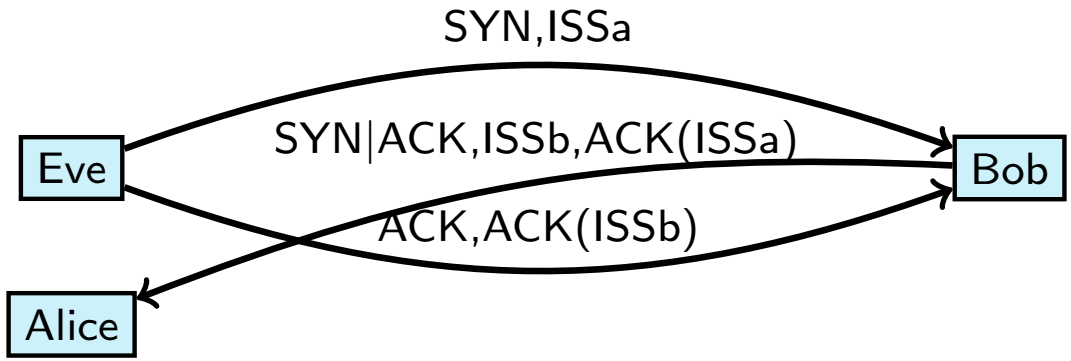
TCP session hijacking

- The below exchange starts a TCP session
- The acknowledgements are simple: ACK(ISSa)=ISSa+1
- Eve sends SYN,ISSa with Alice's response address
- She doesn't see the response, but ...

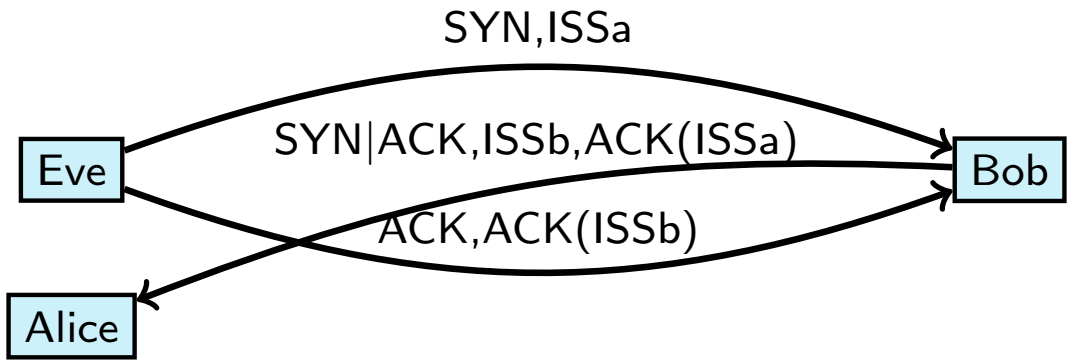


TCP session hijacking

- The below exchange starts a TCP session
- The acknowledgements are simple: ACK(ISSa)=ISSa+1
- Eve sends SYN,ISSa with Alice's response address
- She doesn't see the response, but ...
- If Eve can guess ISSb, she can hijack the session



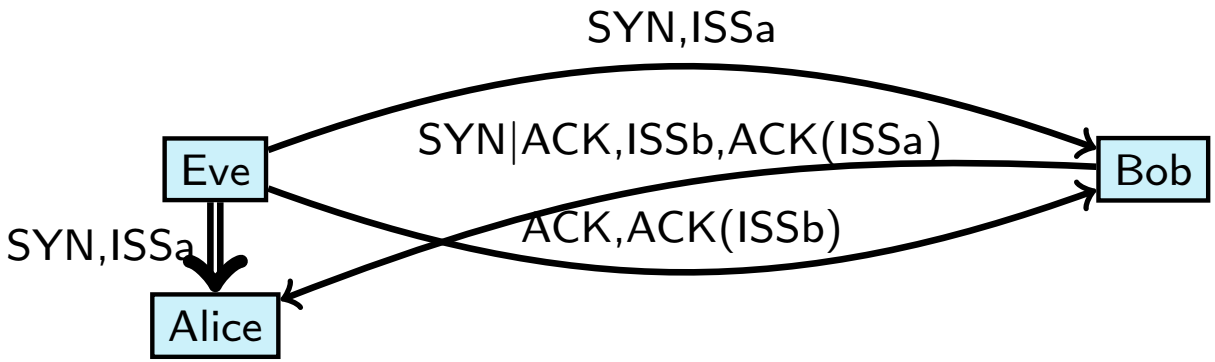
TCP session hijacking



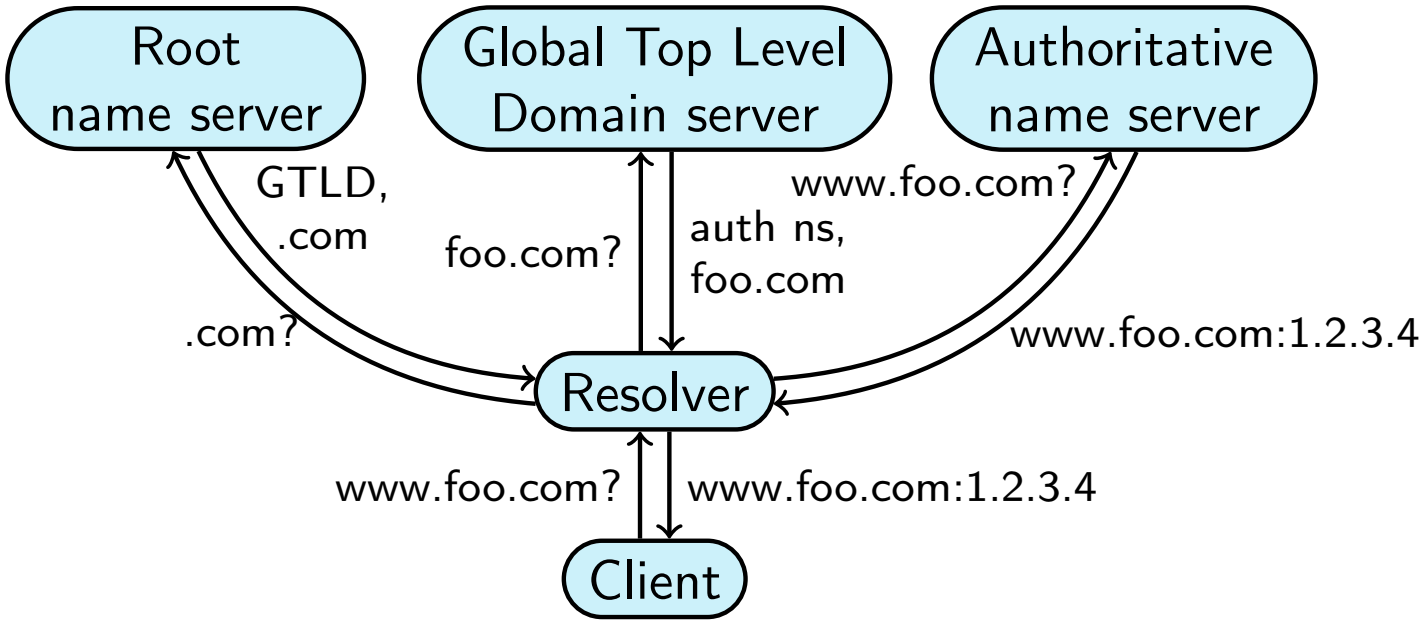
- Eve has established a (blind) session through session hijacking
- Certain protocols use no more authentication than this
- For these, Eve can use Alice's credentials at Bob
- Solution: firewall, or don't use services with address-based authentication

TCP SYN flooding

- To stop Alice from tearing down the faulty (to Alice) session, Eve can mount a SYN flood attack against Alice
- This is to exhaust Alice's resources
- An Availability attack here results in possible Integrity damage

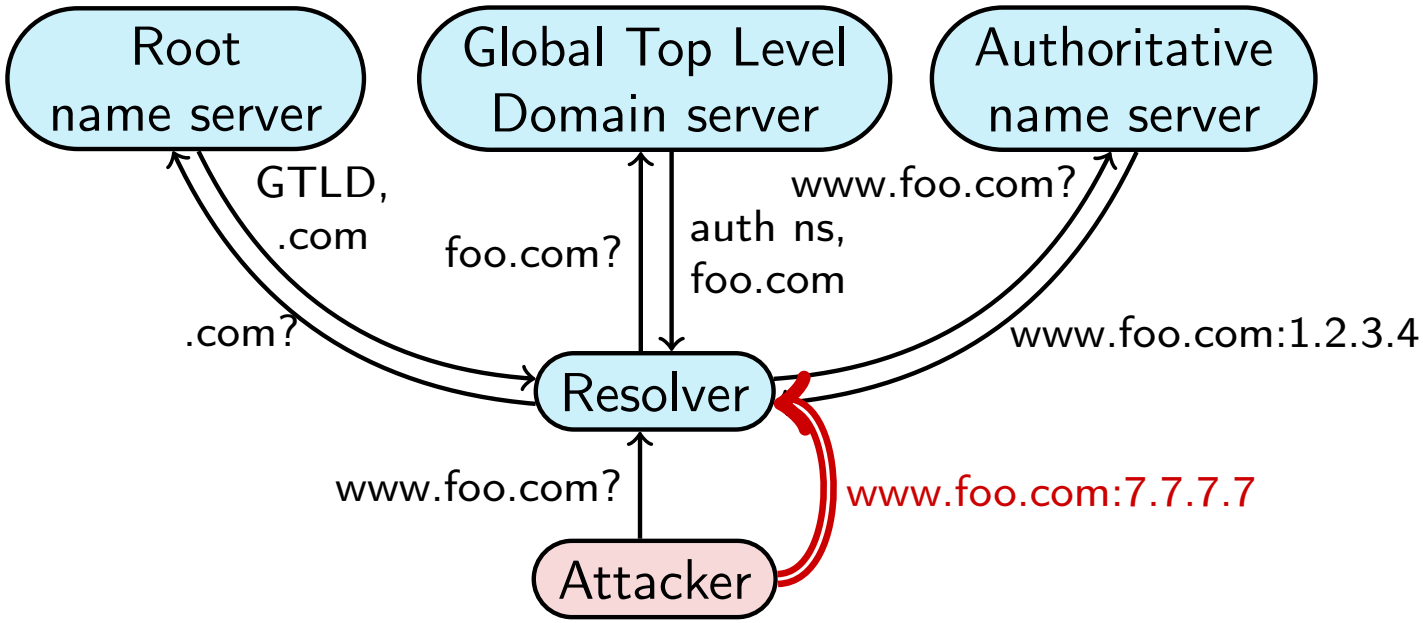


Domain Name System, DNS



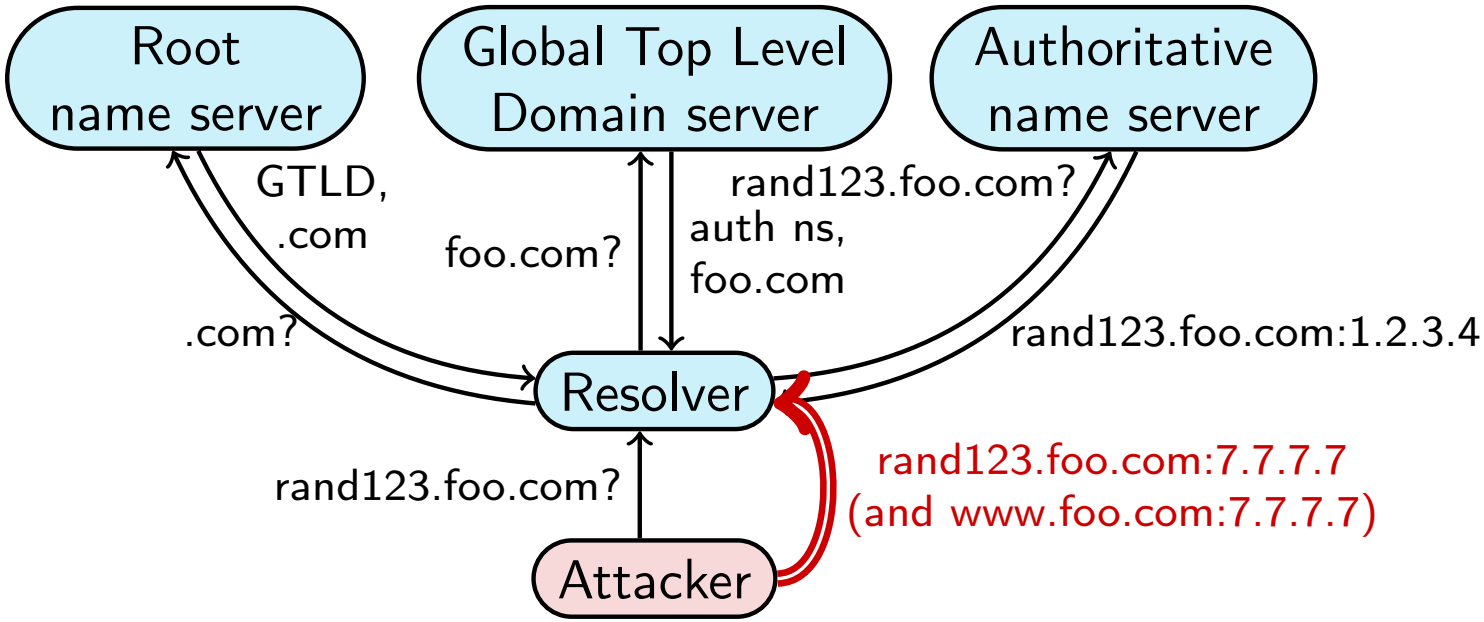
- DNS uses "Lightweight authentication", a 16-bit QID and a UDP response port that the answering server should use

DNS Cache poisoning



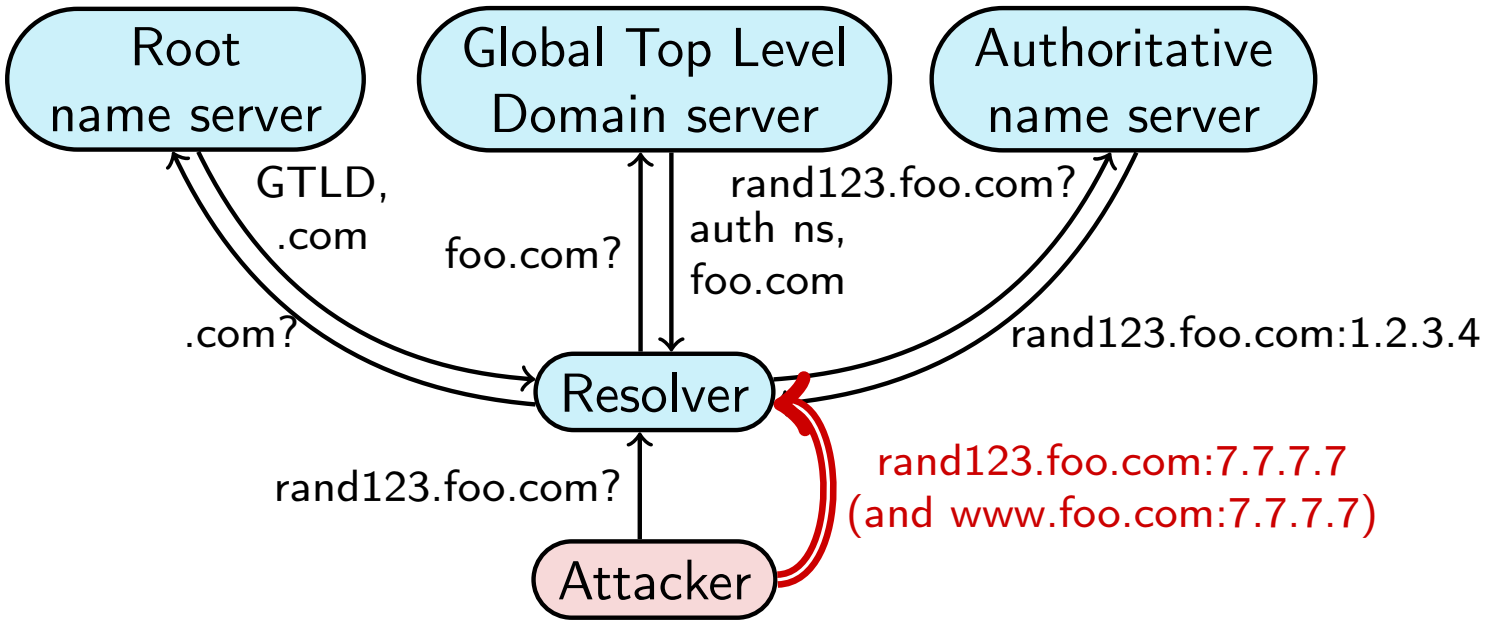
- Attacker asks for IP for target, then immediately floods the resolver with guessed QIDs at guessed UDP ports
- If successful, the attacker gets to decide Time To Live for the record

Dan Kaminsky's attack



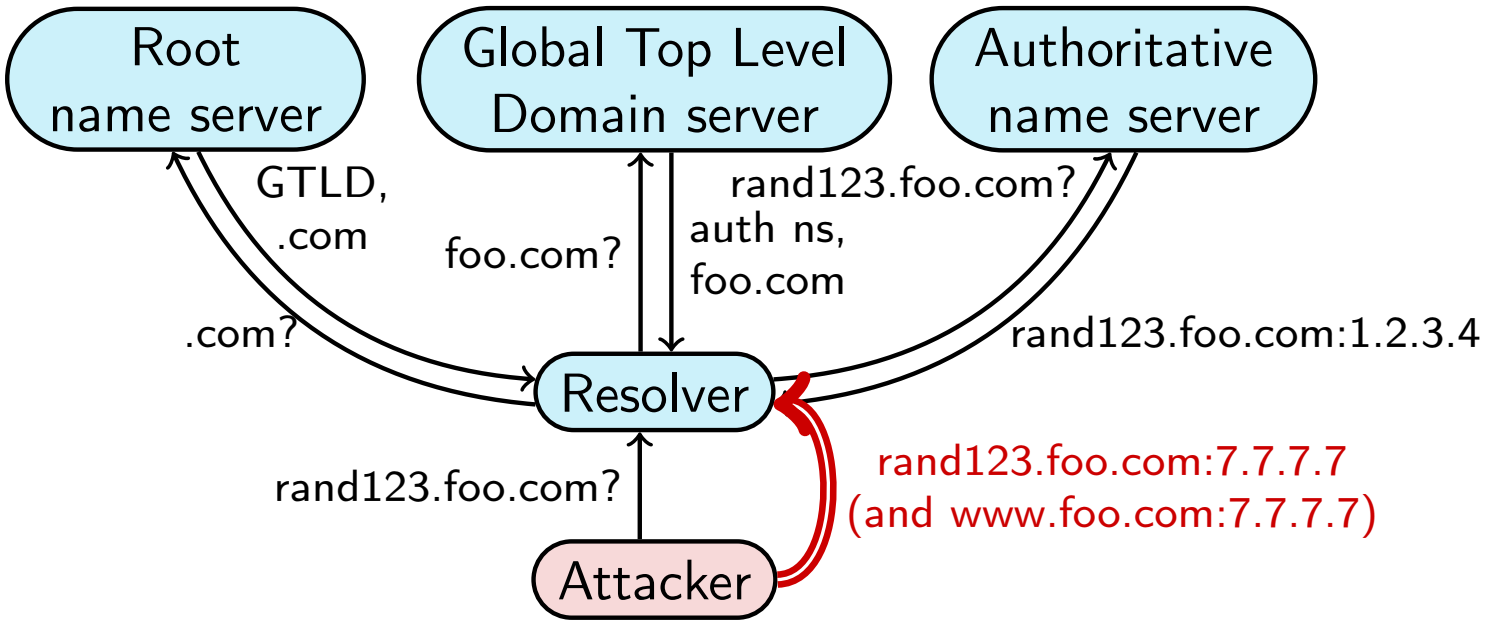
- Attacker asks for IP for random host in target domain, then immediately floods the resolver with guessed QIDs at guessed UDP ports

Dan Kaminsky's attack



- The attacker can include Additional Resource Records in the spoofed responses

Dan Kaminsky's attack



- The attacker can now try again without waiting for TTL expiry

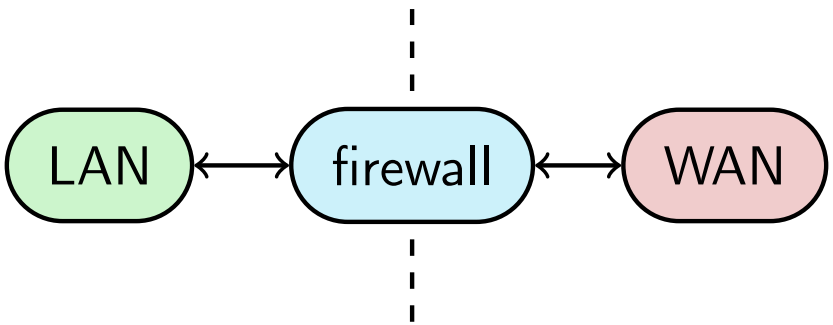
DNSSec

- DNS Security Extensions uses digital signatures to protect DNS records
- The DNS root is the trusted party
- The signature chain is built from the DNS root, through the TLD, and down to the current subdomain

- Not so easy to design a backward-compatible standard that can scale to the size of the Internet
- Worries of "zone enumeration", many feel their DNS info is confidential
- Disagreement among implementers over who should own the top-level domain root keys
- DNSSEC deployment is thought to be complex

Firewalls

- Main function: Filter traffic according to IP address and TCP port
- Do Network Address Translation to hide internal network
- Application proxies can do more, like filtering email for viruses and spam



Firewalls: packet filters

- Based on IP address and port (the Internet and Transport layers)
- Typical rules specify source and destination IP and TCP/UDP port number
- . . . and of course ALLOW or DENY
- In its simplest incarnation only allows static rules, say FTP to certain servers, or HTTP to others

Firewalls: stateful packet filters

- These can change the filtering rules depending on the session history
- For example, can handle the TCP SYN, SYN|ACK, ACK
- Another example are more complicated patterns like an inbound HTTP response from a server after an outbound HTTP request
- Filtering is limited by what is available in the packet headers

Firewalls: policy types

- There are two possible choices: Permissive policies or restrictive policies
  - Permissive policies (blacklisting) allow all connections except those known to be dangerous
  - Restrictive policies (whitelisting) deny all connections except those known to be secure (and useful)
- The latter is the more secure option
  - if you forget to allow something that people need, you will hear about it
  - if you forget to block a known attack path, you might lose your job

Firewalls: application proxies

- A proxy implements both server and client roles for a given protocol
- When a client connects to the proxy, the proxy checks if the request should be allowed
- If so, it acts like a client and connects to the destination server
- Responses come back to the proxy and also gets filtered before being passed on to the client

- An example is email virus filtering
- Proxies can be seen as performing controlled invocation

Firewalls: application proxies

- Application proxies typically run on hardened PCs
- These give a high level of control on filtering, so are secure from that point of view
- But they are more work-intensive to maintain
- Configuration is more complicated
- Often, one server is needed per service

Firewalls: Perimeter networks, or DMZ

- DMZ stands for De-Militarised Zone
- Some services must be accessible from the outside and from the inside
- One way to solve this is to place it on the border, with different access rules from the other machines on the protected network
- Sometimes this is done through two firewalls, an inner and an outer
- Some firewalls have a special interface for the DMZ clients

Firewall problem: protocol tunneling

- Since ports other than HTTP are often blocked, services tend to tunnel through port 80
- Even worse, some tunnel through SSH (or HTTPS), and these cannot be monitored
- Creating a proxy will not be popular, since end-to-end security will be lost

- Some think that separate firewalls soon will be a thing of the past
- The personal firewall will move network security back onto the end system

Intrusion Detection System, IDS

- Intrusion cannot always be stopped
- One common tool is an IDS, that detects attacks under way, or after the fact
- An IDS has *sensors*, either on the hosts or on the network, or preferably both
- Two approaches: misuse detection and anomaly detection
- Communication between sensors and IDS should be protected
- The IDS itself can be attacked

IDS: vulnerability assessment

- How secure is the network?
- Open ports, running software, patch schedule, network topology, ...
- Several organizations track vulnerabilities and publishes lists of available patches
- CERT (Computer Emergency Response Team), SANS, Security Focus (which carries Bugtraq) are examples (high traffic!)

IDS: misuse detection

- Misuse detection looks for attack signatures
- Known attacks create special network use patterns, or failed login attempts, or particular types of packets like the ones used in SYN flooding, ...
- Misuse detection can only be as good as the attack signature database
- Needs constant updates to keep up with the latest attacks
- Commercial IDSes are of this type (also called knowledge-based IDS)

IDS: anomaly detection

- Anomaly detection uses statistical techniques
- The IDS first establishes a baseline, of "normal" behaviour
- Under operation, monitored data is compared to the baseline, and if very different, an alarm is raised
- This does not need an attack signature database, but an anomaly may not mean there has been an intrusion
- Requires predictable "normal" behaviour
- Attacks may not necessarily give an anomaly
- A careful attacker can perhaps shift the "normal" level over time so that the IDS does not notice his/her attack

Network IDS, NIDS

- Typically monitors all network traffic at a node
- Captures what goes on when for example a network is scanned
- Can detect attacks before they hit a host
- But it can be hard to find the pattern in the immense dataset
- Can not do much about encrypted traffic

HIDS

- Monitors what happens on a host
- Uses logfiles and regularly checks for changes of system-critical files
- Some listen to port activity; here encryption is less of a problem
- Automated responses are tricky, since the attacker may use the response of the IDS as part of his attack

Honeypots (honeynets)

- False "targets" set up to lure attackers away from real target
- Can keep the attacker busy, so that tracing information can be collected
- Can keep the attacker occupied, while stronger defences are deployed
- Can avoid attacks at real target
- Can serve as "canaries"
- Needs some care to set up; they must look real to serve their purpose

Jonathan Jogenfors

www.liu.se