

1 Crypto Recap

1.1 Objectives:

- Confidentiality
- Integrity
- Authenticity
- Availability
- Authorization
- Non-Repudiation, Accountability
- Freshness
- Anonymity, Unlinkability
- Intervenableity, Contro
- Transparency

1.2 Confidentiality-Encryption

1.2.1 Symmetric Ciphers

- Secret key for en- and decryption
- Much more efficient
- Block cipher:** encrypts a plaintext block of fixed len e.g.: *Advanced Encryption Standard (AES)*
- Stream cipher:** encrypts a bitstream e.g.: *ChaCha20*

1.2.2 Asymmetric Ciphers

- Public key for encryption
- Private key for decryption
- Ex.: RSA-based encryption

1.3 Integrity, Authenticity-Signatures, MACs

1.3.1 MACs

- Symmetric cryptography
- Protects data integrity & authenticity
- Ex.: Hash-based MAC

1.3.2 Digital Signatures

- Asymmetric cryptography
 - Signing with private key
- Protects data integrity & authenticity
- Provides non-repudation

1.4 Block Cipher Modes of Operation

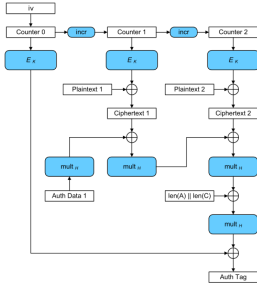
1.4.1 Electronic Code Book (ECB)

- Each plaintext block is encrypted seperately
- Inherintly insecure! -> Smae block = Same cipher

1.4.2 Cipher Block Chaining (CBC)

- Plaintext is chained to previous ciphertext by XOR and encrypted afterwards
- Difficult to apply securely -> implementations often vulnerable

1.4.3 Galois Counter Mode (GCM)



2 Tranport Layer Security (TLS)

2.1 TLS handshake protocol

- Parameter Negotiation
- Key exchange
- Authentication

2.2 TLS record protocol

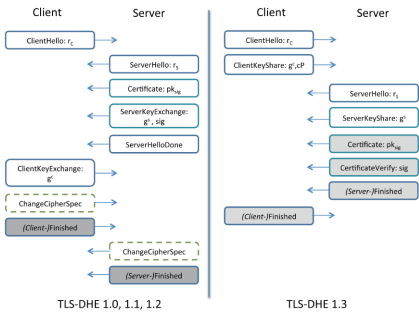
- Protection of integrity, authenticity and confidentiality
- Symmetric Cryptography:** e.g., block cipher, usually AES

2.3 Attacks

- Attacks on the record layer
- Attacks on the session key
- Attacks on the private server key
 - Attacks on implementations: Timing Attacks, Heratbleed, Invalid Point Attacks
- Attacks on TLS eco system

- Attacks on certificates and the PKI
- Attacks on the browser GUI

2.4 Overview TLS 1.0-1.2 & TLS 1.3



2.5 HKDF Key Derivation Function

2.6 Forward Secrecy

- TLS 1.3 using certificate-based Authentication: **forward secrecy**
- TLS 1.3 using pre-shared keys (PSK):
 - PSK using *elliptic-curve* Diffie-Hellman: **forward secrecy**
 - PSK without EC-DHE (symmetric-only) and zero-round-trip data: **no forward secrecy**

2.7 Datagramm TLS

- DTLS is identical to TLS where possible
- DTLS has to introcue new mechanisms
 - Explicit sequence numbers
 - Retransmission timer for handshake message
 - Message re-ordering for the handshake
 - Fragmentation of large handshake messages into severall DTLS records
 - Optional replay detection
 - Invalid records are discarded (silently)
 - Denial-of-Service countermeasures: statless cookies, HelloRetryRequest

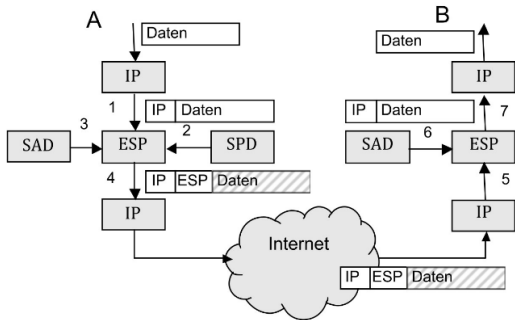
3 Network Layer Security: IPsec

3.1 TLS limitations

- Does not protect all traffic between hosts
 - Protects only a single connection; new connection => new key agreement or session resumption necessary
 - Non-TCP traffic (resp. non-UDP traffic for DTLS) cannot be protected
- Does not protect the TCP layer (RST attacks on TCP are possible which terminate TLS connection)
- Application specific: applications have to be modified to use TLS/DTLS
- Using a TLS-based tunel for VPNs has disadvangates (TCP) (DTLS good option tho)

3.2 Overview IPsec

- Goal: Protect IP packets cryptographically
 - Confidentiality
 - Integrity & Authenticity
- Seperation of packet protection from key exchange
 - Protocols for packet encryption and authentication: symmetric crypto, implemented in the TCP/IP stack
 - Key Exchange completely unrelated: asymmetric (+symmetric) crypto, implemented as network daemon



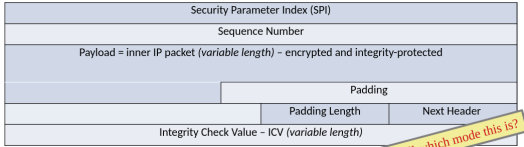
3.3 Packet Formats and Operational Modes

3.3.1 Authentication Header (AH)

- Integrity & Authenticity only
- Partially protects outerIP header
 - Variable fields cannot be included
 - Flags, Fragment Offset, TTL, Header Checksume
 - Other fields are included in MAC computation
 - Version, IP header length, total length, identification, Protocol (must be AH), source/dest Address

3.3.2 Encapsulating Security Payload (ESP)

- Confidentiality, Integrity & Authenticity
- Does not protect outerIP header



- SPI: SA-Identifier
- Next Header: type of payload data (e.g. IPV4)
- Integrity Check Value: data for authentication / integrity protection
 - Message Authentication Code (MAC)
- SAs are negotiated in the key exchange
- Sender side: Packet must be categorized to determine which SA applies (By parameters such as destination=)
- Receiver side: Determines SA from IPsec header of received packet
- SA Parameters:
 - IPsec protocol (AH or ESP)
 - Authentication algorithm and key
 - Encryption algorithm and key
 - IPsec mode (transport or tunnel)
 - SA lifetime
 - Sequence number: current counter
 - On reciver side: SSliding Window"for replay protection

3.4 Scenarios

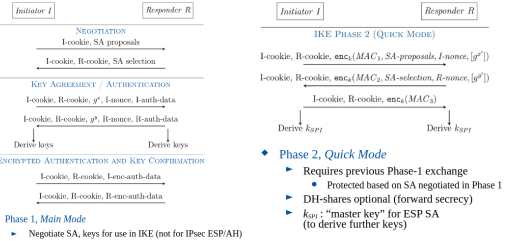
- Host to host
- Host to gateway
- Gateway to gateway

3.5 The Internet Key Exchange (IKE)

- So far, keys are already exchanges, SAs negotiated... but how?
- Authentication of the communication endpoints
- Dynamic negotiation fo algorithms and parameters
- Key establishment
 - Forward Secrecy
- Resistance to DoW attacks
- Efficiency (computation, messages, round trips)

3.6 IKEv1

- Phase 1: main mode vs. aggressive mode
- Authentication modes:
 - Digital Signatures
 - Public key encryption(PKE): two variants (rarely used)
 - Pre-shared keys(PSK)
- Phase 2 (quick mode)
 - With DH (=> forward secrecy)
 - Without DH (=> no forward secrecy)



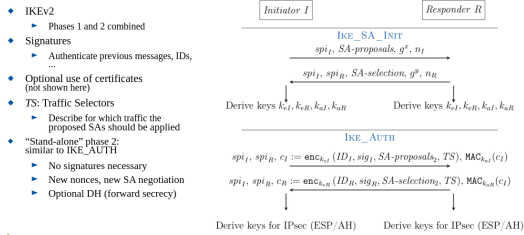
3.6.1 Discussion

- Complicated
 - Information from severall (complex) RFCs required
 - Many options variants
- Generic
 - Clean separation of different phases and functionalities
 - Can be advantageous in theory, but leads to inefficiencies
 - Intended as general handshake and key agreement protocol, not only for IPsec
 - In practice: only used for IPsec

- Inefficient
 - Requires too many round trips
- Inadequate DoS protection
 - IKEv1 uses statelful cookies

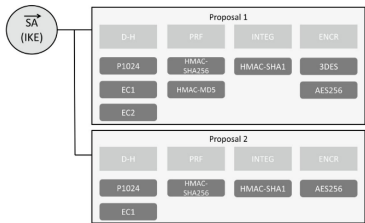
3.7 IKEv2

- Phase 2 (partially) combides with Phase 1
 - More efficient than IKEv1
 - Initial negotiation of IPsec SA included
- Additional SAs can be negotiated in further Phase 2 protocol runs
- Simplified specification
 - Essential information in one RFC
 - No distinction main mode vs. aggressive mode
- DoS protection using statless cookies: optional



3.8 SA Proposals

- Serverl proposals possible
- Proposal contains transforms
- Different options for each transform possible
 - Exactly one transform has to be selected



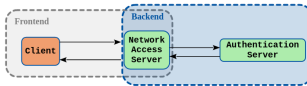
4 Security on Layer 2 (MAC Layer, Ethernet)

4.1 Attacks

- MAC address spoofing
- ARP spoofing
- ARP cache poisoning

4.2 Network Access Control

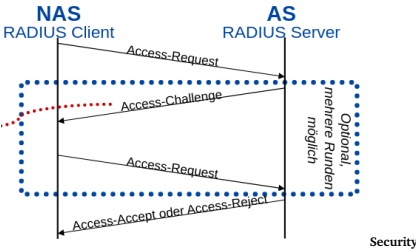
- Frontend protocols: Client -> Network Access Server
 - Point-to-Point connections: PPP
 - LAN: PPPoE, EAPoL
 - WLAN: WPA/WPA2/WPA3, EAPoL
- Backend protocols: Network Access Server -> Authentication Server AAA protocols (Authentication, Authorization, Accounting)
 - Radius, Diameter
 - TACACS+



4.2.1 Radius - Remote Authentication Dial-In User Service

- Backend protocol
- Dial-In Server
 - E.g. DSL
 - Company infrastructure

Packet Format



- Old protocol
 - Tries to protect sensitive data only
 - Insecure cryptography
- Uses only over secure protocols
 - Usually TLS
- RADIUS successor: Diameter
 - More flexible, extensible, application-aware
 - SSecurity features removed, no more insecure crypto of RADIUS => To be used over secure protocols TLS/DTLS/IPsec

4.2.2 Point-to-Point Protocol (PPP)

- PPP: Frontend protocol to connect two devices
 - Typically used for dial-up connections (e.g. DSL)
 - Transported over Layer 2 protocol (e.g. ethernet)
- Authentication mechanisms
 - PAP: username/password
 - CHAP: Challenge Handshake Authentication Protocol
 - EAP: Extensible Authentication Protocol
- PPPoE: PPP over Ethernet (e.g. used for DSL connections)
 - Discovery: Client negotiates session with network access server
 - PPP Session: PPP frames are encapsulated in ethernet frames; potentially several hops

4.2.3 Extensible Authentication Protocol (EAP)

- Extensible authentication framework
 - Fronted protocol for client authentication
 - Server send request (e.g. challenge, ident. request, ...)
 - Client responds
 - Server sends success or failure message
 - EAP messages can be transported in RADIUS messages (as RADIUS attribute) for backend communication
- EAPoL: EAP over LAN

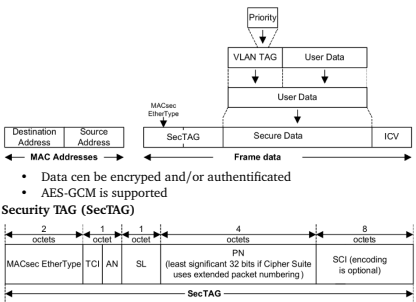
4.2.4 Port Based Network Access Control (PNAC)

- Authentication of clients before they can use the network (LAN)
 - Client connect to the network
 - * Authentication traffic is allowed
 - Switch port can only be used for other purposes after successful authentication
 - Re-authentication after timeouts, link-down, etc.
- PNAC uses EAPoL
- Terminology: Port Access Entities (PAE)
 - Client: supplicant
 - Network access server: authenticator
 - Attention: "port"

4.3 MACsec

- cryptographically protect the ethernet layer
- Based on MACsec Key Agreement (MKA)
 - After successful authentication (typically EAP)
 - Derives key from CAK - connectivity association key (pre-shared)
 - Key can be set up by EAP (e.g. EAP-TLS)

4.3.1 Frame



- TCI: TAG Control Information

- AN: Association Number
- SL: Short Length (payload length; for short frames only)
- PN: Packet Number
- SCI: Secure Channel Identifier (identifies SA, when CA needs more than 4)
 - SA is identified by SCI (optional) and AN

4.4 Summary

- Security on Layer 2 can protect all higher layers
 - But: only in same network
 - Does not work across Layer 3 routers
- Network Access Control protects access to the network
 - Client Authentication
 - RADIUS/DIAMETER
 - EAP
 - Port-Based Access Control
- MACsec protects traffic
 - Encryption & authentication

5 Wireless Security

5.1 WiFi Security

5.1.1 Historic Overview

- 1999: WEP (Wired Equivalent Privacy)
 - Goal: as secure as a wired LAN
 - Insecure, various attacks known
- 2003: WPA (WiFi Protected Access)
 - Improved protocols; most known attacks on WEP prevented
 - Enterprise Mode
 - But: Requirement of hardware-compatibility with WEP devices => Encryption improved, but still based on obsolete stream cipher
- 2004: WPA2 (still used)
 - Similar to WPA, but AES-based encryption: AES-CCMP
- 2018: WPA3 (supported by new devices)
 - Several improvements: Prevention of offline-attacks on pre-shared keys, forward secrecy, encryption for open WLANs

5.1.2 WPA/WPA2 Security

- Personal Mode
 - Pre-Shared Keys
- Enterprise Mode
 - EAP-TLS, PEAP, EAP-TTLS
- AES-CCMP

- Authenticated Encryption with Associated Data (AEAD)
- AES-CCM
 - Authentication: CBC-MAC
 - Encryption: Counter Mode (CTR)
 - MAC and encryption: computed simultaneously

4-Way Handshake

- Based on Pairwise Master Key (PMK)
 - Personal Mode (WPA-PSK): Computed from passphrase and SSID (as f(alt)) using PBKDF2 (password-based key derivation function)
 - Enterprise Mode: Established by key exchange protocol (e.g. EAP-TLS, PEAP)
- 4-Way HS to derive Pairwise Transient Key (PTK)
 - Exchange nonces
 - PTK is derived by hashing PMK, nonces, MAC address
 - Further key (for different purposes) derived from PTK
 - Client gets Group Temporary Key from AP (encrypted)
 - Message Integrity Code (MIC): MACs for integrity protection, key confirmation
- KRACK (2018): Key reinstallation attacks => meanwhile prevented by software/firmware updates
- Problem: Offline attacks against passphrase

5.1.3 WPA 3 Improvements

- Mandatory Protected Management Frames
 - Prevents deauthentication attacks (DoS)
- Replace PSK Authentication with SAKE (SAE): "Dragonfly" handshake
 - Simultaneous Authentication among Equals (SAE): "Dragonfly" handshake
 - Prevents offline attacks on passphrase
 - Based on elliptic curve cryptography by default
- Forward Secrecy based on Diffie-Hellman
- 192-bit Security Mode (optional)
 - AES-256 (GCM)
 - SHA-384
 - 284-bit elliptic curves or RSA with at least 3K bits

5.1.4 Simultaneous Authentication among Equals

- SAE "Dragonfly" authenticates participants and establishes PMK
 - Based on passphrase and (EC-)Diffie-Hellman
 - Can be initiated simultaneously by both parties (useful for mesh networking)
- 4-Way Handshake
 - Establishes PTK based on PMK
 - Same as in WPA2
 - But now: PMK with much higher entropy => Offline attacks not practical
- Hash-to-Group: "Hunting and Pecking"
 - Generate point on elliptic curve from passphrase (and MAC addresses, etc.)
 - Cryptographic hash function generates pseudo random numbers (by including a counter in the input)

- * Both parties must use the exact same inputs in the same order
 - Fixed procedure to derive x-coordinate
 - * Check if point on curve can be generated
 - * If check fails: increase counter and try again
- Auth-Commit messages
 - Exchange ECDH shares
- Auth-Confirm messages
 - Key confirmation, authentication of messages

5.2 Bluetooth Security

- Authentication: device authentication, no user authentication
- Pairing/bonding: create shared keys; used in connections later on
- Confidentiality: encryption of BT communication
- Message Integrity: MACs (authenticated encryption) to protect BT communication
- Authorization: control access to resources (based on devices, not users)
- Security Modes
 - Mode 1: no security
 - Mode 2: service level (only for backward compatibility)
 - Mode 3: link-level enforces security (only for backward compatibility)
 - Mode 4: authenticated link key using Secure Connections, based on device pairing
- Eavesdropping not trivial: Bluetooth uses frequency hopping (not a security feature)

5.2.1 Device Pairing

- Authentication and generation of link key / long term key
- PIN/Legacy Pairing: enter PIN on both devices
 - Key generation based on PIN, device address, and random values
- Secure Simple Pairing (SSP): since Bluetooth 2.1
 - Numeric Comparison
 - * Compare 6-digit numbers
 - Passkey Entry
 - * Read 6-digit from one device, enter on the other one
 - Just Works
 - * User accepts connection without verification
 - Out of Band (OOB)
 - * Transmit data using other communication channels (e.g. NFC)

5.2.2 Simple Secure Pairing (SSP)

- Unauthenticated ECDH
- 2-Stage Authentication
 - Stage 2: depends on pairing method
 - Stage 2: Cryptographic authentication based on Stage 1 values and ECDH secret
- Key derivation to generate link key / long term key

5.2.3 Secure Authentication

- Paired (bonded) devices authenticate each other
- Challenge-Response scheme
 - 128-bit random challenges
 - Response: HMAC of BT addresses and challenges (using link key from pairing)
 - * Before Bluetooth 4.1: based on Bluetooth-specific algorithm E1
- Authentication failure: introduce delay (exponential back-off)

5.2.4 Confidentiality

- Bluetooth-specific stream cipher E0
 - Designed for efficiency
 - Serious attacks have been published
 - "Practical theory (but complex, hard to apply in practice)"
- AES-CCM
 - Used since Bluetooth 4.1
 - Key derived from link key (pairing) and the authentication step

5.2.5 Privacy

- Privacy problem: Devices (users) can be identified by Bluetooth MAC addresses
- Mitigation: BLE private device addresses
 - Resolvable Private Address (RPA) is changed periodically
 - Identity Address remains constant (but is not transmitted over the air)
 - Identity Resolving Key to map RPA to Identity Address
 - Especially important to discoverable devices (which advertise identity info)

5.2.6 5.x Security

- No major changes to security protocols and algorithms
- Bluetooth 5.0
 - PHY improvements, no relevant security changes
- Bluetooth 5.1
 - HCI support for debug keys (should not be relevant in production systems)
- Bluetooth 5.2: adds new features (Extended Attributes, Isochronous Communication, ...)
 - Isochronous communication: connection-oriented or connection-less
 - * Group communication: group keys need to be established
 - * Broadcast Authentication
- Bluetooth 5.3: Key Size Negotiation
 - Enables host to define minimum key size

5.2.7 BLUFS

- BLUFS: New attacks against bluetooth
 - Breaks Forward Secrecy and Future Secrecy
 - Enables man-in-the-middle attacks, impersonation if one session key compromised
 - Forces weak key: spec allows minimum of 7 Bytes entropy (56 Bits)
 - * Brute-force attack: offline, parallelizable

- * Forces reuse of compromised key
- Attack against bluetooth spec (BR/EDR: "Bluetooth Classic") versions 4.2 to 5.4): All compliant devices are affected
- Published and presented at ACM CCS 2023

5.2.8 implementations Vulnerabilities

- BlueBorn (2017): Collection of implementation vulnerabilities
 - On Windows, iOS, Linux, Android
 - Buffer overflow, integer overflows, ...
- Android (2018): implementation flaws in L2CAP and SMP
 - Remote Memory Disclosure
- BleedingTooth (2020): several bugs in Linux
 - Can even lead to arbitrary code execution in kernel mode
- Windows (2021): BT Driver Elevation of Privilege
- BrakTooth (2021)
 - Bluetooth controllers: SoC firmware vulnerabilities (Link Manager)
 - Estimation 1400 bluetooth chips/modules affected

5.2.9 Summary

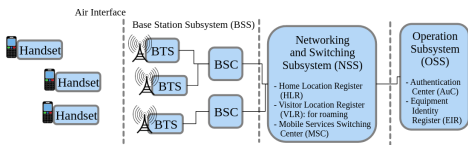
- Complex protocol stack, not easy to implement
- Many attacks in the past
 - on cryptography algorithms
- Bluetooth versions before 2.1 are basically completely insecure
- Bluetooth versions since 4.2 are relatively secure (...but: "BLUFS")
 - But the implementations not necessarily!
- Bluetooth 5.2 architecture similar to 4.x
 - Introduces new features and minor security improvements

6 Security Mobile Networks

6.1 Overview

- 1st generation: analog telephony networks (voice only)
- 2nd generation (2G): GSM 1990s
 - Digital, circuit-switched, SIM-based; introduced cryptographic protection
 - "Generation 2.5": GPRS (packet-switched data traffic)
 - "Generation 2.75": EDGE (increased bandwidth)
- 3rd generation (3G): UMTS 2000s
 - Much higher bandwidth
 - Improved security: mutual authentication, better crypto algorithms
 - "Generation 3.5": HSDPA/HSUPA (increased bandwidth)
- 4th generation (4G): LTE 2010s
 - Higher bandwidth
 - Completely new IP-based core network
 - Changes to security
- 5th generation (5G): 2020-?
 - ... even better, of course ;) ...

6.2 GSM Network



BTS: Base Transceiver Station
BSC: Base Station Controller

- Circuit-switched network, designed for voice communication
 - Stable channel between communication partners is reserved
 - Separate channel for signaling (different time slices)
 - Slow, expensive
 - GSM introduces SMS: small texts can be transferred in signaling channel

6.2.1 Security

- Goal: At least as secure as a landline"
- Security functions
 - Authentication and Key Agreement (AKA)
 - * Shared secret in AuC and SIM
 - * Authentication via challenge-response protocol
 - * Agreement on short term session keys for encryption
 - Confidentiality Protection
 - * Encryption of user data ("user plane")
 - * Encryption of signalling ("control plane")
 - Integrity Protection
 - * Only for signalling
 - * Often implicitly by encryption

6.2.2 (U)SIM

- SIM card introduced with GSM
 - SIM: Subscriber Identity Module
 - International mobile subscriber identity (IMSI)
 - * Temporary mobile subscriber identity (TMSI): derived from IMSI
 - Security token (Smartcard) of the network operator
 - * Typically implemented on a JavaCard

- **Network Operators and 3rd Parties Offering SServices[®]:**
 - **Network Functions (NF):** Core network features exposed as services.
 - **Other Services:** Additional functionalities provided by the network.
 - **TLS Support:** Ensures secure communication between entities.
 - **OAuth 2.0-based Authorization:** Allows secure access control using access tokens.
- **Network Slices:**
 - Provides virtualized and isolated network partitions with specific **QoS (Quality of Service)** properties.
 - Uses **EAP-based Authorization** for access control.

7 Firewalls, Intrusion Detection and Prevention

7.1 Firewalls

- Firewalls separate networks
 - Restricts traffic: enforces policy
 - Logging
 - Implemented in SW and/or HW)
- 7.1.1 Packet Filtering
 - Packet filtering firewalls:
 - Policies defined by filtering rules (source, destination, port, protocol, etc.).
 - Rules decide actions: accept, block, log, or modify packets.
 - Advanced filters can modify packets (e.g., NAT or port translation).
 - Advantages:
 - Cheap, widely available, and efficient.
 - Examples: Linux `netfilter`, Windows firewall.
 - Simple implementation; no modification to applications required.
 - Limitations:
 - Susceptible to IP and port spoofing attacks.
 - Rules limited to individual packets and lack comprehensive state awareness.
 - Cannot enforce application-specific filtering; lacks deeper protocol knowledge.
 - Usually not capable of rejecting specific packets within a particular application session.
 - Niche Aspects:
 - Filters can reject IP spoofing by blocking packets from internal addresses arriving at external interfaces.
 - Access control by ports can restrict services to specific source addresses (e.g., SMTP limited to internal IPs).
 - Can filter based on packet type (TCP/UDP) or flags (e.g., SYN for connection establishment).

7.1.2 Connection Tracking and Stateful Firewalls

- Connection Tracking:
 - Tracks packets belonging to the same logical "connection" (not limited to TCP).
 - Examples:
 - * UDP-based connections (e.g., DTLS).
 - * DNS requests/responses and ICMP echo/replies.
 - Supports protocols with separate control and data connections (e.g., active FTP).
 - Enables NAT functionality by rewriting addresses for complex protocols.
- Stateful Firewalls:
 - Maintain state information about active connections.
 - Rules can dynamically adapt based on connection states (e.g., allow replies to outgoing requests).

7.1.3 Linux Firewall: Netfilter

Netfilter, part of the Linux kernel, is a framework for packet filtering and network address translation (NAT). It operates using a series of hooks in the network stack where rules, organized into **tables** and **chains**, are applied.

Chains and Their Functions:

- **Input Chain:** Handles packets destined for the local system (e.g., filtering SSH traffic).
- **Output Chain:** Manages packets generated locally and leaving the system (e.g., limiting outgoing ICMP traffic).
- **Forward Chain:** Processes packets passing through the system but not destined for it (e.g., traffic routing in a gateway).
- **Prerouting Chain:** Alters incoming packets before routing decisions (e.g., DNAT for port redirection).
- **Postrouting Chain:** Alters outgoing packets after routing decisions (e.g., SNAT for source IP modification).

Key Components:

- **Tables:** Contain chains for specific tasks:
 - **Filter Table:** Default table for packet filtering (Input, Output, Forward chains).
 - **NAT Table:** Used for network address translation (Prerouting, Postrouting chains).
 - **Mangle Table:** For modifying packet headers.
- **Tools:**
 - `iptables`: Legacy tool for managing rules.
 - `nftables`: Modern replacement offering simplified syntax and better performance.

Use Case Example: A packet enters the system through the **Prerouting Chain**, is routed to the appropriate interface (via **Forward Chain** if being forwarded or **Input Chain** if destined locally), and exits through the **Postrouting Chain** after necessary modifications. Netfilter's modularity and flexibility make it a powerful tool for managing network traffic.

7.1.4 Application-Level Gateways

An **Application-Level Gateway (ALG)** acts as an intermediary between clients and applications, offering fine-grained control and monitoring of traffic specific to application protocols. It provides a specialized interface tailored for applications, such as SMTP or HTTP, allowing for filtering, logging, and user profiling at the application layer.

Functionality:

- **Connection Handling:** Clients connect to the gateway, which filters and analyzes the traffic before forwarding it to the application server.
- **Protocol Awareness:** Equipped with detailed knowledge of application-specific protocols, enabling in-depth traffic inspection and filtering.
- **State Management:** Tracks the state of connections or application data to enforce security or logging policies effectively.
- In-depth analysis and traffic filtering based on application logic.
- Enables user profiling, attack detection, and intrusion prevention.

- Provides application-specific caching and potential for secure communication.
- App extensions for additional features, such as authentication, without modifying the original application.

Limitations:

- **Performance Overhead:** The inspection and filtering process can reduce throughput.
- **Maintenance Effort:** Requires frequent updates to stay compatible with evolving application protocols.
- **Limited Scope:** Only supports predefined applications, limiting flexibility in heterogeneous environments.

Use Case: ALGs are frequently employed in enterprise environments for managing email servers (e.g., SMTP gateways) or for secure access to critical applications by acting as a filter and logging interface for external traffic.

7.1.5 Proxy Firewalls

Proxy firewalls act as a generic proxy on the transport layer, enabling advanced traffic control and enhancing communication security.

Key Features:

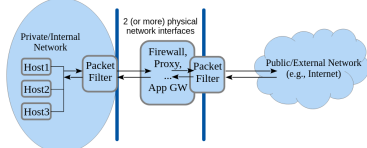
- **Connection State Tracking:** Monitors and manages the state of connections.
- **Authentication and Security:** Adds user authentication and communication security.
 - Frequently used in corporate environments:
 - * Outgoing traffic is routed only via the proxy.
 - * User authentication is required before access.
 - * Only specific ports or applications are allowed.
- **Flexibility:** Can extend functionality toward application-level filtering (e.g., web proxies or caches).

Comparison to Application-Level Gateways:

- Proxy firewalls are more general than application-level gateways.
- The distinction between proxy firewalls and application gateways is not always clear.

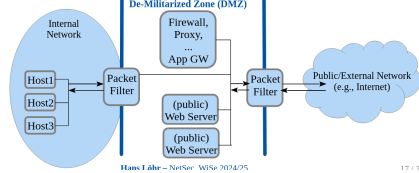
Firewall Architectures: Dual-Homed Firewall vs. DMZ

- Dual-Homed Firewall:



- **Concept:** Firewall with two interfaces separating internal and external networks.
- **Operation:** Traffic flows through the firewall, using packet filters or proxies for mediation.
- **Advantages:**
 - * Strong physical isolation.
 - * Centralized traffic control.
 - * Flexible integration of security mechanisms.
- **Disadvantages:**
 - * Requires additional hardware resources.
 - * Limited scalability for complex networks.
- **Use Cases:** Suitable for small/medium networks needing strict isolation.

- Demilitarized Zone (DMZ):



- **Concept:** Segregated network zone for public-facing services (e.g., web servers).
- **Operation:** Packet filters manage traffic to/from DMZ, preventing direct internal access.
- **Advantages:**
 - * Protects internal networks from compromised public-facing services.
 - * Granular traffic control for specific services.
 - * Limits exposure of internal systems.
- **Disadvantages:**
 - * Adds configuration and maintenance complexity.
 - **Use Cases:** Ideal for hosting public services (e.g., web hosting, e-commerce).
- **Key Differences:**
 - Dual-Homed focuses on isolation, while DMZ separates public services from internal systems.
 - DMZ adds a dedicated network segment for external-facing servers.

7.2 Intrusion Detection and Prevention System (IDPS)

- **Intrusion Detection Systems (IDSs):**
 - Detect attempts to attack or intrusions.
 - Focused on monitoring and logging suspicious activities.
- **Prevention:**
 - React to detected intrusions to prevent successful attacks.
 - Goal: Stop attacks in progress by blocking or responding.

- **IDPS = IDS + Prevention:** Combines detection and prevention to ensure proactive and reactive defense mechanisms.
- **Historical Work:**
 - James Anderson: *Computer Security Threat Monitoring and Surveillance* (1980).
 - Dorothy Denning: *An Intrusion Detection Model* (1987).

Why Use IDPS?

- Identify incidents effectively.
- Support incident response to minimize damage.
- Identify gaps in security policy and practices.
- Deter insiders from violating policies.

7.2.1 IDPS Functionality (High-Level)

- **Record Events:**
 - Logging and accounting for detected activities.
 - Observations useful for detecting and analyzing incidents.
 - Integration with Security Information and Event Management (SIEM) systems.
- **Notify Administrators:**
 - Alert administrators to take appropriate action.
- **Produce Reports:**
 - Summarize activities and detected threats for analysis.
- **Automated First-Level Reaction:**
 - Stop attacks or intrusion attempts automatically.
 - Dynamically change the security environment, e.g., reconfigure firewalls.

7.2.2 Intrusion Detection Model (Denning, 1987)

- **Components:**
 - **Subjects:** Entities initiating actions.
 - **Objects:** Resources being accessed or affected.
 - **Audit Records:** Logs of subject actions for monitoring.
 - **Profiles:** Behavioral characterizations based on metrics.
 - **Anomaly Records:** Highlight abnormal activities.
 - **Activity Rules:** Define responses to anomalies (e.g., profile updates, intrusion reporting).
- **Metrics:**
 - Event Counters
 - Interval Timers
 - Resource Measures
- **Models:**
 - **Statistical Models:** Identify statistical anomalies.
 - * Mean and Standard Deviation Model
 - * Multivariate Correlation Model
 - * Markov Process Model
 - * Time Series Model
 - **Operational Models:** Compare metrics against predefined thresholds.
 - **Modern Approaches:** Incorporate Machine Learning/AI for advanced anomaly detection.

7.2.3 Types of IDPS

- **Network-Based Intrusion Detection System (NIDS):**
 - Monitors and analyzes network traffic.
 - Often integrated with firewalls (tightly or loosely).
 - Includes specialized Wireless IDPS for wireless networks.
 - Incorporates Network Behavior Analysis (NBA) for deeper traffic insights.
- **Host-Based Intrusion Detection System (HIDS):**
 - Monitors activity on a specific host (server or end-user device).
 - Analyzes system and application logs, network activity, running processes, and system calls.
 - Tracks file accesses and configuration changes for anomaly detection.

7.2.4 IDPS Architecture

- **Sensors (Agents):** Monitor and analyze activity.
- **IDPS Management Server:** Collects information from sensors and manages them.
- **Database:** Serves as a repository for audit records.
- **Admin Console:** Provides a user interface for administrators.

7.2.5 Detection Methodologies: Categorization

- **Signature-Based Intrusion Detection**
 - Uses fixed patterns to identify *known threats* (e.g., hash values of malware).
 - Limited to known threats and can be evaded by intelligent attackers.
- **Anomaly-Based Intrusion Detection**
 - Detects previously unknown threats by modeling normal/abnormal behavior.
 - Requires a *training/learning phase* for data collection and profiling.
 - Often uses Machine Learning or AI; main focus of current research.
- **Specification-Based Intrusion Detection (Stateful Protocol Analysis)**
 - Compares observations to models/specifications of legitimate behavior.
 - Can be integrated into application-level gateways.

7.3 Summary

- **Firewalls**
 - Essential building block for network security.
 - Alone cannot effectively prevent attacks.
 - Various types:
 - * Packet filters.
 - * Application-level gateways/proxies.
 - Example: Linux firewall (netfilter, iptables/nftables).
- **Intrusion Detection and Prevention Systems (IDPS)**
 - Detect and prevent intrusion attempts.
 - Types:
 - * Host-based vs. Network-based IDPS.

- Detection methodologies:
 - * Signature-based.
 - * Behavior-based (e.g., anomaly detection).
- Specialized IDPS for specific systems (e.g., industrial automation, vehicle networks).

8 Border Gateway Protocol (BGP), Domain Name System (DNS)

8.1 Routing

- **Routing:** Finding the best path from source to destination.
 - *Internal routing:* Within a network or AS (Autonomous System).
 - *External routing:* Between networks or ASes.
 - Different types of routing algorithms exist.
- **Distance Vector Algorithms (e.g., Bellman-Ford):**
 - Practical for networks of limited size \Rightarrow used for internal routing.
- **Link-State Algorithms (e.g., Dijkstra):**
 - Computes the shortest path.
 - Suitable for networks of limited size \Rightarrow used for internal routing.
- **Path-Vector Algorithms:**
 - Advertises paths to reach destinations.
 - Suitable for external routing between networks.

8.2 Border Gateway Protocol (BGP)

- **Border Gateway Protocol (BGP):**
 - Originally specified in RFC 1105 (1989).
 - Current version: BGP4 (RFC 4271, 2006).
- **Border Routers:** Connect Autonomous Systems (AS).
- **External BGP (eBGP):**
 - Operates between ASes.
 - Uses Classless Inter-Domain Routing (CIDR) prefixes.
 - Advertises:
 - * Routing prefixes for the router's own network.
 - * Prefixes from connected ASes (includes the *AS path*).
- **Internal BGP (iBGP):**
 - Operates within an AS.
 - Used to propagate BGP information internally.

8.2.1 Internet Interconnection Ecosystem

- **Internet Exchange Points (IXP):**
 - Locations where ISPs and ASes exchange traffic.
 - Examples: DE-CIX, AMS-IX, N-IX.
- **Connections between Autonomous Systems:**
 - **Peering:**
 - * ASes agree to exchange traffic without charge.
 - * Typically between ASes of similar size.
 - **Transit:**
 - * One AS pays another for interconnection services.
- **Challenges:**
 - Missing economic incentives to improve interconnection infrastructure.

8.2.2 Resilience and Security

- **Route Filtering:**
 - Best practices: filter route announcements to limit accepted routes.
 - Issues: not universally supported; mainly benefits others.
 - Example: lack of IP spoofing packet filtering.
- **Enormous Scale:**
 - Hard to attack large portions of the internet due to diversity.
 - Increasing centralization: few large players (e.g., Google, Akamai, Cloudflare).
- **Missing Data and Metrics:**
 - Lack of global transparency; commercial secrecy.
 - Difficult to evaluate network interconnection resilience.
 - Historically robust, but future concerns exist.
- **Regulation:**
 - Historically minimal.
 - Country-dependent: some regulations harm free/open internet.
 - Market concentration may require anti-trust regulation (e.g., oligopolies).
- **Future:** Moving towards **BGPsec** for improved security.

8.2.3 BGPsec: Simplified Overview

- **Definition:** BGPsec (RFC 8205, 2017) focuses on AS path validation.
- **Key Features:**
 - Utilizes **Resource PKI (RPKI)** to certify which AS is responsible for specific IP prefixes.
 - AS explicitly authorizes the next AS in the path to announce the route.
 - * BGP UPDATE message includes AS_PATH (BGP) \rightarrow BGPsec_PATH.
 - * Each BGPsec-capable AS:
 - Needs an RPKI certificate.
 - Signs prefix and BGPsec_PATH with its private key.
 - * Recipient verifies all ASes on the path were explicitly authorized.
- **Issues:**
 - Lack of incentives for ASes to support BGPsec.
 - Cryptographic overhead may impact resilience (e.g., handling unsigned announcements during recovery).
 - Preventing malicious ASes (with legitimate IPs) from injecting routes.

8.2.4 Summary

- BGP seems fragile?
 - Easily disturbed by failures or attacks.
 - No global view: Each AS has only local knowledge of its network and direct neighbors.
 - Explicit contracts/partnerships only with direct neighbors.
 - Limited technical security: Relies on trust, contracts, and decentralized motivation.
- Little is known about the internet interconnection ecosystem:
 - Lack of global view and detailed measurement data.
 - Effects of incidents are unpredictable.
- BGP seems resilient?
 - Internet functions well despite limitations.
 - Disaster recovery has been effective in the past.

8.3 DNS

- DNS lookup retrieves a resource record:
 - Types of records:
 - A: IPv4 address (domain name → IP address)
 - AAAA: IPv6 address
 - MX: Mail server responsible for the domain
 - NS: Authoritative name server for the domain
 - TXT: Textual descriptions
 - E.g., used for data that does not fit other record types
 - ... other types exist

8.3.1 DNS Attacks

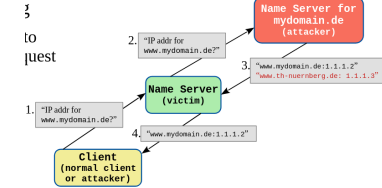
DNS (Domain Name System) attacks exploit vulnerabilities in the DNS infrastructure to misdirect users, disrupt services, or compromise sensitive data. This subsection details two common types of DNS cache poisoning attacks, the Kaminsky attack, and effective countermeasures.

- Local Attack: Malware modifies the /etc/hosts file on Unix-like systems, redirecting requests to malicious IPs.
- Compromised Name Server: Attackers exploit vulnerabilities in DNS server software to take control of legitimate name servers.
- Spoofing: Fake responses are sent to queries by forging the name server's IP address (e.g., by spoofing UDP packets).
- DNS Cache Poisoning:
 - Insert fake records in DNS responses, causing the victim to cache wrong IPs.
 - Effect: Users are redirected to malicious websites.

DNS Cache Poisoning: Insert Fake Record In this attack, an attacker inserts a fake DNS record into a legitimate DNS response. The steps are as follows:

- Step 1: A client (or attacker) queries the victim's name server for the IP address of a domain (e.g., www.mydomain.de).
- Step 2: The victim name server forwards the query to the authoritative name server for the domain.
- Step 3: The attacker, controlling the authoritative server, sends a fake response containing false DNS records (e.g., redirecting another domain to a malicious IP).
- Step 4: The victim name server caches the malicious response and serves it to the client.

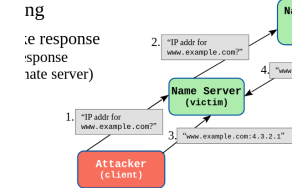
The effect is that users querying the victim name server are redirected to the wrong IP address. This attack can be mitigated by randomizing query IDs, validating responses using DNSSEC, and employing stricter validation rules.



DNS Cache Poisoning: Fake Response Faster than Legitimate Server This attack involves the attacker sending a fake DNS response faster than the legitimate authoritative server. The steps are:

- Step 1: The attacker sends a DNS query for a domain (e.g., www.example.com) to the victim's name server.
- Step 2: The victim forwards the query to the legitimate authoritative name server.
- Step 3: The attacker sends a forged response containing a malicious IP address before the legitimate server can reply.
- Step 4: The victim name server caches the fake response and returns it to the attacker.

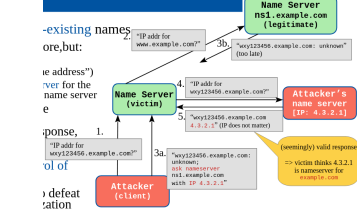
The result is that the legitimate response is ignored, and the malicious IP address is cached. Countermeasures include randomizing ports and query IDs, implementing DNSSEC, and stricter validation rules.



Kaminsky Attack The Kaminsky attack is a sophisticated DNS cache poisoning technique that exploits a vulnerability in DNS protocol design to inject malicious entries into DNS caches. The steps are as follows:

- Step 1: The attacker repeatedly queries the victim name server for non-existent subdomains (e.g., xyz123.example.com).
- Step 2: The victim name server forwards the query to the authoritative server but receives a fake response from the attacker instead.
- Step 3: The fake response contains an authoritative NS record pointing to a malicious name server controlled by the attacker.
- Step 4: The victim caches the malicious name server as authoritative for the domain.
- Step 5: All subsequent DNS queries for the domain are redirected to the attacker's malicious name server.

This attack is particularly dangerous as it allows the attacker to poison entire DNS zones, redirecting all traffic for affected domains. Mitigations include randomizing query IDs, query source ports, and deploying DNSSEC to authenticate responses.



Countermeasures Against DNS Cache Poisoning To protect against DNS cache poisoning and related attacks, the following countermeasures are commonly implemented:

- Randomized Query IDs: Use randomized 16-bit query IDs to reduce predictability.
- Randomized Source Ports: Use random UDP source ports to add an additional layer of entropy.
- DNSSEC: Deploy DNSSEC to authenticate DNS responses and ensure their integrity.
- Rate Limiting: Implement rate limiting for queries and responses to reduce the likelihood of brute-force attacks.
- Strict Validation: Enforce strict validation rules for DNS responses.
 - These countermeasures significantly increase the difficulty for attackers to successfully perform DNS cache poisoning or similar attacks.

8.3.2 DNSSEC (Domain Name System Security Extensions)

Goal of DNSSEC:

- Origin Authentication: Ensures authenticity of DNS data.
- Authenticated Denial of Existence: Proof of non-existence of records.
- Data Integrity: Verifies DNS data has not been tampered with.

History:

- 1993: Discussions and development started.
- 1997, 1999: Initial proposals (RFCs 2065, 2535, now obsolete).
- 2005: Current approach (RFCs 4033-4035).
- 2010: All root servers supported DNSSEC.
- 2011: DNSSEC adopted for .de domains.

Key Resource Records Introduced:

- DNSKEY: Public keys for signing zone data, distinguishing Key Signing Keys (KSK) and Zone Signing Keys (ZSK).
- DS: Delegation Signer; links parent to child zones.
- RRSIG: Digital signatures for verifying records.
- NSEC/NSEC3: Proof of non-existence through authenticated linked lists of DNS records.

General Procedure:



- Zone entries are signed with private ZSK.
- ZSK is signed by private KSK.
- Parent domain stores the hash of the child zone's KSK in a DS record.
- Establishes a recursive chain of trust, starting from the root zone.
- Public key of the root zone acts as the trust anchor.

Chain of Trust:

- Starts at the root zone.
- Extends through TLDs, domain zones, and subdomains.
- Requires the public key of the root zone to be securely distributed.

NSEC: Proof of Non-Existence

- Authenticates denial of existence by creating a sorted linked list of all domain names in a zone.
- Each signed entry includes the next domain name.
- Example: If hello.example.com is queried but doesn't exist, the response includes proof of its neighboring domains.
- Disadvantage: Enables zone walking, where attackers can list all (sub-)domains.

NSEC3: Mitigation of Zone Walking

- Uses hashed domain names instead of plaintext.
- Sorted hashes prevent attackers from easily enumerating domain names.
- Clients compute the hash of the requested domain and compare it to NSEC3 records.
- Dictionary attacks are still possible, but mitigated using salt and multiple hashing iterations.

Other Uses of DNSSEC DNSSEC can be utilized to enhance other security mechanisms by securely distributing associated data:

- SSH fingerprints: Distributed as per RFC 4244.
- IPsec/IKE public keys: Defined in RFC 4025.
- TLS trust anchors via DANE (RFC 6698):
 - DNS-Based Authentication of Named Entities (DANE).
 - Protects against clandestine TLS certificate exchange by attackers.
 - Enables referencing of X.509 certificates in DNSSEC-protected resource records (TLSA records).

DNSSEC in Practice

- Slow Adoption: DNSSEC uptake has been slow due to:
 - Overhead in communication, computation, delays, and administrative effort.
 - Limited operational experience despite its long history.
 - Larger resource records, increasing the potential for DoS attacks.
- No Confidentiality: DNSSEC was not designed to provide confidentiality.
- Typical Deployment:
 - The resolver is usually the nameserver of the ISP or network, not the local system.
 - Resolver verifies signatures but requires the path from the end-user system to the resolver to be secured.

8.3.3 DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH)

- DNS-over-TLS (DoT):
 - Uses TLS (instead of UDP) to transport DNS queries.
 - Provides confidentiality, integrity, and authenticity between the client and resolver (nameserver, often referred to as the "DNS provider").
 - Does not protect traffic between nameservers (e.g., between the DNS provider and the authoritative nameserver).
- DNS-over-HTTPS (DoH):
 - Similar to DoT but uses HTTPS to send queries via POST or GET requests.
 - Supports normal query formats (as used in UDP) or even JSON.
 - Particularly suited for browsers and web applications.
 - Comes with additional overhead and dependencies.

8.4 Zusammenfassung DNS

- Domain Name System (DNS)
 - Developed without security in mind
 - Vulnerabilities: Cache poisoning attacks, Kaminsky's attack
- DNSSEC
 - Provides digital signatures and keys in DNS resource records
 - Ensures integrity and authenticity of DNS responses
 - Implements NSEC/NSEC3 for proof of non-existence of DNS names
 - Can bootstrap other security mechanisms
- DNS-over-HTTPS (DoH), DNS-over-TLS (DoT)
 - Protect traffic from the end user system to the DNS resolver / nameserver
 - Both provide confidentiality, integrity, and authenticity via TLS

9 eMail and Messaging Security

9.1 eMail

9.1.1 Sending Emails: SMTP, POP3, IMAP

- Protocols:
 - Send: SMTP (Simple Mail Transfer Protocol)
 - Receive: POP3 or IMAP
- Security:
 - TLS:
 - SMTP over TLS: sender to mail server
 - POP3/IMAP over TLS: mail server to recipient
 - Between Mail Servers: Opportunistic encryption (e.g., STARTTLS)
 - Starts unprotected, upgrades to TLS if supported.
 - Falls back to plaintext if TLS fails.
 - No identity verification (certificates often not checked).

9.1.2 PGP: End-to-End Security for Emails

- Pretty Good Privacy (PGP):
 - Developed in 1991 by Phil Zimmermann.
 - Signs and encrypts data (not limited to email).
 - Standardized as OpenPGP (RFC 2440, 4880).
 - Implementations: commercial PGP, GnuPG (open source), others.
 - Trust model: Web of Trust (decentralized, "anarchic").
- End-to-End Security:
 - Hybrid encryption:
 - Symmetric key (session-specific) encrypted by recipient's public key.
 - Message encrypted with the symmetric key.
 - Independent Signature & Encryption:
 - PGP handles signing and encryption as separate operations.
 - Issue: Without binding the signature to the encryption context, attackers can:
 - Remove encryption and present the signature in plaintext (signature repudiation).
 - Forward signed plaintext to unintended recipients without the encryption.
 - Modify unsigned metadata or associate it with a different context.
 - Modern cryptographic standards prefer combining these operations into a single, secure AEAD process to ensure both confidentiality and integrity.
 - Authenticated Encryption (AE):
 - Introduced in 2001 as a partial fix (Modification Detection Code, MDC).
 - MDC enabled detection of tampering but lacked proper AE guarantees.

- Issues with early AE:
 - MDC was not strongly integrated into the cryptographic protocol.
 - Attacks possible if MDC was not always applied correctly.
- Recent standards include "real" AEAD (Authenticated Encryption with Associated Data):
 - Combines encryption and integrity into a single operation.
 - Protects both the ciphertext and associated metadata (e.g., headers).
 - Ensures signatures cannot be separated from encryption.
- Main Critiques:
 - Usability issues: easy to misuse.
 - Poor integration with many mail clients.

9.1.3 MIME, PGP/MIME, S/MIME

- MIME (Multi-purpose Internet Mail Extensions):
 - Enables email attachments and sending arbitrary data (not just text).
 - MIME header specifies transfer encoding and content type (MIME type).
- PGP/MIME:
 - Allows sending PGP-encrypted or signed ciphertext as MIME attachments.
- S/MIME (Secure MIME):
 - Alternative solution for encrypting and signing emails.
 - Based on X.509 certificates and Public Key Infrastructure (PKI).
 - Provides encryption and signing with properties similar to PGP but utilizes a different trust model.

9.1.4 E-Mail End-to-End Security: S/MIME and PGP

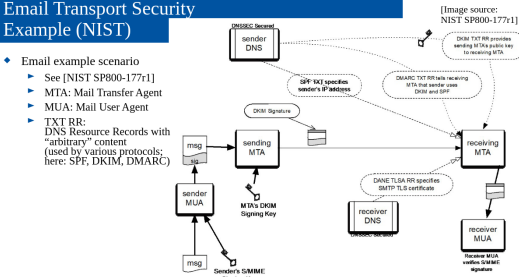
- Both S/MIME and PGP are widely used for secure email communication:
 - S/MIME: Better suited for corporate environments.
 - Companies can distribute certificates to employees.
 - IT departments can pre-configure email clients (e.g., install plugins).
 - Users simply click to encrypt/sign emails.
 - Works seamlessly within the organization's infrastructure.
 - PGP: Decentralized, popular among individuals and open-source communities.
 - No need for a centralized PKI; individuals maintain control.
 - Less user-friendly and more error-prone than S/MIME.
- Efail attack (2018): Affected both S/MIME and PGP by exploiting how email clients handled encrypted content.

9.1.5 Improving Transport Encryption for Email

Email Transport Security Example (NIST)

Email example scenario

- See (NIST SP800-177r1)
- MTA: Mail Transfer Agent
- MUA: Mail User Agent
- TXT RR: DNS Resource Records with "arbitrary" content (used by various protocols; here: SPF, DKIM, DMARC)



DANE (DNS-based Authentication of Named Entities)

- DNS entry (TLSA record) acts as a trust anchor for TLS certificate verification.
- Requires DNSSEC to ensure integrity.
- Prevents downgrade attacks by enforcing TLS if TLSA records are available.

SPF, DKIM, and DMARC

- SPF: Specifies authorized mail servers for a domain.
- DKIM: Signs outgoing emails with private keys; recipients verify signatures.
- DMARC: Combines SPF and DKIM, defines policies for authentication and reporting.

MTA-STS (MTA Strict Transport Security)

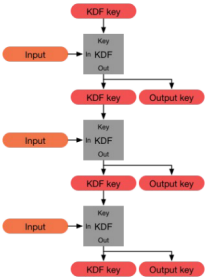
- Alternative to DANE, uses a "trust on first use" (TOFU) approach.
- Does not require DNSSEC.

9.2 Instant Messaging: Signal Protocol

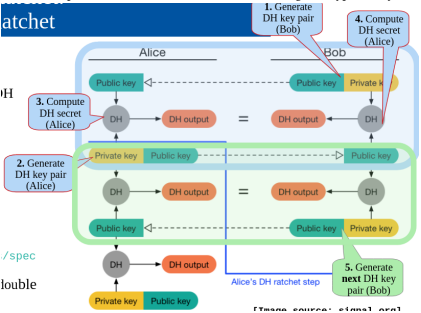
- State-of-the-art security protocol for messaging, used by apps like WhatsApp.
- History:
 - Developed by Open Whisper Systems (Moxie Marlinspike, Trevor Perrin) for TextSecure (now Signal).
 - Originates from Off-the-Record (OTR) protocol → TextSecure v1 (2013).
 - TextSecure v2 (2014): Introduced Axolotl Ratchet.
 - TextSecure v3 (2014): New cryptographic primitives and formats.
 - Signal (2016): Protocol updated and renamed.
 - Formal security analysis in 2016; published in 2017.

Key Components

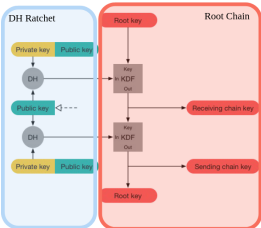
- **KDF Chain:** A Key Derivation Function (KDF) is used in a forward-only hash chain to ensure:
 - *Resilience:* Output keys appear random even if adversary controls inputs.
 - *Forward Security:* Past keys remain secure if current keys are compromised.
 - *Break-in Recovery:* Future keys remain secure with sufficient entropy.



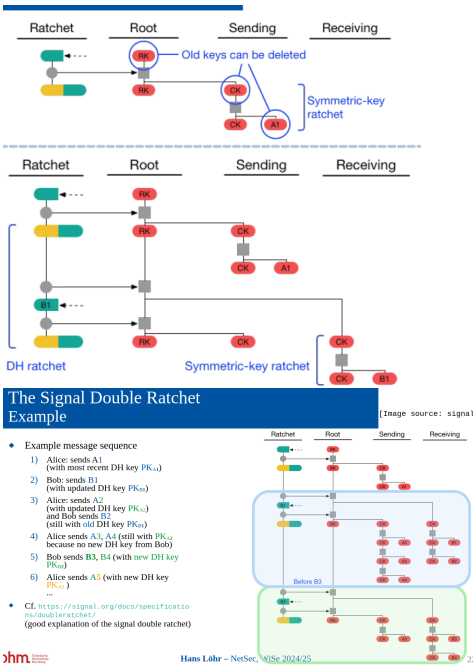
- **Double Ratchet Mechanism:** Combines two ratchets:
 - *Symmetric Key Ratchet:* Uses a KDF to generate encryption keys for messages.
 - *Diffie-Hellman (DH) Ratchet:* Updates DH key pairs for forward secrecy after every message exchange.
- **Diffie-Hellman Ratchet:**
 - Continuously performs DH key exchanges for each message.
 - Transmits DH key shares with messages.
 - Computes new shared secrets to derive message encryption keys.



- **Root Chain:** Combines the DH ratchet and symmetric key ratchet:
 - Generates a root key (RK) through DH exchanges.
 - Derives sending and receiving chain keys (CKs) for message encryption.
 - Complete ratchet step => new root key



- **Message Keys:** Each message key is derived from the corresponding chain key (CK):
 - Sender's CK encrypts messages.
 - Receiver's CK decrypts messages.
 - Ensures messages cannot be decrypted after keys are deleted.



9.3 Summary: Email and Messaging Security

- **Email Security**
 - **End-to-End Security:**
 - Practical but rarely used due to usability issues.
 - Solutions: PGP, S/MIME.
 - **Email Transport Security:**
 - Originally not designed with security in mind.
 - Improvements: DANE, SPF, DKIM, DMARC.
- **Instant Messaging Security**
 - **Modern Use Case:**
 - Instant messaging is more recent compared to email.
 - Requires tailored security solutions.
 - **Closed Ecosystems:**
 - Easier to deploy robust security.
 - High dependency on service providers, limiting competition.
 - **Signal Protocol:**
 - Designed for end-to-end security and confidentiality.
 - Based on the **double ratchet** mechanism.
 - Ensures forward secrecy; used by multiple messengers.