Network Security
TUM winter term 2012/13
Lecturers: Georg Carle & Heiko Niedermayer

Janosch Maier

February 20, 2013

Pictures originate from the slides of the NetSec course 2012/13 available at http://www.net.in.tum.de/de/lehre/ws1213/vorlesungen/network-security/

Contents

1 Introduction

Achieve Security

- By Policy
- By Architecture

Security Threads

- Violation of security goal
- Realization of thread is called attack
- Masquerade (active)
- Eavesdropping (passive)
- Loss / Modification of Information (active)
- Denial of Communication (active)
- Forgery (active)
- Sabotage / Denial of Service (active)
- Authorization Violation (active)

Security Goals:

- Confidentiality
- Data Integrity
- Accountability
- Availability
- Controlled Access

Security Methods

- Cryptographic Algorithms
- Cryptographic Protocols
- Security Supporting Mechanisms

2 Basics

2.1 Symmetric Cryptography

2.1.1 Terms

- Plaintext P
- Ciphertext C
- Key K
- Block Cipher – En-/Decrypt input of block lnegth n
- Stream Cipher – En-/Dcrypt message by key stream C = P XOR key stream

2.1.2 Attacking Cryptography

- Brute Force Attack: Plaintext has to be identifyable
- $\approx 8,37 \cdot 10^{77}$ Electrons in the universe
- Cryptoanalysis: Discover plaintext and / or key. ciphertext only vs. known / choosable plaintext / ciphertext pairs; Public key may be exploited $\Rightarrow$ breaking cryptosystem

2.1.3 One-Time-Pad

- Perfect symmetric cipher
- Random cipher stream XOR plaintext
- Key of same size as message
- No attack possible – Every plaintext could be created

2.1.4 Classification of encryption algorithms

- Substitution (simple, polgygraphic, monoalphabetic, polyalphabetic) – S-box (block wise substitution) vs. Transposition (permutation) – P-box (maximal entropy) $\Rightarrow$ Product Cipher
- Symmetric vs. Asymmetric
- Stream Cipher vs. Block Cipher

2.1.5 Feistel Ciphers

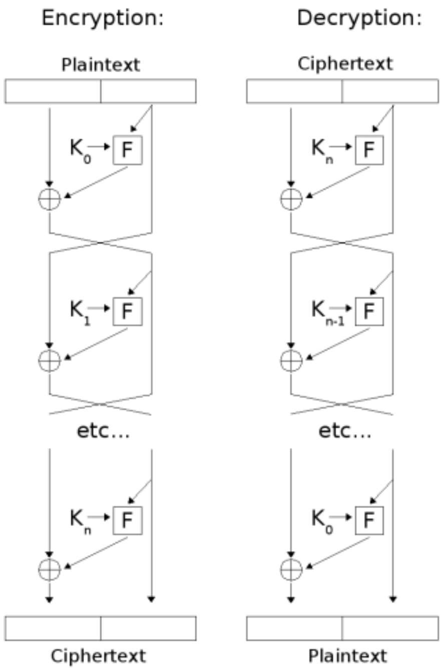

Figure 1: Feistel Cipher

2.1.6 DES

- Block Size: 64 bit
- Key Size: 56 bit
- Main Weakness: Key length
- Trippel-DES: $C = E(K_3, D(K_2, E(K_1, P)))$

2.1.7 AES

- Rijndael Algorithm
- Block Size: 128, 192 or 256 bit
- Key Size: 128, 192, 256 bit
- 10 rounds ByteSub (S-Box), ShiftRow, MixColumn, RoundKey (XOR)
- CBC or CTR possible

2.1.8 Block Ciphers

- Segment plaintext p into blocks $p_1, p_2, \ldots$ of length $p \leq b$ (block size b)
- c is a combination of $c_1, c_2, \ldots$

2.1.9 Encryption Modes

- Plaintext Input to block cipher
  - ECB (Electronic Code Book Mode) – $c_i = E(K, p_i)$
    Bit-Error $\Rightarrow$ Wrong Block
  - CBC (Cipher Block Chaining Mode) – $c_i = E(K, c_{i-1} \oplus p_i)$, Initialization Vector (IV) used for first block
    Bit-Error $\Rightarrow$ 2 Wrong Blocks
- Plaintext XORed wit output of block cipher
  - OFB (Output Feedback Mode) –
    $K_0 = IV, K_i = E(K, K_{i-1}), c_i = p_i \oplus K_i$
    Bit-Error $\Rightarrow$ Bit-Error
  - CTR (Counter Mode) – $K_i = E(K, Nonce||i), c_i = p_i \oplus K_i$
    Bit-Error $\Rightarrow$ Bit-Error

2.2 Public Key Cryptography

- K_{priv}: Private key
- K_{pub}: Public Key
- $p = D(K_{priv}, c) = D(K_{priv}, E(K_{pub}, p))$
- Trap door functions: Factorization problem (RSA), Discrete logarithm problem (Diffie-Hellmann, ElGamal)

2.2.1 Discrete Logarithm

- p is prime (e.g. 7)
- g is primitive root of $\{1, 2, ..., p-1\}$, if $\{g^a | 1 \leq a \leq (p-1)\} = \{1, 2, ..., p-1\}$
  $1 \equiv 3^6 \mod 7, 2 \equiv 3^2 \mod 7, ...$
- $c \in \{1, 2, ..., p-1\}$
- z is discrete logarithm of $c \mod p$ to the base g: $g^z \equiv c \mod p$
- runtime to calculate z is exponential in the bit-length of p

2.2.2 Diffie-Hellmann Key Exchange

- Random a: $X = g^a \mod p$
- Random b: $Y = g^b \mod p$
- $K = Y^a \mod p = X^b \mod p = g^{a \cdot b} \mod p$
- TLS/SSL uses Diffie-Hellman as Public Key Algorithm
- Signature of encrypted messages not possible

2.2.3 El Gamal

- random z: $c = m * g^{az} \mod p$
- Bob sends $g^z \mod p$ and c

2.2.4 RSA

Mathematical Background

- $\Phi(n) = m$: $m < n$, with m relatively prime to $n \Rightarrow$ greatest common divisor is 1
- p prime $\Rightarrow \Phi(p) = p - 1$
- p, q prime, $n = p \times q \Rightarrow \Phi(n) = (p - 1) \times (q - 1)$
- Euler: $m^{\Phi(n)} \equiv 1 \mod n$

Key-Generation

- Choose p, q large primes
- $n = p \times q$
- Choose e: $1 < e < \Phi(n)$, e relatively prime to $\Phi(n)$
- Choose d: $e \times d \equiv 1 \mod \Phi(n)$
- Public Key: (n, e)
- Private Key: d

RSA function

- Encryption: $C \equiv M^e \mod n$
- Decryption: $M \equiv C^d \mod n$

Using RSA

- Asymmetric Cryptography slower than symmetric Cryptography
- Padding used against certain attack scheme (OAEP)
- Difficulty: $n = p \times q$
- 2084 bit key length reccomended „if you want to protect your data for 20 years“ (Schneier)

2.2.5 Digital Signatures

- Create Hash Value $h(M)$
- Encrypt with private Key: $E_{K\_priv}(h(M))$
- Everybody can check: $D_{K\_pub}(E_{K\_priv}(h(M)))$

2.2.6 Elliptic Curve Cryptography (ECC)

- Elliptic Curve: $y^2 = x^3 + ax + b$
- Multiplication: $Q = nP = P + P + P + ... + P$
- Find n based on Q, P: Elliptic Curve's Discrete Logarithm Problem (ECDLP) – Believed to be harder than DLog

2.2.7 Key Sizes (Informal Comparison)

Symmetric	RSA	ECC
74	1024	139
256	15360	512

General Reccomendation difficult. (See 2.2.4)

2.2.8 Encryption and Signature

- Encryption after Signature ⇒ Attacker can decrpyt, re-encrypt replace receiver ⇒ Signature must include sender, receiver, ...
- Signature after Encryption ⇒ Attacker can strip signature, replace with his own. receiver cannot determine correct sender ⇒ Sign plaintext / message must include sender, receiver, ...

2.3 Cryptographic Hash Functions

2.3.1 Motivation

Not only Error-Detection-Code (e.g. CRC – No cryptographic hash function) needed, but Modification-Detection-Code (MDC)

2.3.2 Definition

- Hash function
  - Compression: $h : \{0,1\}^* \to \{0,1\}^n$
  - Ease of computation: $h(x)$ is easy to compute
- One-way function
  - Given y, x is hard to compute with $h(x) = y$
- Cryptographic hash function (Hash Function h with additional properties)
  - One-way function: Pre-image resistance
  - 2^{nd} pre-image resistance: Given x: x' with $h(x) = h(x')$ is infeasible to find (np-problem)
  - Collision resistance: Pair with same hash is infeasible to find
  - Random oracle property: Infeasible to distinguiash h(m) from random n-bit value

2.3.3 Applications

1. Data Integrity
   - Public Key cryptography: Hash need to be signed
   - Symmetric cryptography: Message authentication code (MAC)
     - Hash function that needs plaintext + a key as input → $h_k(x)$
     - Computation-resistance: Infeasible to compute a text-MAC pair without knowledge of the key
     - Key-non-recovery: Knowing a text-MAC pair it is impossible to recover the key
2. Pseudy-random number generation
   - uniform distribution
   - $b_0 = seed$, $b_{i+1} = h(b_i|seed)$
3. Encryption (OFB)
   - Keystream: $k_0 = h(K_{A,B}|IV), k_{i+1} = h(K_{A,B}|k_i)$
4. Authentication (challenge-response)
   - Alice → Bob: r_A
   - Bob → Allice: $h(K_{A,B}, r_A)$
   - e.g. HTTP digest
5. Authentication wit One-Time Passwords (OTP)
   - Initial setup:
     - A → B: r_a
     - B → A: $(PW_n = H^N(r_a, password_B), N)$
   - Authentication:
     - A → B: $N - 1$
     - B → A: $PW_{n-1} = H^{N-1}(r_A, password_B)$
     - If $h(PW_{n-1}) = PW_n$, B is authenticated. $(N - 1, PW_{n-1})$ are used for next authentication.
6. Error detection: possible but expensive

2.3.4 Stucture of Cryptographic Hash Functions

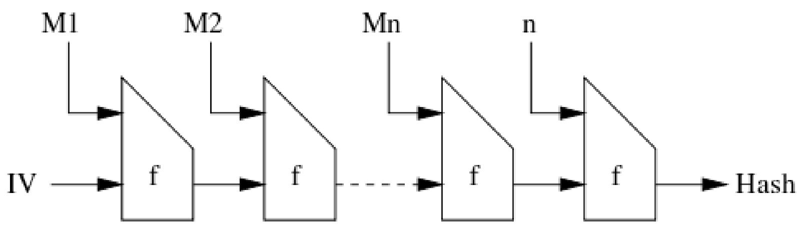

Figure 2: Merkle Damgård Construction

- H is collision resistant if f is collision resistant
- Hash length should be at least 160bit due to the birthday attack

2.3.5 SHA-1

- Chaining value with registers A - E
- 4 Rounds (each using another fuction f) with 20 steps earch
- SHA-1 value is final chaining value

2.3.6 Bithday Phenomenon

How many people needed, that possibility of two people with same birthday greater 0.5 $\Rightarrow$ 23 people.

$$P(n,k) = 1 - \frac{n!}{(n-k)! \cdot n^k} > 1 - e^{\frac{-k\cdot(k-1)}{2n}} \tag{1}$$

$$k \approx 1.18\sqrt{n} \tag{2}$$

- Only $\mathcal{O}(\sqrt{n})$ tries needed to get a collision
- Yuval's square root attack. Produce variation of messages by adding <space>. Effort to find a collision much less than standard approach

2.3.7 CBC-MACs

- Encrypt message in CBC mode, take last ciphertext block as MAC
- Already signed by shared secret key K (sender or receiver)
- Block cipher needed (DES, AES, ...)
- Must not use same key as encryption! (MAC will be equal to last cipher text block)
- AES-CBC-MAC secure, fast

2.3.8 MAC and Cryptographic Hash Function

- Mix secret key K wit input ande compute hash value
- HMAC: $H(K \oplus opad \mid H(K \oplus ipad \mid m))$

2.3.9 SHA-3 and Skein

- Winner of NIST SHA-3 competition: Keccak
- Skein
  - 512 (default), 1024 (conservative), 256 (low memory)
  - Block cipher Threefish
  - Unique Block Iteration (UBI) as chaining mode
  - Optional Arguments possible
  - Tree Hashing (paraller CPUs)
  - Skein-MAC: HMAC possible, Skein with optional argument “key”

2.3.10 Integrity Check and Digital Signature

- Integrity Check with hash function / MAC
  - Shared key, message, MAC (based on hash function or symmetric cipher)
  - Message and MAC send to receiver
  - possible MACs: HMAC, CBC-MAC, Encrypt(k, h(m))
- Digital Signature
  - Sender signs message using private key and hash function
  - Receiver compares h(m) and h(m) that was signed by sender, using sender's public key

2.4 Random Number Generation for Cryptographic Protocols

Attacker must not be able to reproduce key generation process

2.4.1 Random Number Generator

- Device or Algorithm which outputs a sequence of statistically independent and unbiased binary digits

2.4.2 Entropy

- Measurement for randomness
- Perfect entropy: key of length n bits has n bits entropy (All outputs equally probable)
- Human passwords usuallye have much lower entropy

2.4.3 Pseudo-Random Number Generator

- Pseudo Random Bit Generator (PRBG) is deterministc algorithm with outputs a pseudo random bit sequence of length $m >> k$ given a seed of length k as input
- Output not random Only 2^k sequences for length m possible (not 2^m)
- Generation of long random number too expensive, therefore create small random number and use PRBG
- k has to be big enough to make brute-force over all seed infeasible
- Output of PRBG should be statistically indistinguishable from random sequences
- Output of PRBG should be unpredictable if seed unknown
- Definition: Pass all polynomial-time statistical tests – No polynomial time statistical algorithms can distinguish between PRBG output and random sequence

- Definition: Pass the next-bit test – No polynomial time algorithm that can predict the rest of the sequence if beginning of sequence is given as input $\Leftrightarrow$ Passes all Polynomial time statistical test $\Rightarrow$ Cryptographically secure pseudo-random bit generator (CSPRBG)

2.4.4 Hardware-Based Random Number Generation

- Randomness based on physical phenomena: Radioactive decay, sound from microphone, etc.
- Should be in enclosed device

2.4.5 Software-Based Random Number Generation

- Based on: system clock, user input, system load, etc.
- Ideally multiple sources of randomness
- Usuallye used to set seed of PRNG

2.4.6 De-skewing

- Random generator with biased bits. Probability for 1: $p \neq 0.5$, for 0: $1-p$
- Group outpit into pairs, discard 11 and 00
- 10 $\Rightarrow$ 1 and 01 $\Rightarrow$ 0 is an unbiased generator

2.4.7 Statistical Test

- Monobit: Equally many 1s and 0s
- Serial Test: Equally many 00, 01, 10, 11 pairs?
- Runs Test: Number of runs (0 sequences or 1 sequences) expected?
- Autocorrelation Test: Correlation between sequence and shifted versions of it?
- Maurer's Universal Test: Sequence compressed?

2.4.8 Examples for PRNGs

- Blum Blum Shub
- Symmetric Encryption: Output of block cipher (OFB or CTR mode), Output of stream cipher (RC4)
- Based on Hash function: X_0 = seed, $X_{i+1} = H(X_i|seed)$

3 Cryptographic Protocols for Encryption, Authentication and Key Establishment

3.1 Introduction

3.1.1 Cryptographic Protocol

- Series of steps / message exchanges between entities to achieve specific security objective
- Properties of a protocol (in general):
  - Everyone knows all steps in advance and agrees to follow
  - Protocol is unambiguous (every step is well defined, no misunderstanding possible) and complete (response for every action)
- Additional property of cryptographic protocol
  - Not possible to do / learn more, than protocol specified

3.1.2 Application of Cryptographic Protocols

- Key Establishment
- Data Origin Authentication:
  - Message is originated by particular entity and has not been altered (Implies data integrity)
- Entity Authentication
  - Enables communication partners to verify their peers
  - Basis for most other security goals
  - Accomplished by
    * Knowledge (Password)
    * Posession (Key)
    * Immutable characteristcis (Fingerprint)
    * Location (Bank agent)
    * Delegation of Authenticity (Web of Trust)
  - Cryptographic Protocols, as direct verification is difficult / insecure
- Authenticated key establishment
- Data integrity
  - Message has not ben altered
  - Basis for most other security goals
- Confidentiality
- Secret sharing
- Key escrow

- Zero-knowledge proof
- Blind sigantures
- Secure elections
- Electronic money

3.2 The Secure Channel

3.2.1 Properties

- PDUs (Protocol Data Units) created from messages (Service Data Units)
- Message loss (or deletion) possible
- Message numbering, Authentication, Enrcyption
- Encryption before MAC creation – Don't caste CPU time, when MAC mismatches
- Encryption after MAC creation – MAC also protected (Authenticate what you mean, not what you say)

3.2.2 Authentication / Encryption

- HMAC-SHA-256 $a_i := MAC(i||x_i||m_i)$ (Message number, message, authentication data of fixed size)
- AES-CTR-256
- Frame: $i, E(m_i||a_i)$
- 4 keys (Enrcyption, Authentication in both directions)

3.2.3 Design Criteria

- Relay protection window (reordering of packages on transit)
- Negotiation of crypto algorithms
- Identifier for connection

3.3 Authentication and Key Establishment Protocols

3.3.1 Introduction

- Problems:
  - Generation of new session key
  - Cryptographic algorithms
  - Verification of partners
  - (Mutual) Entity authentication with / without key establishment
- Diffie-Hellman

- No Authentication (Man-in-the-middel attack possible)

- Static Approach
  - Keys / Algorithms personally agreed on
  - Simple and good authentication
  - Manual process required, not scaling, no session key
  - E.g. GSM: Long-term secred key stored in SIM card

- Trusted Third Parties (TTP)
  - Secure channel to TTP who always behaves honestly
  - If compromised, attacker controls the whole network
  - Online (KDC) or Offline (CA) possible
  - Key Distribution Centers (KDC)
    * TTP that shares secrets with all entities
    * Problem: KDC can monitor all authentication, session keys and is single-point-of-failure
  - Public Key Infrastructures (PKI)
    * Certificate Authority (CA) is TTP, every entity knows CAs public key
    * CA signs Certificates with his private keys

- Attacks
  - Replay Attack
  - Man-in-the-Middel Attack

3.3.2 Key Distribution Centers (KDC)

Needham-Schroeder Protocol

- Needham-Schroeder Symmetric Key Protocol
  1. $A \rightarrow AS : (A, B, r_1)$
  2. $AS \rightarrow A : \{r_1, K_{A,B}, B, Ticket_{A,B}\}_{K_{AS,A}}$ with $Ticket_{A,B} = \{K_{A,B}, A\}_{K_{AS,B}}$
  3. $A \rightarrow B : (Ticket_{A,B})$
  4. $B \rightarrow A : \{r_2\}_{K_{A,B}}$
  5. $A \rightarrow B : \{r_2 - 1\}_{K_{A,B}}$

- Needham-Schroeder Symmetric Key Protocol with ticket reuse
  1. $A \rightarrow B : (Ticket_{A,B}, \{r_2\}_{K_{A,B}})$
  2. $B \rightarrow A : \{r_3, r_2 - 1\}_{K_{A,B}}$
  3. $A \rightarrow B : \{r_3 - 1\}_{K_{A,B}}$

  - Problem: If one session key is known a Eve can use a replay attack to sucessfully impersonate Alice ($\rightarrow$ Timestamps in Kerberos)

- Needham-Schroeder Public Key Protocol
  1. $A \rightarrow AS : (A, B)$
  2. $AS \rightarrow A : \{K_{B-pub}, B\}_{K_{AS-priv}}$
  3. $A \rightarrow B : \{r_A, A\}_{K_{B-pub}}$
  4. $B \rightarrow AS : (B, A)$
  5. $AS \rightarrow B : \{K_{A-pub}, A\}_{K_{AS-priv}}$
  6. $B \rightarrow A : \{r_A, r_B\}_{K_{A-pub}}$
  7. $A \rightarrow B : \{r_B\}_{K_{B-pub}}$

  - If A initates a session with M, M can celay the messages to Bob and impersonate A.
  - Fix in message 6: $B \rightarrow A : \{r_A, r_B, B\}_{K_{A-pub}}$

Kerberos

- Design goals: Security, Reliability, Transparency, Scalability
- Kerberos V. 4
  1. A $\rightarrow$ AS (Authentication Server): Request Ticket granting ticket (TGT)
  2. AS $\rightarrow$ A: TGT, Session Key $K_{A,TGS}$
  3. A $\rightarrow$ TGS (Ticket Granting Server): Reqest Service granting ticket (SGT)
  4. TGS $\rightarrow$ A: SGT, Session Key $k_{A,S1}$
  5. A $\rightarrow$ S1: Request Service
  6. S1 $\rightarrow$ A: Service Authenticator

  - Inter-realm authentication: TGS of different realms share a secret key, TGS of another realm requires a ticket of TGS of local realm
  - Advantages: Simple, High performance (hard coding of parameters)
  - Disadvantages: Limitations (hard coding, ticket lifetime, only DES, only IPv4)
  - Misuse of Propagating Cipher Block Chaining (Damages all remaining blocks, when one bit flipped), Checksum (probably unsecure) used together with PCBC
- Kerberos V. 5
  - ASN.1 syntax
  - Longer Ticket lifetimes
  - Invalidation / re-validation of tickets possible
  - Delegation of rights (inclusion of different addresses / no address in ticket)
  - Master key hashed from password and realm

  - Better encryption algorithms
  - Pre-authentication (timestamp in message 1 encrypted with Master Key) to avoid active attacks
- Kerberos Usage
  - Often application server perform Kerberos exchange in behalf of the user
  - Application servers often use PAM (Pluggable Authentication Modules) for Kerberos support
  - Sigle-Sign-On (Password only entered once)
  - Realiability only implemented with backup KDCs
  - Synchronizes clocks needed (Random nonces?)
  - Dictionary attacks on passwords possible (DH keyexchange?)

3.3.3 Public Key Infrastructures (PKI)

- Each entity has public / private keypair. A certificate binds an entity's name to its public key. A CA assures the certificate by signing it with its private key
- X.509 Public Key Certificates
  - Version
  - Certificate Serial Number
  - Signature Algorithm
  - Issuer Name
  - Validity Period
  - Subject Name
  - Subject's Public Key Info (including public key)
  - Issuer Unique ID (V2)
  - Subject Unique ID (V2)
  - Extensions (V3)
  - Signature
- Certificate chains usually in certification hierachy
- Revocating a certificate
  - Information not valid anymore
  - Private key cannot be used anymore (password forgotten / disk failure)
  - Private key (partially) revealed
  - Parametrs of certificate inaqequate (Key length insufficient)
  - Problem: Certificate of CA is compromised. All issued certificates have to be revoked

  ⇒ Certificate Revocation Lists (CRL) can be accessed viae Onlin Certificate Status Protocol (OCSP) – Sloc / Expensive operation

3.3.4 Building Blocks of key exchanges protocols

Forward Secrecy

- Protocol provides perfect forward secrecy (FPS) if compromise of long-terk key does not compromise session keys of previous protocol runs $\Rightarrow$ DH key exchange

DoS protection with cookies

- DDoS flood of secure channel establishment requests
- Computation and stored information can easily lead to denial of service
- Solution: Verify, that initiator can receive messages send to claimed source of request
- Send cookie e.g. Hash($N_a | Address_A | < secret >$) to source of request
- Request has to be sent again with cookie
- Only legitimate initator or host on path can send cookie

4 IPSec

4.1 Introduction

IP does not meet any security obejctives.

4.1.1 IPSec Security Objectives

- Data origin authentication
- Connectionless data integrity
- Relay protection
- Confidentiality
- Packages migth be dropped based on policies

4.2 The IPSec Architecture

4.2.1 Overview

- Protocols: Internet Security Association Key Management Protocol (ISAKAPM), Internet Key Exchange (IKE), IKEv2
- Secure Channel: Authentication Header (AH) – data integrity, or Encapsulating Security Payload (ESP) – data integrity + confidentiality
  - AH: IP Header – AH Header – Data
  - ESP: IP Header – ESP Header – Data – ESP Trailer
- Key Management / Security Association (SA) Setup
  - ISAKMP: No authentication protocol, but only package format
  - IKE: Authentication and key exchange protocol – Estblisch IKE SA, then IPSec SA
  - IKEv2: Reduced complexity compared to IKE

4.2.2 IPSec Replay Protection

- Sequence number intialized to 0 on creation of SA
- Sequence number 32 bits long and increased with every package
- Minimum window size 32 (64 recommended)
- After authentication verification: If package in window: accept, if right of window: accept and advance window

4.2.3 IPSec security protocol modes

- Transport mode
  - Usable if cryptographic endpoint is communication endpoint
- Tunnel mode
  - Used if at least one cryptographic endpoint is not communication endpoint
  - Tunnel IP package for communicating entity through IPSec tunnel (→ encapsulated IP header)

4.2.4 IP Security Policies and the Security Policy Database (SPD)

- Traffic selector (TS)
  - IP source address
  - IP destination address
  - Name
  - Protocol (may not be accessible with ESP)
- Policy definition
  - Package defined by TS
  - Required security attributes: security Protocol (AH/ESP), protocol mode (transport/tunnel), other parameters
  - Action: discard, secure, bypass
- Policies are stored in the SPD
- IPSec protection for certain application possible based on port number in TS

4.2.5 Security associations (SA) and the SA Database (SAD)

- SA is simplex connection describing how to process traffic
- Identified by security parameter index (SPI) specified while SA creation (Used for header creation / map traffic to SA)
- Connected to either AH or ESP
- 2 SAs needed for bidirectional traffic
- Stored in SAD
  - IP source address
  - IP destination address
  - Security protocol identifier (AH/ESP)
  - Sequence number counter
  - AH algorithm and key / ESP algorithm, key, mode, IV
  - SA lifetime
  - IPSec protocol mode (transport/tunnel)
  - Additional items

4.2.6 Package Processing

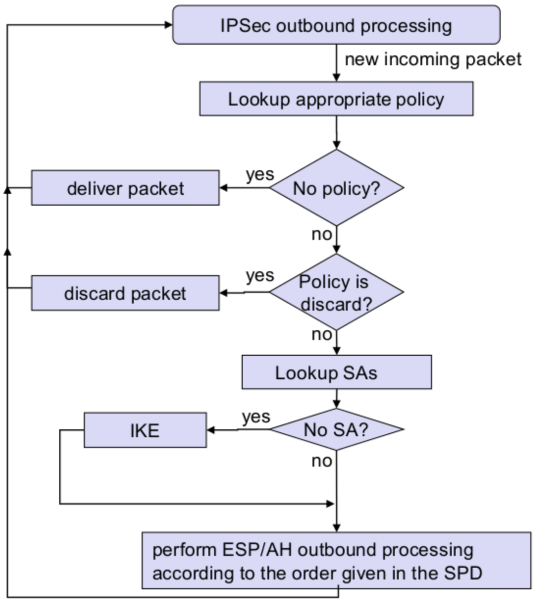

Figure 3: Outgoing IPSec package

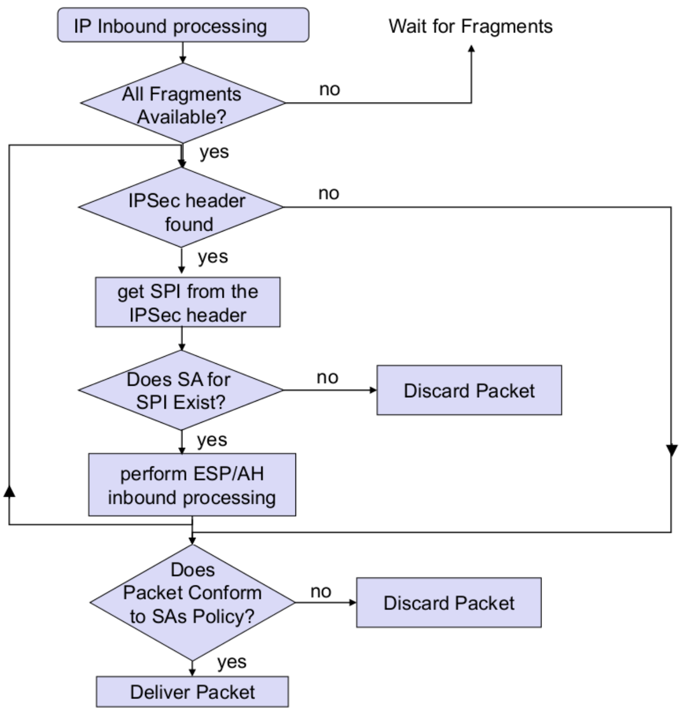

Figure 4: Incoming IPSec package

4.2.7 Implementation alternatives

- spdadd fec0::1 fec0::2 any -P out ipsec esp/transport//require ; (IPv6 – ESP Transport)
- spdadd fec0::1 fec0::2 any -P out ipsec esp/transport//require ah/transport//require ; (ESP Transport – AH Transport)
- spdadd 10.0.1.0/24 10.0.2.0/24 any -P out ipsec esp/tunnel/172.16.0.1-172.16.0.2/require ; (VPN – ESP Tunnel)

Rules have to be applied with in parameter for incoming packages as well.

4.3 IPSec Security Protocols

4.3.1 Encapsulating Security Payload (ESP)

- Security parameter index (SPI) – Chosen by receiving side, as needed for SA
- Sequence number
- IV
- Protected data
- Padding
- Padding length
- Next Header (Tunnel: IP, Transport: TCP/UDP)
- Authentication data (MAC)

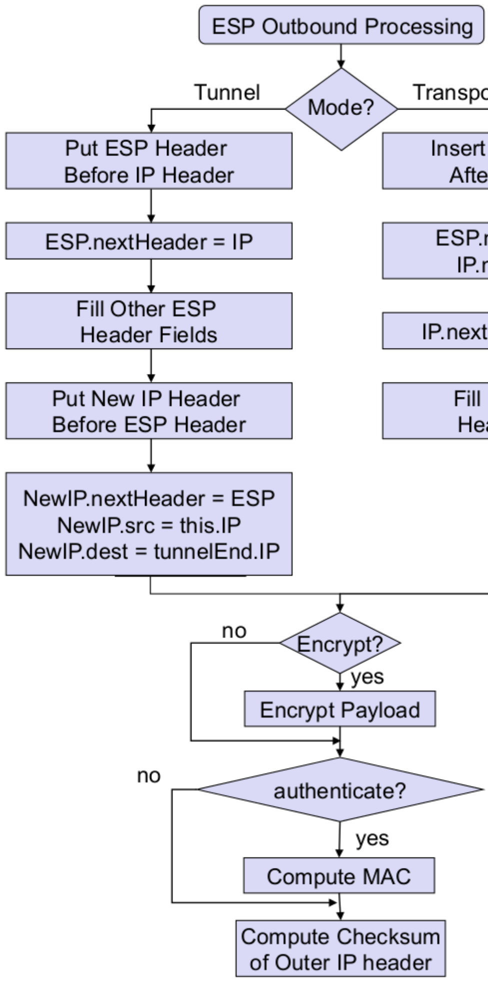

Figure 5: ESP outbound processing

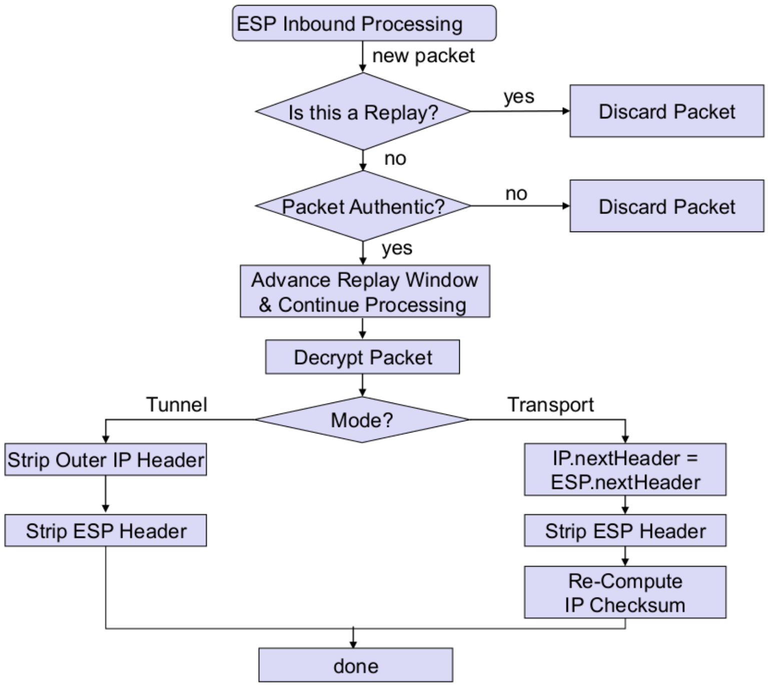

Figure 6: ESP inbound processing

4.3.2 Authentication Header (AH)

- IP Header (next Header set to 51 = AH) – Mutable fields cannot be protected by AH
- SPI
- Sequence number
- Authentication data

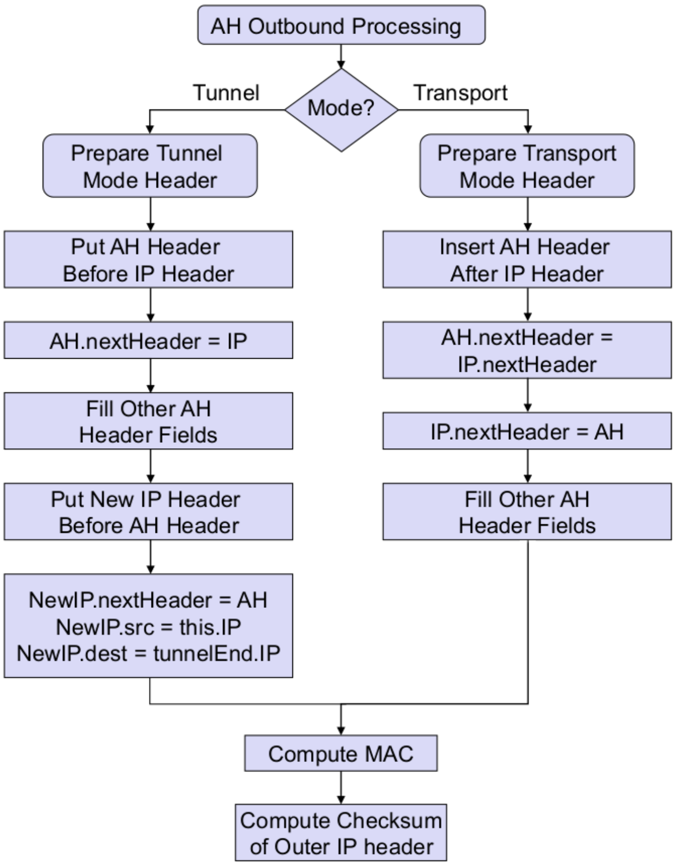

Figure 7: AH outbound processing

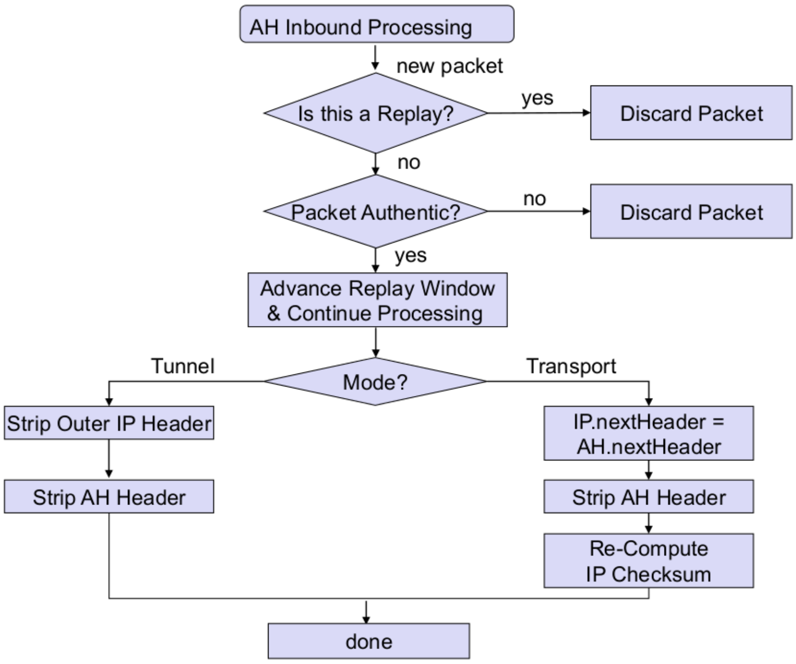

Figure 8: AH inbound processing

4.3.3 Sample Crypto Protocols

- AES-CBC
- AES-CTR
- HMAC-SHA1-96
- AES-XCBC-MAC-96

4.4 Entity Authentication and Key Establishment with the Internet Key Exchange Version 2 (IKEv2)

4.4.1 Introduction

- Establishment of SAs (manual difficult and not scalable) dynamically achieved with IKE
- Version 1 poorly described and eventually insecure
- ⇒ Version 2 as desribed here IKEv2 is not interoperable with IKEv1 but can be run on the same port
- IKEv2 provides
  - Mutual authentication

    - DoS mitigation using Cookies
    - Remote address aquisition (for VPNs)

- Latency 2 round-trips (4 messages)

4.4.2 Protocol Exchanges

- Always pairs of messages (= exchanges); IKE_SA_INIT and IKE_AUTH in this order start SA session
- UDP connection – requester ist responsible to ensure reliability
- IKE_SA_INIT: Negotiates SA paremeters, sends nonces, DH-values
    - KEYSEED generated from DH-values
    - 2 keys for integrity protection
    - 2 keys for encryption
    - 1 key for deriving CHILD_SAs
    - 2 keys for generating AUTH payloads in IKE_AUTH
- IKE_AUTH: Authenticates init, creates first CHILD_SA (= SA) piggibacked
    - Authentication achieved by using either public-key or long time shared secret to generate AUTH payload
- CREATE_CHILD_SA: Creates another CHILD_SA, rekeying
    - Maximum 4 SAs per request
- INFORMATIONAL: Keep-alive, delete SA, reporting errors, …

4.4.3 Flood Protection

- Switching protocol if too many half-open IKE_SA_INIT connections
- Same approach as described in chapter 3.3.4

4.4.4 Traffic Selector (TS) Negotiation

- Send parts of SPD to peers
- Consistency check
- TS includes Address range, Port range, IP protocol ID

4.4.5 Negotiation of Security Associations

- SA payload consits of proposals (set of security protocols with algorithms)
- Proposals ordered by preference
- Proposal contains transform (algorithms) + attributes if needed

5 X.509

5.1 Comprehensive overview of X.509 for the WWW

5.1.1 Root stores

- Trusted tfo isso certificates correctly
- Every application using X.509 has root store (Operating Systems / Browsers)
- Root store vendors have their own rules how to add a CA to the root store

5.1.2 Intermediate Certificates

- Delegate Signing Authority
- Protect main root certificate (Keep root Cert offline, Replace intermediate CA when compromised)
- Same signing authority as root certs
- SSL proxies problematic, when sub-CAs used there
- Cross-signing: Breaks root store model in WWW, Signing authority can sign even though not in the root store (although usefull in business-to-business models)
- Weakest (sub-)CA determines strength of whole PKI
- DNS path restrictions possible (But need to be used by CA)

5.1.3 Certificate Issuance

- Domain Validation (by email)
- Organisational Validation (rare)
- Extended Validation (rare)

5.1.4 Certificate Revocation

- Certificate withdrawn in some cases (key compromised, CA compromised, Service no longer operating)
- Certificate Revocation List (CRL)
  - List of revoked certivicates
  - Should be maintained by CAs
  - Should be downloaded by clients and checked before connection to server
  - Problems:
    * Intermediate certs checked as well $\Rightarrow$ heavy load
    * Time between updates
    * Large CRLs unsuitable

    * Man-in-the-middle attacks

- Online Certificate Status Protoco (OCSP)
  - Query-response model
  - Lookup of a certificate
  - Signed response: good / revoked / unknown
  - Problems:
    * Latency
    * unknown ?? ⇒ either accepted or denied
    * High availability of servers needed
    * Man-in-the-middle attacks
    * Privacy!
- OCSP stapling
  - Server staples proof that certificate is still valid onto certificate during SSL/TLS handshake
  - Solves problems with privacy / heavy load

5.2 Recent Results

Not relevant for exam

- Several PKI weaknesses
- Only 18% of certificates fully verifiable
- Several certificates issued for ples or localhost
- 3 categories of certificate quality (good, acceptable, poor)

5.3 Proposals to enhance or replace X.509

- No good way to solve all issues at once
- The following ideas are not yet really implemented but only conecpts

5.3.1 Hardening Certification

- Extended Validation
- State-issued documents before certification
- Certificates carry sepecial OID for browser to show 'green bar'
- Rarely bought (expensive) → Should be CA standard / requirement for Certification

5.3.2 Pinning Information

- Client stores certificate information on connection
- Warns user if certificate changes

\+ Raises barriers for attackers

– No defencen on first connect / Legitimate certificate changes

- Pinning information shipped with client (Chrome) possible

5.3.3 Use of DNSSEC

- DANE: DNS-based authentication of named entities
- Integrity protection and authentication on DNS queries / responses
- DANE adds support for e.g. certificates stored within DNS records (TLSA record)

\+ STron reassurance of certificates

– DNS operators are PKI operatores / Influence of states

5.3.4 Notary Principle

- Double check of certificates with notaries

\+ Good detection unless attacker controls paths to all notaries

– Privacy

5.3.5 Public Logs

- Sovereign Keys: Sites cross-sign their certificates. Key is published in public log
- Public Log: Store info about certification in the public log
- Detection of rogue CAs issuing keys

\+ No CA support needed, Evidence can be based on DANE, CAs, …

– Monitoring needed, Entries not space efficient, Key loss can lead to loss of domain

5.3.6 Certificate Transparency

- Proof of certification in public log

\+ Protoctes against rogue CAs, Proofs in $\mathcal{O}(\log n)$

– Monitoring needed, Monitors need full logs, Storage linear

5.3.7 Summary

- Nothing can solve all issues
- Vendor support needed (DANE, Certificate Transparency)
- Pinning works well, but no scaling

6 Security Protocols of the Data Link Layer

6.1 Point-to-Point Protocol (PPP)

6.1.1 Tasks

- WAN connections between routers / Dial up connections using (DSL) modems
- Protocols: Serial Line IP (SLIP), PPP (Layer-2 frame), Link Control Protocol (LCP) for connection establishment
- Entity authentication (Password Authentication Protocol – PAP, Challange Handshake Authentication Protocol – CHAP, Extensible Authentication Protoco – EAP)
- Enrcyption (but no key management, therefore practically useless)

6.1.2 PPP Reality Check

- Lack of keay management led to propreatary protocols e.g. Microsoft's MSCHAP
  - Poor hash function
  - Widely used (e.g. with Radius and WPA2)
  - TLS-tunnel or Certificates as comping methods

6.2 Extensible Authentication Protocol (EAP)

- PPP authentication protocol with multiple authentication methods
- Framework for authentication e.g. EAP-MD5, EAP-TLS

6.3 IEEE 802.1x

- Authentication standard in networks
- Uncontrolled port for authentication
- Controlled port for authenticated devices
- Authentication initiated by client or authenticator possible

6.3.1 Roles

- Supplicant reqersts access to the controlled port
- Authenticator demands supplicant to authenticate itself
- Authentication Server checks credentials for authenticator

6.3.2 Protocols

- EAP used for device authentication. PPP EAP TLS recommended
- Authenticator aund Authentication Server communicate via AAA protocol
- Exchange of EAP messages using EAP over LAN (EAPoL)

6.4 AAA Protocols

- Generic architecture for Authentication, Authorization, Accounting
- Delegate tasks to dedicated servers
- AAA data not stored on access points
- Database can be reused

6.4.1 Back-End and Front-End Protocols

- Back-end between Authenticator and Authentication Server (AS)
  - Radius
  - Diameter
- Front-end between Supplicant aund Authenticator
  - PPP
  - LAN, EAPoL
  - WLAN, WEP

6.5 Wireless LAN Security

6.5.1 IEEE 802.11

- IEEE 802.11 standardizes medium acces controll (MAC) and characteristics of WLAN
- 2.4 GHz band
- MAC operations controlled by access point or between independend stations
- Security Services
  - Entity Authentication service
  - WEP for Confidentiality, Authentication, Integrity (using RC4, CRC checksum)

6.5.2 Wired Equivalent Privacy (WEP)

- RC4 stream cipher in (OFB mode), generates pseudo-random sequence XORed with plaintext
  - Keylength up to 2048 bit
  - Known-Plain-Text attacks if IV is reused (generation of keystream ⇒ decryption possible)
  - First bits leak information of key ⇒ Trunkate first 1024 bytes
  - Keystream reused after $\sim 2^{12}$ frames (Birthday Paradox)
- No keymanagement ⇒ Keys rarely changed
- Keylength specified with 40 bits ⇒ Too short
- Ciclyc redundancy code (CRC) additive and no cryptographic hash function ⇒ Integrity insecure
- Weakness in RC4 key scheduling

6.5.3 Access Control with 802.1X

- Recommends EAP with AAA
- Not solving any WEP problems

6.5.4 Wi-Fi Protected Access (WPA)

- Snapshot of 802.11i
- Short-term solution to patch WEP
- Authentication with 208.1X standard (Supplicant, Authenticator, AS)
- Data Privacy (Encryption) using Temporal Key Integrity Protocol (TKIP), rapid re-keying to patch WEP, Per-packet key for WEP encryption, Workaround for WEP
- Data integrity: Message Integrity Code (MIC)
- Authenticator in Stand-Alone mode (serving as AS) or Pass-through mode

6.5.5 WPA2

- Counter-Mode / CBC-MAC Protocol (CCMP) for Confidentiality, Data Integrity and Replay protection using AES-CTR, AES-CBC-MAC (with different key)

7 The OpenPGP Web of Trust

7.1 Concept of a Web of Trust (WoT)

- Everyone can sign anyone
- Decentralized
- CA is just very active user

7.1.1 Directed Graph

- Communities
- Linked communities
- Isolated islands

7.1.2 Certification

- Issue certificate = sign(User ID, Public Key)
- Network of keyservers (Synchronizing Keyservers protocol – SKS – Complete history in network)
- Owner Trust: I trust this person to properly identify a person before signing (privately stored)
- Public Key Trust: I have checked that this is a person's public key (stored with signature)
- Valiy keys: Path length ≤ 5, 1 path with full trust or 3 paths with marginal trust
- Best use in "local neighbourhood"

7.2 Investigation of the current OpenPGP WoT

7.2.1 Requierements for good WoT

- Certification paths between many/all keys
- Short certification paths
- Redundand parts
- Robustness
- Captures social relations

7.2.2 Dataset

- 2.7 million keys (570.000 revoked)
- 325.000 keys with 817.000 signatures

7.2.3 Makrostructure

- How can users profit from WoT?
- Strongly Connected Components – SCC ($\geq$ 1 signature chain between any two keys)
- Largest SCC (LSCC) only 45.000 keys
- 240,000 SCCs in total. Most sigle nodes, 10.000 pairs
- Prominent CAs: Heise, CAcert, DFN-Verein
- Ration edges/nodes in LSCC 9.85, 2.51 in rest of WoT

7.2.4 Usefulness (of LSCC)

- 2-neighbourhood mostly $\sim$ 100 keys
- 5-neighbourhood 50% $\leq$ 22.000 keys
- Indegree: Key highly verifyable
- Outdegree: Many redundant paths vor verification
- Mutual Verification: Increases In-/Outdegree
- Many keys have In-/Outdegree of 1 or 2
- Only 50% mutual signatures
- Redundand paths too rare

7.2.5 Robustness (of LSCC)

- Random Invalidation of keys (expires, ...) – Very robust, Remove $\frac{1}{3}$ of all keys to shrink size to 50%
- Targeted attack – Quite robust
- CAs revoked – LSCC still at size 94.4%

7.2.6 Social Structures (of LSCC)

- Community structure existing
- Mapping to TLDs ok, but not to Second level domains
- Signing process supported by social link

7.2.7 Crypto Algorithms

- Few keys with weak keylength / hash algorithms

8 Middleboxes

- Intermediary device perfonming unusual functions on data on its path

8.1 Firewall

- Restricts people to enter / leave at one controlled point
- Prevents attacker getting close to other defenses
- Between trusted / untrusted network ⇒ Access control
- Incoming / outgoing packates
- Working based on rule set and default rule (Default deny strategy – Whitelisting or Default permit strategy – Blacklisting)

8.1.1 Functions of firewalls

- Forward a packet (Allow / Permit / Accept Pass)
- Delete a packet, do not forward (Drop / Deny / Reject)
- Create logs, send error to sender, inform admin, …

8.1.2 Information available to firewall

- Link layer: direction, next hop (Usually link layer communication does not pass the firewall)
- Network layer: communication end point, transport protocol
- Transport layer: port, protocol state
- Application layer: deep package inspection

8.1.3 Packet Filtering

- Detailed knowledge of high layer protcols ⇒ Proxy
- Simple but fast operations on individual packets ⇒ Packet filtering
- Packet filtering controls data based on source / destination address, transport protocol, ports, protocol flags, network interface

8.1.4 Stateless Packet Filtering

- Packet information need to be trusted (Attackerd could send SYN/ACK packates initialy)
- UDP has no useful state information
- Simple to implement, high performance

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Mailserver	TCP	>1023	25	Any	Permit
B	Outbound	Mailserver	External	TCP	25	>1023	Yes	Permit
C	Outbound	Internal	External	TCP	>1023	25	Any	Permit
D	Inbound	External	Internal	TCP	25	>1023	Yes	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

Figure 9: Example of a stateless firewall table

8.1.5 Stateful Packet Filtering

- Arriving packets may generate state in the firewall
- Check protocol state (even for UDP)
- Reaction to abusive behaviour possible (dropping packets for some time)
- Rate limiting, …
- Possileb states: New, Established, Related (e.g. FTP), Invalid (Invalid header fields)

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest.Port	State	Action
A	Inbound	External	Webserver	TCP	>1023	80	New	Permit
B	Outbound	Internal	External	TCP	>1023	80	New	Permit
C	Outbound	Internal	External	UDP	>1023	53	New	Permit
D	Either	Any	Any	Any	Any	Any	Established	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

Figure 10: Example of a stateful firewall table

8.1.6 De-militarized zone (DMZ)

- Subnetwork to provide additional security (also called perimeter network)

8.1.7 Bastion Host

- Computer that must be highly secured, as it is especially xulnerable
- Bastion host in firewall is usually contact point for user processes
- Purposes: Packet filtering, Proxy service
- Set-Up principles: Simple, Prepare for Bastion host to be compromised (No trust, Sniffing not possible), Extensive logging
- Make host unattrictive as target (slow, few tools), Disable user accounts, Secure syslog, Backups / Monitoring

8.1.8 Firewall Architectures

- Simple Packet Filter Architecture
- Dual Homed Host (Proxy and Packet Filter) Architecture (Proxy Server with two network interfaces, Bottleneck possible)
- Schreened Host Architecture (Packet filter allows traffic between screened host and the external network)
- Screened Subnet Architecture (DMZ between two packet filters, inner packet filter for additional protection, www-server can be placed in DMZ)
- Split Screende Subnet Architecture (Dual Homed Bastion Host splits DMZ in two networks)

8.2 Application Proxy

- Deals with external servers on behalf of internal clients
- If proxy understands application layer protocol → Application Level Proxy
- If proxy just forwards PDUs → Circuit Level Proxy (SOCKS Proxy)
- Client thinks of talking to the actual server
- Server thinks of talking to the proxy

8.3 Networks Address Translators (NAT)

- Router changes data an packets modifying the network address
- Hide internal net topology
- Security by making client unreachable directly from net

8.4 Virtual Private Networks (VPN)

- Public telecomunication infrastructure used
- Access control through partitioning (logical network)
- Make use of dedicated links
- Controlled route leaking / filtering
- Tunneling

8.5 Case Study: Linux Netfilter

- Package management in Linux with Netfilter
- Iptables
- Packets processed in chains: input / output (to localhost) and forward (router)
- Tables used to group chains

9 Secure Socket Layer (SSL) / Transport Layer Security (TLS)

9.1 Classification in the OSI Reference Model

- Transport layer provides end-to-end service for applications
- TLS adds security features to transprot layer
- Usually session layer protocol
- Transport layer security in internet, as application layer directly on top of transport layer

9.2 SSL/TLS History

- SSL designed to protect HTTP sessions
- IETF specifieds TLS based on SSL (Some algorithmic changes)

9.3 TLS Security Services and Protocol Architecture

9.3.1 Security Services

- Peer entity authentications (Simple / Mutual)
- User data confidentiality (IDEA, DES, 3DES, RC2-CBC, RC2, AES, null)
- User data integrity (MD5, SHA, null)
- Replay protection
- IP address of client known → Reduces potential for DoS attacks (TCP SYN still possible)

9.3.2 TLS Sessions

- TLS handshake creates TLS session (different connections possible)
- Session state:
  - Session identifier
  - Peer certificate
  - Cempression method
  - Cipher specifieds
  - Master secret
  - Is reumable
- Connection state:
  - Server / client random nonce
  - Server / client write MAC secret
  - Server / client write key

9.3.3 TLS Protocol

- Handshake: authentication, setablishment of shared secret
- Change Cipherspec: Transitions in ciphers
- Alert: Error conditions
- Application Data: Transparent access to record protoccol
- Record: Fragmented user data, compression, encryption, integrity protection
  - Type (Change Cipherspec, Alert, Handshake, Application Data)
  - Version of TLS
  - Length of data

9.3.4 TLS Record Protocol Processing

- Sender: User data fragmented, compessed (default is null), MAC added (seq_num not correlated with TCP sequence number, set to zero on init), encrypted
- Receiver: Decrypt, check MAC, decompress, de-fragment, delifer to application

9.3.5 TLS Handshake Protocol

- Different methods for authentication / key establishment
  - RSA: Pre-Master-Secred generated by client, sends encrypted to server
    * $C \to S$: ClientHello(Ver, Random, CipherSuite, Compr)
    * $S \to C$: ServerHello(Ver, Random, SessionID, CipherSuite, Compr), [ServerCertificate], [CertificateRequest], ServerHelloDone
    * $C \to S$: [Client Certificate], ClientKeyexchange, [CertificateVerify], ChangeCipherSpec, Finished
    * $S \to C$: ChangeCipherSpec, Finished
  - Diffie-Hellmann: DH exchange performed, shared secret taken as pre-master-secret (Ephemeral / Temporal – Perfect forward secrecy, Static – certificate algorithm is DH)
    * $C \to S$: ClientHello(Ver, Random, CipherSuite, Compr)
    * $S \to C$: ServerHello(Ver, Random, SessionID, CipherSuite, Compr), [ServerCertificate], [ServerKeyExchange], ServerHelloDone
    * $C \to S$: ClientKeyexchange, ChangeCipherSpec, Finished
    * $S \to C$: ChangeCipherSpec, Finished
- Not protected, as no shared secret
- ChangeCipherSpec denotes, that communication is now protected
- Finished is first protected message, verifies, that key exchange and authentication were successfull

- Key Exchange algorithms
  - RSA
  - DHE_DSS / DHE_RSA: Ephemeral DH values signed with DSS/RSA
  - DH_DSS / DH_RSA: Static DH values signed with DSS/RSA
- master_secret = PseudoRandom(pre_master_secret, "master secret", ClientHello.random, ServerHallo.random), MD5 and SHA-1 HMACs used, for security, even if one has function is broken
- TLS session can be reusmable (reuse security context for several TCP connections) → Abbreviated Handshake
  - ClientHello(Random, SessionID)
  - ServerHello(Random, SessionID), ChangeCipherSpec, Finished(MAC)
  - ChangeCipherSpec, Finished(MAC)

9.3.6 SSL/TLS Alert Protocol

- Transmit errors and exeptions: closed_notify, unexpected_message, bad_record_mac, decryption_failed, ...

9.3.7 SSL/TLS Change Cipherspec Protocol

- Signal transitions in ciphering strategies
- Single message "ChangeCipherSpec" sent using the current connection state

9.3.8 TLS Cipher Suites

- Set of pre-defined cryptographic algorithms (>50 for TLS V1.1)
- TLS_WITH_KeyExchangeAlgorithm_RecordProtocolAlgorithms
- E.g: TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- Ephemeral DH with RSA certificate
- 3DES with Encryption → Decryption → Encryption in CBC mode
- SHA-1 as MAC

9.3.9 Datagram TLS (DTLS)

- TLS running on top of TCP
- DTLS can run on UDP (very similar to TLS)
- Unreliable transport
- Relay protection with sequence number of record header
- Protection agains DoS attack with cookies similar to IKEv2
- Takes care of re-tarnsmitting of controls messages if lost

9.4 IPSec vs. TLS

- TLS always end-to-end, IPSec end-to-end / middle-to-middle / end-to-middle
- TLS protects payload, IPSec protects payload, transport header (and IP header)
- TLS mutual authentication possileb, IPSec mutual authentication is a must
- TLS perfect forward secrecy if ephemeral DH used, IPSec perfect forward secrecy if KE values in IKE_SA_INIT only used once
- TLS on TCP, DTLS on UDP, IPSec on IP protocol (not caring about transport protocol)
- TLS used by any application, IPSec policies require administrative access
- TLS no issues with Middleboxes, IPSec incompatibility as IP header manipulated

10 System Vulnerabilities and Denial of Service Attacks

10.1 Threat Overview

- Private Networks
- Public Internet
- Mobile Communication Networks
- Sensor Networks
- Support Infrastructure
- ISP Networks

10.2 Denial of Service Threats

- Denying / Degrading legitimate access to service or bringing down the server
- Montivation
  - Hacking
  - Gaining information leap
  - Discrediting
  - Revenge
  - Political reasons

10.2.1 DoS Attacking Techniques

- Resource destruction (disabling services, Hacking, Buffer overflow)
  - Ping-of-Death (Oversized IP packet)
  - Teardrop (Overlapping offset fields)
- Resource depletion (State information, High traffic loade, Expensive computations, Resource reservation)
  - Abusing ICMP (Broadcast messages, Routers responding)
  - Smurf attack: Broadcast ICMP echo request with forged source address ⇒ Victim is flooded with ICMP replies (Network as amplifier: Reflector Network)
  - TCP-SYN flood (TCP SYN packets with forged source addresses) ⇒ Useless state information
  - DDos: Overwhelm victim with traffic; Master Systems control slave systems, which launch attack; No traffic from attacker (not even masters); Network Topologies: Master-Slave-Victim / Master-Slave-Reflector-Victim

  - CPU Exhaustion: False digital signature that server tries to verify; Usually some values need to be received from server / guessed; Victim must repeatedly perform expensive computation

- Genuine / forged source addresses of single source or Distributed DoS (DDoS)

10.3 DoS Attacks: Classification

- Exploited vulnerability
  - Software vulnerability attacks (Ping-of-Death, Teardrop)
  - Protocol attacks (TCP SYN flood, Authentication server, Ping-of-death, Teardrop)
  - Brute-Force / flooding attacks
    * Filterable attacks (Flood packages not servic critical, can be filtered – UDP flod / ICMP request flood on web server)
    * Non-filterable attacks (Flood packages request service – HTTP request flood on web server, DNS request flood on name server)
- Attack rate dynamics
  - Continues rate (Sudden packet flood disruprst services quickly, may be noticed quickly)
  - Variable rate (increasing / fluctuating – Detection avoidance)
- Impact
  - Disruptive (Goal fully deny service to clients)
  - Degrading (Portino of recources occupied by attacker, can remain undetected → Slow response, customers change provider

10.4 System Vulnerabilities

10.4.1 Basic Attacking Styles

- Origin: Remote attacks (Host in same network), Local attacks (root privileges on attacked machine)
- Techniques: Buffer overflow, Race condition, Exploiting trust in program input / environment

10.4.2 Identifying Vulnerable Systems with Port Scans

- Identify vulnerable systems to compromise, Automated distribution of worms
- Scan types
  - Vertical: Multiple ports of single IP
  - Horizontal: Multiple machines at same target port
  - Coordinated: Distributed scan of particular port
  - Stealth scan: horizontal / vertical with very low frequency to avoid detection

10.5 Honeypots

- Resource pretends to be attacked / compromised real targed, but redundant / isolated resource ⇒ No damage
- Get to know the enemy
- Low-Interaction Honeypots
  - Emulated services / operationg systems
  - EAsy to deploy / maintaing
  - Long only limited information / limited capture of activities
- High-Interaction Honeypots
  - Real os / applications
  - Captures extensive amount of information
  - Problem: Can be used to attack non-honeypot systems
- Can capture unknow attacks
- Can slow down spread of worms
- Can be taken offline for analysis
- High-interaction honeypots effective to prevent intrusion, provide in-depth knowledge about attacker

10.6 Upcoming Challenges

- IP in mobile communication introduces DoS vulnerabilities to these networks
- Smart phones possible salave nodes for DDos attack
- New attacking techniques through radio implementation
- Integration of communication / automation may enable new DoS threats

11 Attack Prevention, Detection and Response

11.1 Attack Prevention

- Measures taken to prevent an attacker realizing a threat, taken before an attack takes place (Cryptographic measures, Firewall techniques, ...)
- Impossible to prevent all possible attacks

11.1.1 Defense Techniques Against DoS Attacks

- Defense against disabling services
  - Hacking defense (good administration, firewalls, loogging, IDS)
  - Implementation weakness defense (code review, stress testing)
  - Protocol deviation defense (Fault tolerant protocol design, error logging, IDS, DoS-aware protocols)
- Defense against resource depletion
  - Rate controle, accounting & billing, identification & punishment of attackers
  - Authentication of clients
  - Stateless protocols against memory exhaustion
- Origin of malicious traffic
  - Defense against single source attacks (Disableng address range)
  - Defense against forged source address (fngress filtering at ISPs, source verification e.g. cookies)

11.1.2 Ingress / Egress Filtering

- Reduce address spaced usable by an attacker
- Ingress filtering: Incoming packets with source address from network / Packets from internet with private source address are blocked
- Egress filtering: Packets with source not in network are blocked

11.1.3 TCP SYN Flood Attack

- Victim flooded with SYN packet and forged IP address ⇒ Half-Open connections in backlog
- Load Balancing / Replication: Attack unnoticed / Sufficient number of attacker is still successful
- TCP stack tweaking: Increase backlog size / Decrease TCP timeout
- TCP proxies forward only after handshake: Do not solve problem
- Anti-spoofing features

- SYN cookies
  - Initial sequence number $\alpha = h(K, S_{SYN})$
  - Server calculates α on arrival of ACK message. If correct, packet is valid
  - No table to store, transparent for client
  - CPU power for calculating α, vulnerable to crypto-analysis (K need to be changed regularly)

11.2 Attack Detection

11.2.1 Introduction

- Prevention no sufficient (expensive, annoying to users, may fail)
- Detection of attacks / attackers / system misuse, limitation of damage, gain of expeinece
- Intrusion are actions that attemt to compromise integrity, confidentiality or availability
- Intrusion detection are all measures to recognize attacks while or after they occure (Recording and analysis / On-the-fly traffic monitoring)
- Clissification by scope of detection
  - Host-based (HIDS)
  - Network-based (NIDS)
- Clissification by detection strategy
  - Knowledge-based detection
  - Anomaly detection
  - Hybrid attack detection

11.2.2 Host IDS vs. Network IDS

- HIDS
  - Use system information (logs, timestamps)
  - Detects attacks by insiders (illegal file access, installation of trojans / root kits)
  - Need to be installed on every systen
  - Can only defend when attack reaches victim
- NIDS
  - Use network information (sniffed packets)
  - Used at network edges (ingress/egress points)
  - Can detect known attack signatures, port scans, invalid packets, DDos, spoofing
  - Uses signature detection (stateful), protocol decoding, statistical anomaly analysis, heuristics

11.2.3 Knowledge-based Detection

- Attack signatures in database → Communication checked against
- Known attacks reliably detected, But: slightly different attacks not detected
- Pattern specified at each protocol level

11.2.4 Anomaly Detection

- Model of normal system behaviour (normal traffic, expected performance)
- Current network state compared with model
- If state differs alarm is raised
- Anomalies detected in traffic, protocol, application behaviour
- Might recognize unknown attacks
- Might raise many false-positives
- Challenges: Modeling not easy, Data collection expensive, Different reason for aomalies
- Network Operation Anomalies (link failure, configuration change)
- Flash Crowd Anomalies (rapid traffic rise due to sudden interest)
- Network Abuse Anomalies (DoS flood, port scans)

11.3 Response Mechanism

- Packet Filtering
  - Drop attack packets
  - Challanges: Distingish packets, spoofed source address
  - Filterabe attack vs. Non filterable attacks (See chapter 10.3)
- Kill Connections
  - TCP connection killed using RST packets (correct sequence numbers needed)
- Rate Limiting
  - Congestion control
  - Pushback
- Tracking
  - Traceback techniques
  - Re-configuration of monitoring
- Redirection

12 Application Layer Security

12.1 WWW Security

- Application layer security to prevent e.g. Cross-Site scripting, Buffer Overflows
- Attacks that are not detectable on lower levels

12.1.1 WWW and its Security Aspects

- URI (Uniform Resource Identifier): <scheme>://<authority><path>?<query>#<fragment> e.g. URL (Locator), URN (Name)
- HTTP as carrier protocol for HTML
  - Stateless ⇒ Sessions (target, weakness)
  - Mostly simple GET/POST ⇒ Cross Site Request Forgery
- HTTP Authentication
  - Basic Authentication: No security (plain text messages)
  - Digest Authentication
- Cookies
  - Text files stored by browser (e.g. session information)
  - Which cookies is which site allowed to access?
  - Privacy issues (3rd party cookies, user tracking, ...)
- JavaScript
  - Script language executed on client side
  - Dynamic web content (AJAX)
  - Security issues: Malicious code on client, cross site scripting ⇒ Sandboxing, Same-origin policy (Protocol, Hostname, Port have to match for DOM access, Do not cover SSL connectivity, cookies, ...)

12.1.2 Internet Crime

- Organized crime in the internet
- Credit card information, bot nets, ... can be bought in the internet

12.1.3 Vulnerabilities and Attacks

- Web vulnerability
  - Many important businesses, High functionality and complexity, Global availability
- Context
  - State information of session / process

  - In browser: cookies, scripts, plugins, …
- Attack on session variables
  - Server does not keep state information
  - Attacker can swap variable information (e.g. Domain of certificate request)
- Guideline 1: Everything relevant has to be stored locally
- Guideline 2: All input is evil
- Cross site scripting (Script input accepted, Abuse of user's trust in website)
- Cross site request forgery (User looged in on site B, M shows malicious code) → Confused deputy (browser) problem ⇒ Secret tokens
- SQL injection (SQL code as input accepted)
- Defense against XSS, XSRF, SQL-injection
  - JavaScript sandboxing / security features activated
  - Treat all input as untrusted: Propper escaping (use functionality of modern script languages), Whitelisting instead of blacklisting
- Buffer overflow (Overwrite return variable on stack) → Programming mistake in application, Input not checked ⇒ Data execution protection, Adress space layout randomization, Canaries (prceed return value witch checksum, check before return)

12.2 Web Service Security

12.2.1 XML and Web Services

- Web-Service: Technologies using HTTP for application interoporations (not human users)
- XML: Extensible Markup Language – Syntax rules to encode documents (Can be complex, deeply nested trees, White spaces / ordering do not matter)
- Web-services as middleware similar to remote procedure calls (RPC), mostly asynchronous
- HTTP because of well supported / accepted technology though stateless architecture has drawbacks
- XML because of easiness (used by vendors, …) though parsing very slow, encryption / signature problems with ordering / white spaces
- Blocks of web services: WSDL (Web services description language) and SOAP (carrier protocol)

12.2.2 Securing Web Services

- Security challenges: Securing identities, messages, multi-hop message flow
- Why not just SSL? SSL point-to-point security between hosts, not services (Think of e-mail encryption); Multi-hop scenarios
- Legally binding signatures needed, Intermediate servers might be involved
- SOAP
  - Transfering structured XML over networks (stateless, one-way messages)
  - Foundation for web service protocols
  - Information can be secured with XML DSig and XML-Enc (at price of high complexity)
- XML Digital Signature (XML DSig)
  - Sign XML document for message integrity, origino authentication, non-repudiation
  - Usual cryptographic mechanisms (but modified for XML)
  - Encryption is XML fragment itself
  - Sign anything, referable by an URI (whole document, part, external document), Multiple signatures allowed
  - Pitfals: Encoding, line breaks, white spaces, … $\Rightarrow$ Canonicalize document before signature (UTF-8, Line break normalization, …, Lexicographical order)
  - 5 different transformations before encryption possible
  - Kinds of signature: Enveloping, Enveloped, Detached
  - Signed documents very large, Parsing, canonicalization, transformation very slow, very complex (5 transformations, 3 kinds of signature)
  - Sanely used it provides security
- XML Encryption (XML-Enc)
  - Encrypt XML content for confidentiality
  - Similar to XML DSig
- WS Security
  - Defines how XML DSig and XML-Enc can be safely deployed with SOAP
  - Standardizing the standards (no new mechanisms) $\rightarrow$ XML DSig, XML-Enc, Transport Security Tokens, Timestamps
- WS-Interoperability Basic Security Profile (WS-I BSP) $\rightarrow$ Clearification (e.g. older protocols forbidden)
- Security Assertion Markup Language (SAML)

- Shared identities with attributes between organisations
- Assertion (Authentication, Authorization, Attributes) – Claim abount subject, that must be proved
- SAML constists of Assertions, Protocol (XML schema), Bindings (e.g. SOAP)
- SAML profiles specify use patterns
- Elements: Issuer, Subject, Timestamp, Conditions, Audience, Signature

12.2.3 Identity Federation

- Shared Authentication (forward authentication to other provider)
- Identity provider (authentication with credentials from identity provider)
- Concept: Sharing identities between organisations, organisations form circle of trust (See Kerberos in chapter 3.3.2)

13 Some More Secure Channel Issues

13.1 Stream Cipher and Block cipher

- Stream Ciphers
  - Reuse of IV leads to same cipher text ⇒ Known plaintext attack ($P_1 + P_2 = C_1 + C_2$)
  - No / weak integrity check ⇒ Single bits can be changed, Changes in text unnoticed
- Block Cipher Modes
  - Reuse of IV can give hints about identical first blocks, But plaintext still safe
  - Weack checksum cannot be targeted directly, as single bits cannot be controlled

But attacks resulted from misuse of algorithms

13.2 Cerckhoff's Principle

- "A cryptosystem should be secure even if everything about the system, except the key, is public knowledge"
- Good guideline for good cryptographic design

13.3 Horton Principle

- "Authenticate what you mean, not what you say"
- Plaintext authenticated, not ciphertext ⇒ MAC-then-Encrypt
- BUT: Security proofs for Encrypt-then-MAC succeed against slighlty stronger attacker

13.4 Attacking CBC and MAC-then-Encrypt

- MAC does not protect ciphertext, Integrity check after encryption ⇒ Performance Issue
- Earlier TLS: Different messages for padding / MAC errors
- Switch last bit to fit padding ⇒ $C_{n-1,n} \oplus P_{n,n} = 1$. (Chance i in 256), Now $P_{n,n}$ known
- Set $C_{n-1,n}$ to satisfy $C_{n-1,n} \oplus P_{n,n} = 2$. Switch second last bit: ⇒ $C_{n-1,n-1} \oplus P_{n,n-1} = 2$
- ...
- TLS stopped diferenciating between errors ⇒ Timing attacks
- Attack not possible with Enrcypt-then-MAC