# ITS - FS20

*Pascal Brunner - brunnpa7*

# Inhaltsverzeichnis

# Kapitel 1

# Vorlesung 1 - Einführung Crypto

## 1.1 Sie können den Unterschied zwischen Kryptographie und Kryptoanalyse benennen und die Ziele, die grundlegende Terminologie und die grundlegenden Eigenschaften verstehen

Der Zweig der Mathematik, der sowohl die Kryptographie als auch die Kryptoanalyse umfasst, ist die Kryptologie und ihre Praktiker sind Kryptologen.
Moderne Kryptologen sind im Allgemeinen in theoretischer Mathematik ausgebildet - sie müssen

### 1.1.1 Kryptographie

Die Kunst und die Wissenschaft wie man Nachrichten sichert ist **Kryptographie** und wird von Kryptographen ausgeführt. Es stammt aus dem Griechischen *versteckt / sicher* (crypto) *schreiben* (graphy)

#### Ziele der Kryptographie

Die Ziele der Kryptographie lässt sich als Dreieck von 'CIA' darstellen, welche nachfolgende beschrieben sind:

- **Confidentiality / Vertraulichkeit** → Eine Nachricht kann nur von den vorgesehen Parteien (bspw. Sender und Empfänger) gelesen werden (bspw. durch Transformation der Nachricht mit einem Key) ⇒ Ist die Vertraulichkeit einmal verloren, so kann diese nicht wieder hergestellt werden.
- **Integrity / Integrität** → Der Empfänger der Nachricht kann verifizieren, dass die Nachricht nicht verändert wurde
- **Authenticity / Echtheit** → Der Empfänger einer Nachricht kann ihren Ursprung feststellen, ein Angreifer sollte sich nicht als jemand anderes ausgeben können. ⇒ ist eine zentrale Eigenschaft
- *Freshness / Frische* (manchmal relevant) → Eine Nachricht ist neu und keine Wiederholung von früheren Nachrichten
- *Non-Repudiation / Nicht-Abstreitbarkeit* → rechtlicher-Natur, nicht technischer, Der Absender kann später nicht abschreiten, dass er die Nachricht gesendet hat. ⇒ echte Nichtabstreitbarkeit ist unterschiedlich hart, und kein heute weit verbreitetes System bietet sie tatsächlich an.
- *Anonymity / Anonymität* → weiteres Ziel ⇒ nicht relevant

**Fachbegriffe**

- Das Hauptziel von Secret Key Cryptograhy ist eine Vertraulichkeit herzustellen.
- Eine Nachricht ist in Plaintext $P$ (oder Cleartext) $\rightarrow$ Der Prozess der Transformation wird **Encryption** genannt.
- Eine verschlüsselte Nachricht wird **Ciphertext** genannt.
- Der Prozess der Entschlüssung wird **decryption** genannt.
- Die mathematische Funktion für das Verschlüsseln bzw. Entschlüsseln ist ein kryptologischer Algorithmus welcehr **cipher** genannt wird
- Die moderne Kryptologie basiert auf **Secret Keys**, welcher einen grossen Wert annimmt. $\Rightarrow$ Die Range des Keys wird **keyspace** genannt.
- Sowohl die Verschlüsselung (Encryption), als auch die Entschlüsselung (Descyrption) ist abhängig vom Key $K$ mit den Funktionen $E_K(P) = C$ und $D_K(C) = P$

**Eigenschaften**

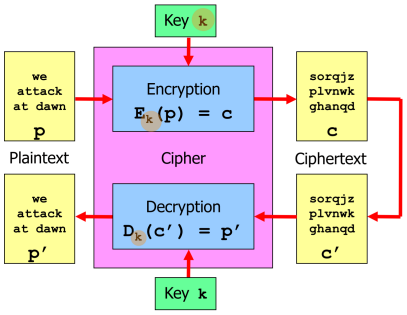


*Abbildung 1.1: Basic Terminology basierend auf Secret Key Cryptography*

- Die Verschlüsselung ist eine Transformation von $P$ (Set von Plaintexte) zu $C$ (Set von Ciphertexts) unter der Anwendung von $K$ (Set von Keys)
- Grundidee ist, dass man nicht einen einzigen Key hat, jedoch eine Vielzahl von Transformationen, wobei jeder Key einen andere Transformation nachvollzieht
- $c = E(k, p)$ oder $E_K(P)$
- $p = D(k, c)$ oder $D_K(c)$
- Jede Transformation (jede Wahl des Schlüssels) muss reversibel (injektiv) sein. $\Rightarrow |C|$ kann nicht kleiner sein wie $|P|$, meistens gilt $P = C$

### 1.1.2 Kryptoanalyse

Hingegen befassen sich Kryptoanalysten mit der Kryptoanalyse, welches die Kunst und die Wissenschaft rund um das Entschlüsseln von *Ciphertext* ist.

## 1.2 Sie können Shannons grundlegende Konzepte definieren: Information, Entropie, vollkommene Geheimhaltung und sein Modell eines Kryptosystems mit geheimen Schlüssel

Claude Shannon: *Üm das Problem mathematisch lenkbar zu machen, gehen wir davon aus, dass der Feind das verwendete System kennt. Das heißt, er kennt die Familie der Transformationen [E(k, p) und D(k, c)] und die Wahrscheinlichkeiten der Wahl verschiedener Schlüssel.*

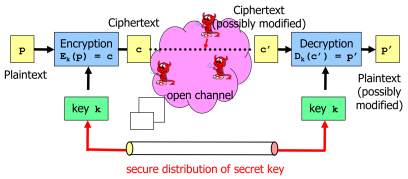


*Abbildung 1.2: Shannon's Model von einem Secret Key Cryptosystem*

- Der gleiche Key / Schlüssel wird für das Verschlüsseln und Entschlüsseln verwendet
- Der Key muss absolut geheim sein
- **Schlüssel-Verteil-Problem** → Der Schlüssel muss über einen vertraulichen und authentifzierten Kannel erfolgen

Problem dabei ist, dass der Kanal nicht sicher ist, Encyrpt und Decrypt hat denselben Key und die Verteilung des Schlüssel muss sicher erfolgen.

### 1.2.1 Entropy - Informationsgehalt messen

Wenn die Schlüssel nicht mit der gleichen Wahrscheinlichkeit gewählt wurden, und der Angreifer das weiss, dann macht das das Leben des Angreifers um einiges einfacher.

#### Definition

Nehmen wir eine zufällig endlich gewählte Variable $X$ mit $n$ Ergebnisse, wobei $p(i) = P(X = i)$ die Wahrscheinlichkeit des Ergebnisses $i$ ist. Wenn die Wahrscheinlichkeit $P(i)$ gross bzw. klein ist, ist das Ergebnis (nicht) überraschend. $\Rightarrow \log_2 \frac{1}{p(i)}$ definiert aus diesem Grund den Überraschungsfaktor.
**Formel** Die Entropy(H) von X, wird in Bits gemessen und ist die durchschnittliche Überraschung des Ergebnis

$$H = \sum_{i=1}^{n} p(i) \log_2(\frac{1}{p(i)} \tag{1.1}$$

**Maximale Entropy**: wird erreicht wenn alle Ergebnisse gleich wahrscheinlich sind. → In diesem Fall ist es am schwierigsten das Ergebnis im Vorfeld zu erraten
**unabhängige Ereignisse**: Wenn die Experimente unabhängig sind, dann ist die Entropy additive. → Durch $\log_2$ liefert $n+1$ bits immer zweimal so viel Informationen wie $n$ Bits. Wobei 4 Bits nicht zweimal so viel Information wie 2 Bits liefern, jedoch $\frac{2^4}{2^2} = \frac{16}{4} = 4$ so viel Information

#### Fairer Münzwurf

Die Entropy eines fairen Münzwurfs ($\rightarrow p_{heads} = p_{tails} = 0.5$) wird wie folgt berechnet

$$H = 0.5 * \log_2(2) + 0.5 * \log_2(2) = 2 * 0.5 * \log_2(2) = 1Bit \tag{1.2}$$

$\Rightarrow$ Ein fairer Münzwurf liefert 1 bit Information

**Unfairer Münzwurf**

Die Entropy eines unfairen Münzwurfs ($\rightarrow p_{heads} = 0.25 p_{tails} = 0.75$) wird wie folgt berechnet

$$H = 0.25 * \log_2(4) + 0.75 * \log_2(2) = 2 * 0.5 * \log_2(\frac{4}{3}) \approx 0.81 Bit \tag{1.3}$$

$\Rightarrow$ Ein unfairer Münzwurf liefert 0.81 bit Information $\rightarrow$ liefert weniger Information als ein fairer Münzwurf, da beim unfairen Wurf das Ergebnis einfacher zu erraten ist. Wäre es ein komplett unfairer Wurf und es kommt immer Kopf, so würde es 0 Bit Information liefern.

**Entropy für Passwörter**

Die Auswahl eines Passworts kann ebenfalls als ein Resultat eines zufälligen Experiments betrachtet werden.

- *Ein Passwort beinhaltet 8 zufällige Hexadezimal Zeichen (0-9, A-F)* $\rightarrow 2^4$ Möglichkeiten pro Zeichen $\rightarrow (2^4)^8 = 2^{32}$ gesamtmögliche Anzahl Passwörter
- *Der konstante String '0000 0000'* $\rightarrow$ nur ein Passwort: $H = \log_2(1) = 0 bitsentropy$, wobei es nicht nur 00000 ist, sondern dass es einen konstanter String ist. Sprich bspw. nur ABCD oder 010101 wäre ebenfalls mit 0 Bit Entropy betitelt.

### 1.2.2 Security of cryptosystems

- Muss resistent gegen die Angriffe sein
- Braucht einen genügend grossen Workfactor
- Should reveal as little as possible about the plaintext
- By construction, a plainttext letter could never be enciphered to itself

### 1.2.3 vollkommene Geheimhaltung

- Ein System auf welches die Eigenschaften eines sicheres Cryptosystem zutrifft, wird **Information-theoretically secure** genannt (oder nach Shannon (*has perfect secrecy*)
- Es gibt genau ein Sytem, welcher diese Anforderungen erfüllt und das ist ein One-Time-Pad

## 1.3 Sie können einen kryptografischen Arbeitsfaktor definieren und verstehen, warum $2^128$ (manchmal $2^256$) ein guter Arbeitsfaktor (work factor) ist

$\rightarrow$ work factor = average number of keys to try

### 1.3.1 Work Factor / Arbeitsfaktor

- Manchmal ist bei einem gegeben C, nur ein K sinnvoll, so dass $p = D(k, c)$ gilt und alle andere Keys keinen Sinn ergeben $\Rightarrow$ in einem solchen Fall kann man den Schlüssel durch einfaches Ausprobieren (Brute-Force) und sehen, wann der Plaintext Sinn ergibt. $\rightarrow$ Dafür muss man die Länge des Keys in etwas abschätzen können
- Die durchschnittliche Dauer / Anzahl welche es benötigt, bis der richtige Key gefunden wird, wird als **work factor / Arbeitsfaktor** betitelt

- Wenn wir von einer dreistelligen Zahl (0,..,9) ausgehen gibt es $10^3 = 1'000$ Kombinationsmöglichkeiten → unter der Annahme, dass alle Zahlen gleich wahrscheinlich sind, beträgt der work factor = 500.5 ⇒ bei 4 Zahlen ($10^4 = 10'000$) beträgt der work factor = 5000.5
- Wenn man eine Zahl hinzufügt, steigt der Aufwand exponential → Mein Aufwand steigt um eins, der von der Drittperson um den Faktor 10
- Falls es keine Shortcuts gibt, ist die Vergrössung des secrets eine gute Idee
- Wenn alle Keys gleichwahrscheinlich sind, ist der work factor ≈ 0.5∗key space size
- Der work factor ist normalerweise gegeben in bits: $\log_2(\textit{key space size})$

### 1.3.2 Was bedeutet gross genug?

Annahme: *Angreifer kann 1 key in 1ps ($10^{-12}$) auf einem Computer ausprobieren. Wobei der Angreifer 1 Milliarde Computers hat*
**32 Bits genügend?** → $2^{32} = 4 * 10^9$ entsprechend braucht der Angreifer $0.5 * 4 * 10^9 \text{keys} 10^{-12}\text{s}/(\text{keys/computer})/1$ computer = 2ms mit einem Computer
**64 Bits genügend?** → $2^{64} = 1.8 * 10^{19}$ entsprechend braucht der Angreifer $0.5 * 1.8 * 10^{19} \text{keys} 10^{-12}\text{s}/(\text{keys/computer})/10^{19}$ computer = 0.01s mit allen Computer
**128 Bits:** → braucht $0.5 * 10^{17.5}\text{s} = 5 * 10^9$ Jahren ⇒ gross genug
**256 Bits:** → braucht $0.5 * 10^{56}\text{s} = 10^{48}$ Jahren ⇒ gross genug

### 1.3.3 Beziehung zwischen Entropy und work factor

**Ergänzen mit zusätzlichen Slides vom OLAT** Der Workfactor muss nicht 0.5-Entropie sein! → Die Aussage, dass der Workfactor der Hälfte der Entropy ist, stimmt nicht in jedem Fall, dies ist abhängig von der Strategie wie man Brute-Force durchführen wird.

## 1.4 Sie können ein One-Time-Pad definieren und seine Eigenschaften benennen

- Ist ein **Informatino-theoretically secure**
- Auch Vernam Cipher genannt
- jedes Bit hat die gleiche Wsk
- jedes Bit ist unabhängig vom anderen
- Schlüssel ist gleich lang wie der Plaintext
- Verschlüssung ist eindeutig ⇒ Die Menge aller Klartexte (P) und die Menge aller Verschlüsselungen (C) sind injektiv. Dies bedeutet, dass jedes Element in P genau ein Element in C hat
- Des Weiteren gilt die Surjektivität, sprich jedes Element in C hat genau eine Abbildung in P.
- Daraus folgt $|P| = |C|$
- wir brauchen so viele Schlüssel wie wir Text haben
- Dies erfolgt durch eine XOR-Verknüpfung
- **WICHTIG** Für jeden Plaintext wird ein neuer Key verwendet. Dies gilt für alle jemals erstellten und alle welche in Zukunft erstellt werden. → Andernfalls würde die Bedingung der eindeutigen Abbildung nicht mehr gelten. Ein Key darf nicht wiederverwendet werden.
- Der Schlüssel ist im Vorfeld bekannt und beide Seiten kennen diesen Schlüssel.

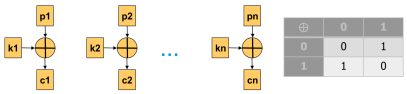


*Abbildung 1.3: One Time Pad*

Da One-Time Pad nicht immer praktikabel ist, werden die Eigenschaften etwas abgeschwächt.

- Gleicher Key für encrypt und decrypt
- Falsche Keys geben eine hohe Entropy als Ergebnis → can usually tell when correct key used
- guter moderner cipher → key entropy ≈ work factor
- Design-Parameter block-size und key-size
- Heute AES (Advanced Encryption Standard) standard

Nach Information-theoretically Secure, folgt *computational security (work factor ≈ key entropy.* Problematik dabei ist, dass es keinen Beweis gibt, dass der Algorithmus computationally secure sicher ist.
*Hinweis:* Hashing-Verschlüsselungen sind am Altern, unteranderem weil man immer mehr Computer-Power hat

## 1.5 Sie können das Zufallsorakel-Modell der Secret-Key Ciphers definieren und seine Konsequenzen benennen

**Prüfungsrelevant**
Das Random-Oracle Model ist ein allgemeines Model

**Vorbedingungen:**

- P entspricht einem Set von allen Plaintexten
- C entspricht einem Set aller Ciphertexts
- K entspricht einem Set von allen Keys
- Wir nehmen an, dass $P = C$

**Eigenschaften:**

- TBD

**Vorgehen:**

- TBD

# Kapitel 2

# Vorlesung 2 - Secret Key Cryptography

## 2.1 Sie verstehen das Konzept der Block- und Stream-Cipher und wissen, wie sie in der Praxis eingesetzt werden

Wenn wir das Prinzip des Secret Key Cryptography von Shannon hervornehmen, erinnern wir uns, dass derselbe Schlüssel für die encryption, sowie für die decryption verwendet wird (dieser muss absolut geheim sein). Der gleiche Key kann für mehrere Nachrichten verwendet werden, dieser Key sollte jedoch periodisch ausgewechslet werde → Verteilungsproblem

### 2.1.1 Block-Ciphers

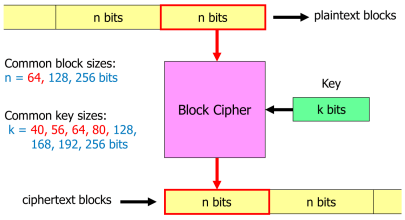


*Abbildung 2.1: Block Ciphers*

- Block-Ciphers schneidet einen Klartext beliebiger Länge in eine Reihe von Blöcken mit konstanten Grösse von n Bit auf
- Anschliessend wird jeweils einen Klartextblock verschlüsselt und entsprechend als Block-Cipher dargestellt.
- Jeder Schlüssel muss von einem Klartextblock zu einem Ciphertextblock führen (1:1 Abbildung), andernfalls könnte ein Ciphertextblock nicht eindeutig entschlüsselt werden.
- Die Feistel-Chipher ist eine gängige Art von Block Ciphers, welche diese Eigenschaften per Design erfüllen.

#### Common Block Sizes

- Bis Mitte 90er Jahre war 64Bit die gebräuchliche Länge
- Ab 2011 128 als Mindestschlüssellänge angenommen um Brute-Force-Angriffe zu verhindern
- 8, 16, 32 Bit Blockgrössen sind nicht sicher → ANgreifer lernt die Paare indem er bspw. eine Tabelle führt ⇒ bei 16 Bit sind dies bspw. nur $2^{16}$ Paar-Möglichkeiten
- Bei 64-Bit-Blöcken hätte diese Tabelle bereits $2^{64}$ Einträge (=16 exa-Einträge). Es ist sehr unwahrscheinlich, dass ein Angreifer so viele Klartext-Chiphertext-Paare, die den gleichen Schlüssel verwenden, sowohl akkumulieren als auch speichern kann

**Common Key Sizes**

Schlüsselgrössen mit 40, 56 und 64 Bit sind eindeutig unsicher und sollten nicht mehr verwendet werden. Verwenden Sie sicherheitshalber eine Schlüsselgrösse von 128 Bit oder mehr.

**Randomness of the mapping**

- Ein guter Block-Cipher hat die Eigenschaft, dass die Zuordnung von Plaintext-Block zu einem generierten Ciphertext-Block bei einem beliebigen Schlüssel zufällig aussieht → als ob die Bits des Ciphertext-Block durch das Werfen einer fairen Münze erzeugt wurde
- Wenn die Abbildung eine gute Zufallseigenschaft hat, ist es für den Angreifer sehr schwierig, Informationen über einen Plaintext-Block durch Betrachtung des entsprechenden Geheimtextblocks abzuleiten
- Aus mathematischer-Sicht sind dann der Plaintext-Block und der Ciphertext-Block *so unabhängig wie möglich* → Änderungen eines einzelnen Bits in einem Plaintext-Block führen zu einem völlig anderen Ciphertext-Block ⇒ Dies erreicht man in dem jedes der n Bits des Chiffrierblocks eine Funktion aller n Bits des Klartextblocks und der k Bits des geheimen Schlüssels sein. DES erreicht dies beispielsweise durch geschickte Verwechslungs- und Diffusionsoperationen

## 2.2 Sie kennen die wichtigsten Block- und Stream-Cipher und können anhand der Schlüssellänge eine Aussage über deren kryptographische Stärke machen

### 2.2.1 Data Encryption Standard (DES)

### 2.2.2 Triple DES

**Meet in the Middle Attack (Known-Plaintext Attack**

- Triple DES wird in zwei Teilen unterteilt
- Sämtliche $2^1 12$ Varianten werden in einer Tabelle dargestellt
- Ebenso wird eine Tabelle für $2^5 6$ erstellt
- Danach werden in den beiden Tabellen nach Paaren gesucht, bei Übereinstimmung hat man den Key

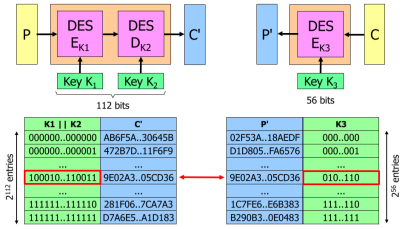


*Abbildung 2.2: Meet in the Middle Attack*

### 2.2.3 Advanced Encryption Standard (AES)

- Stand jetzt ist die beste Möglichkeit einen AES zu knacken, ist mit dem Brute-Force Vorgehen

**Requirements**

- Sollte ein symmetrischer Block Cipher sein

## 2.3 Sie verstehen die verschiedenen Blockchiffriermodi, ihre Stärken und Schwächen und kennen ihre Anwendungen

## 2.4 Block Cipher Mode

**Electronics Code Book Mode (ECB)**
⇒

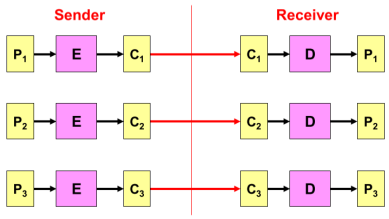


*Abbildung 2.3: Electronic Code Book Mode*

**Cipher Block Chaining Mode (CBC)**

- XOR-Verknüpfung mit dem vorherigen CipherBlock
- Das Inverse der XOR-Verknüpfung ist wiederum die XOR-Verknüpfung
- es entsteht eine Abhängigkeit

## 2.5 Stream Cipher

# Kapitel 3

# Vorlesung 4 - Data Integrity and Authentication

## 3.1 Sie kennen das prinzipielle Konzept einer Einweg Hash-Funktion

- Die Hash-Funktion mappt eine variable-Länge von bit strings als Input (Nachricht) zu einer fixen Grösse als Output-String (Der Hash oder message digest) → many-to-one-mapping
- Typische Hash-Länge sind: 128, 160, 256, 512
- Alle Eigenschaften erfüllt: In der Praxis erzeugt eine Nachricht einen eindeutigen Hash, und ein Hash gehört zu einer bestimmten Nachricht → Verwendung des Hash als Stellvertreter für die Nachricht selbst

**Wichtige Eigenschaften**

- Der Hash kann effizient berechnet werden, auch wenn die Länge der Nachricht mehrere Gigabytes gross ist → Da das Dokument in der Regel viel grösser als der Hash-Wert ist, ist die Abbildung eine many-to-one Funktion. Für jeden spezifischen Hash-Wert gibt es potenziell viele Dokumente, die diesen Fingerabdruck besitzen.
- Das Mapping sollte pseudo-zufällig passieren → Es gibt keine Verbindung zwischen der Nachricht und ihrem Hash. Der Hashwert der Nachricht sollte von jedem Bit entsprechenden Nachricht abhängen. Wenn ein einzelnes Bit der Originalnachricht seinen Wert änder oder ein Bit hinzugefügt oder gelöscht wird, dann sollten etwa 50 Prozent der Digest-Bits ihre Werte in zufälliger Weise ändern.
- Wenn man einen Hash gegeben hat, ist es praktisch unmöglich die Nachricht zu finden, welchen diesen Hash produziert (**preimage resistance**)
- Es ist praktisch unmöglich zwei Nachrichten zu finden, welchen zum gleichen Hash-Wert mappen (**collision resistance**

### 3.1.1 Hashes / Message Digests

Ein Hash mit einer fixen Grössen agiert als eindeutiger Fingerabdruck für eine Nachricht (Dokument etc.) mit willkürlicher Grösse. Mit einer gewöhnlichen Verwertungsgrösse (digests-size) von 128 .. 256, gibt es $10^{38}$ .. $10^{77}$ verschiedene Fingerabdruckmöglichkeiten.

### 3.1.2 Populäre Hash-Funktionen

- MD5 → Message Digest Number 5 ⇒ geknackt
- SHA 1 → Secure Hash Algorithm Number 1 ⇒ geknackt
- SHA 2 → Secure Hash Alogrithm Number 2 ⇒ wird als sicher angesehen

### 3.1.3 Merkle-Damgard Construction of hash Functions

*Vorlesung 4 - Slide 9 evtl ergänzen Vorlesung 4 - Slide 13 evtl ergänzen*

## 3.2 Sie wissen wie man Integrität und Authentizität einer Nachricht / Dokument mittels message authentication codes und digitalen Signaturen schützt

### 3.2.1 Message Authentication Codes

Ein digitaler "message digestßelbst bietet noch keinen Schutz gegenüber unauthorisierter Änderungen eines Dokuments oder Nachricht. Nach jeder Änderung an einem Dokument gibt es einen neuen gültigen Hashwert welcher auf den neuen Inhalt berechnet wird.
Einzig mit der Einführung eines *secret keys* in Form eines berechneten Fingerabdrucks kann ein Dokument gegen unbefugte Veränderungen gesichert werden. Nur der Eigentümer dieses Keys kann die Nachricht entsprechend auswerten → **Message Authetnication Code (MAC)**. Dabei muss natürlich der Empfänger des sicheren Dokuments im Besitz des secret Keys sein. Dies um die Nachricht entsprechend zu verifizieren und validieren, dies wird mittels Berechnung und Vergleich des **Message Authentication Codes (MAC)-Werts**. ⇒ Manchmal wird MAC auch als **Message Integrity Check (MIC)**

### 3.2.2 Digital Signatures

Die ursprüngliche Idee von digitalen Siganturen war, dass man etwas identisches zur handgeschriebenen Signatur zur Unterzeichnung von digitalen. Entsprechend sollten auch deren Eigenschaften sein:

- Die Signatur sollte nicht einfach zu fälschen sein
- Jeder kann die Gültigkeit der Signatur validieren
- Die Signatur ist zu einem Dokument gebunden
- Die Unterschrift kann nicht zurückgewiesen werden, → der Unterzeichner kann sich später nicht darauf berufen, dass er das Dokument nicht unterschrieben hat.

Eine Möglichkeit dies umzusetzen ist mit *public key cryptography*

#### digitale Signaturen mittels public Key Cryptography

Um eine Dokument oder Nachricht zu *unterschreiben*, wird ein **private key** verwendet → Nur der Eigentümer des private key kann die Signatur erstellen
Um eine Signatur zur *verifizieren*, wird ein **public key** verwendet → jeder kann die Signatur verifizieren
**Notation**

$$\begin{aligned} Signatur &\Rightarrow S_{K_Priv}(M) = Sig_M \\ Verify &\Rightarrow V_{K_Pub}(M, Sig_M) = true/false \end{aligned} \tag{3.1}$$

**Vorgehen**

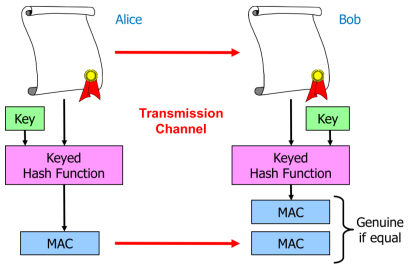


*Abbildung 3.1: Message Integrity / Authenticity with Message Authentication Codes*

1. Alice berechnet mittels der Hash-Funktion den Hash-value zusammen mit dem Private-Key entsteht die Signatur
2. Bob kann die Berechnung nicht von oben nach unten durchführen, denn ihm fehlt die Information des private Keys.
3. Jedoch kann Bob den Hash berechnen
4. Bob kann die Signatur mittels dem Public Keys zurück berechnen und erhält einen neuen Hash-Wert
5. Bob vergleicht nun die beiden Hash-values

## 3.3 Sie verstehen wie schwierig es ist eine Nachricht zu fälschen oder zu kollisionen

### 3.3.1 Fälschung

Ziel ist es durch einen zufälligen Text und der Hash-Function den gleichen Hash-value zu haben. Denn der Hash-value ist repräsentativ für das Dokument. Gelingt einem dies, so können wir dies entsprechend fälschen ohne, dass es bemerkt wird.
Wenn der Hash-value m bits lang ist, braucht man im Durchschnitt $2^{m-1}$ (halb so viele) Versuche um das Dokument zu verfälschen.

⇒ Das Birthday-Paradox-Beispiel ist das gleiche Problem, wie ein Dokument zu finden mit dem identischen Hash

### 3.3.2 Kollision

Eine Kollision ist, dass irgendwelche zwei Dokumente denselben Hash-Wert haben. Dies ist im Vergleich zur Fälschung um einiges einfacher, als bei einem gegegeben Dokument mit einem gegegebenen Hash-value, einen identischen Hash-Value zu finden

#### Birthday Attacks

Birthday Attacken kann man auch gegen Hash-Funktionen anwenden. Man fügt dem original Dokument und dem gefälschten Dokument neuen zufälligen Text hinzu und überprüft ob die Hash-Werte identisch sind. ⇒ Alleine aus diesem Grund sind MD5 und SHA-1 nicht mehr sicher. Aus diesem Grund muss die Kryptographie Kollision resistent sein

### 3.3.3 MD5-Security

- ist nicht kollissionsfrei
- Ab 2007 war es sehr einfach solche Dokumente mit dieser Hash-Funktion zu kollidieren
- nicht trivial, braucht man $\approx 2^{50}$ MD5-Berechnungen
- Selbst wenn MD5 eine vernünftige Länge hätte, sollte MD5 nicht verwendet werden

### 3.3.4 SHA-1 Security

- Nicht viel besser
- Work-factor inkl. möglichen Abkürzungen braucht man $2^{63}$ Berechnungen für eine Kollissino
- Ist kompromentiert $\Rightarrow$ nicht mehr verwenden

### 3.3.5 SHA-2 und SHA-3

- Für SHA-2 gibt es keine bessere Möglichkeit als mit Brute-Force anzugehen
- die NSA war bei der Erstellung dieser Hash-Funktion mitbeteiligt
- Trotzdem kann man diese ohne Sorgen verwenden
- noch nicht klar ob SHA-3 die dominante Hash-Funktion sein wird. $\rightarrow$ grundsätzlich sollte man momentan auf SHA-2 bleiben, jedoch SHA-3 im Hinterkopf behalten

## 3.4 Sie wissen wie Verschlüsselung und Integritäts-Schutz kombiniert werden sollen, um sich vor Angriffen zu schützen

Wir haben festgestellt, dass reine Verschlüsselung uns nicht den vollständigen Schutz bietet, da man durch andere Attacken trotzdem das Dokument nicht vollständig schützen kann. Aus diesem Grund sollte man immer sowohl eine Verschlüsselung anwenden, wie auch einen Integritätsschutz. Dafür hat man zwei Möglichkeiten:

1. Berechnung des MAC mittels Plaintext und hängt den MAC an den Plainttext und verschlüsselt den Plaintext und MAC $\rightarrow$ *MAC then encrypt*
2. Verschlüsselung Plainttext, Berechnung eines MAC mittels Ciphertext und hängt diesen an den Ciphertext an $\rightarrow$ *Encyrpt then MAC*

In der Theorie sind beide Möglichkeiten gleich sicher. Dies sieht jedoch in der Praxis anders aus, in der Praxis funktiontiert *Encrypt then MAC* besser.

### 3.4.1 Handlungsempfehlung

1. Niemals nur Verschlüsselung, sondern immer noch Integritätsschutz miteinbauen
2. Falls das nicht geht, sollte man im Minimum MAC, zusammen mit encyrpt-then-MAC verwenden
3. Bei der Entschlüsselung, sollte man bei einer Fehlermeldung immer alles bereits entschlüsselte Wegwerfen und nicht an den Benutzer aushändigen

### 3.4.2 Galois/Counter Mode

**Vorteile**

- Performanter
- offizieler NIST-Standard
- Es ist eine integrierte Lösung von Verschlüsselung und Integritätsschutz
- Es folgt dem encrypt-then-MAC-Prinzip

## 3.5 Sie wissen wie Passwörter für die Authentifizierung verwendet wird und kennen die best-practice Ansätze vor dem Gebrauch von Passwörtern

Passwörter sind grundsätzlich sehr populär und einfach zu implementieren.

Prinzipien:

- Jeder Benutzer hat eine eindeutige user id und ein Passwort
- Der Server speichert alle user ids und deren Passwörter in einem Passwort-File
- Um sich zu authentifzieren gibt der User seine User Id und das Passwort zum Server, dieser vergleicht es mit den gespeicherten Daten

### 3.5.1 Sicherheitsproblem 1: Sniffing

Wenn Passwörter im Klartext versendet werden, können diese sehr einfach mittels Man-in-the-middle Attacken gesnifft werden.

**Lösung:** Man sollte solche Passwörter nur mittels vergeschützten Links versenden (bspw. mittels TLS)

### 3.5.2 Sicherheitsproblem 2: Phising

Man gaukelt einem user eine Fake-Login-Maske vor und welchem man seine ID und Passwort abfragt

### 3.5.3 Sicherheitsproblem 3: Online Attacks

Der Angreifer ratet verschiedene Passwörter
**Lösung:** Das Zielsystem lässt nur eine gewisse Anzahl von Versuche zu. Des Weiteren kann man anhand der IP-Addresse weitere Einschränkungen zu machen (bspw. max. 10 Passwortversuche pro Minute)

### 3.5.4 Sicherheitsproblem 4: Offline Attacks

Der Angreifer hat Zugriff auf den Computer und klaut die Passwörter vom Password file oder Datenbank
**Lösung:** Dies kann umgangen werden, in dem man die Passwörter nicht im Plaintext abspeichert

### 3.5.5 Sicherheitsproblem 5: Password Re-use

Viele Passwörter werden von den Usern oftmals mehrfach verwendet
**Lösung:** Passwort-Manager verwenden

## 3.6 Sie verstehen, dass die Authentifzierung bassierend eines challenge-response protocol ist, welche shared secret oder digital Signatures verwenden

Bei einem Challenge-Response Protocol wird das Passwort nie über das Netzwerk versendet - dies kann ebenfalls für digitale Signaturen verwenden.

**Grundprinzip**:

- Der Server sendet dem Client eine challenge
- Der Client transformiert die Challenge in einem Weg welcher nur möglich ist wenn mann das Passwort (oder private key) kennt → response
- Die Antwort wird dem Server gesendet, welcher überprüft ob die Antwort korrekt ist
- ⇒ Somit kann man beweisen, dass man im Besitz des Passwortes oder private keys ist → wird auch *Zero Knowledge Proof* genannt

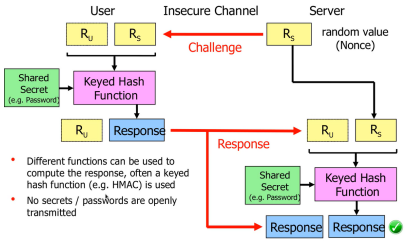


*Abbildung 3.2: Challenge-Response Protocols with Shared Secrets*

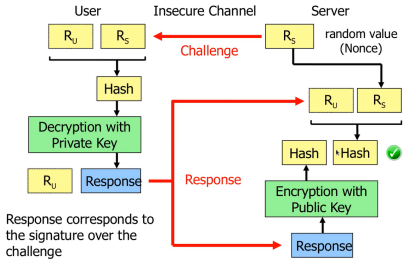


*Abbildung 3.3: Challenge-Response Protocols with Digital Signature*

### 3.6.1 Cracking Password used in Challenge-Response Protocol

- **Online Attack**: Der Angreifer probiert unterschiedliche Passwörter direkt beim Service aus, was jedoch sehr unwahrscheinlich ist, da es immer eine neue Challenge gibt
- **Offline Attack**: Ich zeichne erfolgreiche Authentifikationen auf und probiere nun vers. Passwörter in die keyed hash function zu stecken um die korrekte Response zu erhalten.

⇒ Auch hier kann Key-Streching ausgeführt werden um das Leben des Angreifers zu erschweren

# Kapitel 4

# Vorlesung 6 - Sichere Kommunikation mit Fokus auf Layer 2

## 4.1 Sie kennen was mit sicheren Kommunikationsprotokoll erreicht und nicht erreicht werden kann

### 4.1.1 Grundlegendes

Die fundementalen Internet Protokollen (IP, TCP, etc) stellen keine Sicherheitsmassnahmen zur Verfügung
Daraus resultieren verschiedene Gefahren in der Internet-Kommunikation

- Eavesdropping → Lesen von Daten bei der Übermittlung
- Manipulating Data → Ändern von Daten bei der Übermittlung
- Injecting Data → Hinzufügen von Daten bei einer bestehenden Kommunikationsbeziehung
- Replaying Data → Wiederversenden von bereits gesendeten Daten
- Getting unauthorised access → die Verwendung eines Systems ohne Privilegien, z.B. nach der Erlangung von Berechtigungsnachweisen durch Lauschangriff
- Masquerading → Vorgaukeln, dass man eine andere Person oder System ist

### 4.1.2 Hauptziel der sicheren Kommunikation

Um diese Ziele zu erreichen, werden kryptografische Protokolle eingesetzt

- Confidentiality → Nur der Empfänger der Kommunikation kann die Daten Lesen
- Integrity → Der Empfänger kann feststellen, falls Daten manipuliert wurden
- Authenticity → Das Vorgaukeln, dass man jemand anders ist, kann nicht passieren

**Minor Goals - werden selten gefordert**

- Non-Repudiation → Der Empfänger kann nicht vernein, dass er Daten erhalten bzw. versendet hat
- Anonymity → Der Empfänger (oder auch Sender) können sich gegenseitig nicht identifzieren

### 4.1.3 Limitationen von Sicherheitsprotokollen

- Software Schwachstellen werden dadurch nicht geschützt
- Schützt einem nicht vor Malware
- Schützt einem nicht vor DoS Attacken

⇒ Da die Sicherheitsprotokolle zunehmend komplexer werden (Implementation oder Protokoll Unschönheiten), sind diese insicher jenachdem unsicher → in der Regel sollten keine eigenen Sicherheitsprotokolle geschrieben werden

## 4.2 Sie verstehen weshalb die sichere Kommunikation auf verschiedenen OSI-Layer stattfindet und haben einen Überblick über die verschiedenen Protokollen auf den verschiedenen Layern

- Die Protokolle kommen auf allen unterschiedlichen Layern zum Einsatz, je nach Anwendungsgebiet
- Typerischerweise ist die Implementation auf den höheren Layern einfacher, je höher im Layer, desto spezifischer für einen konkreten Anwendungsfall
- Auf den tieferen Layern, braucht man häufig noch die Absicherung von Hardware-Geräten, kann viel allgemeiner eingesetzt werden

| Layer | Protocols |
|---|---|
| Application Layer | S/MIME, PGP |
| Transport Layer | SSL/TLS |
| Network Layer | IPsec |
| Data Link Layer | EAP, 802.1x, WEP, WPA, WPA2/802.11i |
| Physical Layer | Quantum Cryptography (Appendix only) |

*Abbildung 4.1: Überblick über die verschiedenen Protokollen pro Layer*

## 4.3 Sie verstehen die erweiterbare Authenifzierungsprotokolle und wie diese nach dem IEEE 802.1x Standard funktionieren, allen vor an für Portbasierte Netzwerkzugriffskontrolle aktivieren

- Die Verschlüsselung auf Layer 2 haben bei draht-Verbindungen haben fast keine Singifikanz
- Jedoch ist eine hohe Singifikanz spannend bei der draht- / sowie drahtlosen Verbindung betrf. der Authentifikation → Exentsible Authentication Protocol (EAP), wobei EAP weder eine Authentifikation, noch eine erweiterbare Möglichkeit anbieten (hat daher eine falschen Namen erhalten)

### 4.3.1 EAP

**Wichtig:** EAP ist weder ein Protokoll noch erweiterbar → es ist mehr eine Art und Weise wie Nachrichten verpackt werden

- Ein EAP-Paket packt andere Protokolle ein

- ist mehr ein Standard
- hat 50 unterschiedliche Authentifkation Methoden → wird im Type-Field festgelegt

Darauf aufbauend gibt es nun *IEEE802.1x: Port-based Network Access Control* → eines der wichtigsten Anwendungen von EAP

- LAN-Ports sind nicht per Default nicht offen
- Bevor dieser Port geöffnet wird, muss diese Connection authenifiziert werden
- Dabei wird EAP als Authentifizierungsprotokoll verwendet
- IEEE 802.1x definiert wie EAP die Nachricht enkapsuliert innerhalb des Ethernet / WLAN-Frames → EAP over LAN (EAPOL)

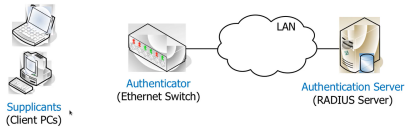


*Abbildung 4.2: Schema bei einem IEEE802.1x Aufbau*

### 4.3.2 IEEE 802.1x mit EAP-TLS Authentikation

Dabei wir jeder Nachricht einfach mit dem EAP eingepackt inkl. dem TLS-Handshake

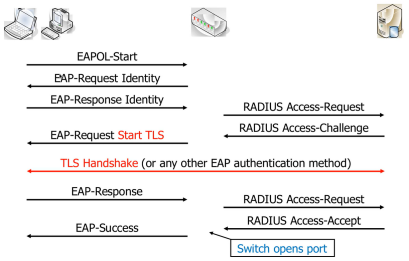


*Abbildung 4.3: Nachrichtenverlauf eines IEEE 802.1x mit EAP-TLS Authentikation*

Für den Switch sind einzig die RADIUS-Nachrichten von Relevanz, vor allem die RADIUS Access-Accept → anschliessend wird der Port geöffnet

## 4.4 Sie verstehen die Funktionalität der verschiedenen Sicherheitsmechanismen für WLAN und können die Sicherheitseigenschaften und -probleme erläutern

### 4.4.1 Security of WLAN

- Es ist kein Kabel vorhanden, so kann jeder in der Nähe solche Pakete abfangen / sniffen → Pakete müssen verschlüsselt werden
- Nur gewisse Personen sollen Zugriff haben
- Dies soll alles auf dem Layer 2 geschehen, so dass jeder Datenaustausch zwischen Clients und Access-Point geschützt ist
- IEEE 802.11i (auch bekannt unter WPA2) as the official successor of WEP (Wired Equivalent Privacy)
- Es gibt immer noch eine wenige Access-Points welche offen sind

### 4.4.2 WLAN Sicherheit mit Wired Equivalent Privacy (WEP)

- Access-Point und alle Clients teilen sich um einen vorkonfigurierten Langzeit-Schlüssel
- Dieser Schlüssel wird für das verschlüsseln der einzelnen Frames verwendet → Alle Clients nutzen den selben Key und können somit den Datentransfer von allen Clients mitlesen.
- Die Länge des Schlüssels ist entweder 40 oder 104 Bits mit der RC4-Verschlüsselung → erreichen sicherlich nicht den gewünschten Workfactor, des Weiteren wird RC4 als geknackt gekennzeichnet

#### Frame Protection

Für jeden individuellen Frame, passiert folgendes:

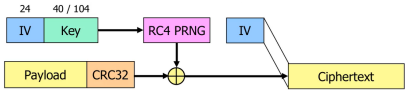


*Abbildung 4.4: Frame Protection*

- CRC Checksume schützt die payload Integrität
- 40-Bits SChlüssel sind eine schlechte Wahl
- 104-Bits grundsätzlich gut, jedoch immer noch nicht total sicher
- Grund dafür spätestens nach $2^24$ wiederholt sich der Schlüssel nochmals → wir attackieren nicht den Schlüssel, sondern die verschiedenen Keystreams

#### erster Angriff

Wir sniffen alle Frames und warten bis der IV sich wiederholt → bei zufälligem IV geschieht dies nach 4'823 Frames (Birthday Paradox), schliemmsten Fall muss man alle $2^24$ abwarten.
Wenn man zwei Ciphertext mit dem gleichen IV ermittelt hat, kann man folgendes durchführen:

$$c_1 = p_1 \oplus ks,\ c_2 = p_2 \oplus ks \rightarrow c_1 \oplus c_2 = p_1 \oplus p_2$$

*Abbildung 4.5: Die Formel für das Erkennen des Ciphertexts*

⇒ Durch den Plaintext erhalten wir in aller Regel viel mehr Struktur (bspw. Englischer-Text oder TCP-Paket). Daraus kann man dann sämtliche Keystreams ausrechnen → Jedoch hat man dann mit ein paar wenigen 100 GBs die Möglichkeit sämtliche Texte zu entschlüssel ohne, dass man den Schlüssel kennt.

#### zweiter Angriff

Angriff auf die Schwachstellen von RC4

- Wenn man nur eine kleine Anzahl von Bits kennt, kann man den Keystream mit einer Wsk von 50 Prozent vorhersagen
- WEP verwendet IV, dadurch kennen wir schon die ersten 24 Bits des RC4 Keys

⇒ WEB ist total unsicher

#### Warum ist CRC kein guter MAC

- Nur Verschlüsselung ist keine gute Wahl und sollte mit Integrität-Schutz erweitert werden
- WEP stellt ein verschlüsselter CRC zur Verfügung, was jedoch nciht von allen Attacken schützt

### 4.4.3 Wi-Fi Protected Access

Der neue Standard von IEEE ging so lange, dass sich eine Gewerkschaft mit der Lösung auseinandersetzte.
Die Client-Authentifikation kann auf zwei Möglichkeiten stattfinden:

- Port-based Network access control gemäss IEEE 802.1x
- Pre-shared Key (PSK) → Vor allem zuhause, wenn man das WLAN-PW verteilt

Der Client und den Access-Point wird richtig verschlüsselt:

- zwei 128-Bit unicast keys für die Verschlüsselung und Integritätsschutz (eindeutig pro Client und Session)
- zwei 128-bit broadcast keys für die Verschlüsselung und Integritätsschutz (gleich für alle Clients)
- periodic re-keying, normalerweise gibt es nach einer Stunde einen neuen Schlüssel

Beide Standards definieren zwei Verschlüsselungs-Modes → TKIP und CCMP
**Temporal Key Integrity Protocol (TKIP)**: tbd **CCMP**: tbd

#### TKIP Security

- den Einsatz von MAC für den Integritätsschutz ist eine gute Idee
- TKIP verwendet jedoch immer noch den RC4-Alogrithmus
- TKIP verwendet Michael als MAC-Alogrithmus

#### CCMP Security

#### Counter Mode CBC-MAC Protocol

- WPA/WPA2 verwendet CCMP mit AES und 128-Bits Key
- WPA2 unterstützt immer CCMP
- Aktuell ist noch keine Unsicherheit betreffend CCMP

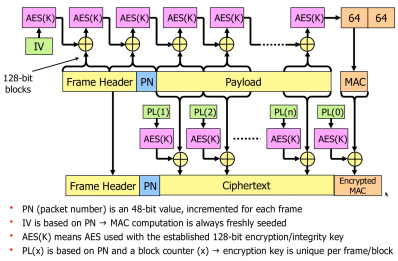


*Abbildung 4.6: Ablauf von CCMP*

# Kapitel 5

# Vorlesung 7 - Firewalls

## 5.1 Sie wissen was eine Firewall ist, und was sie (nicht) kann

- Eine Firewall ist ein Netzwerkgeräte, welche zwischen zwei oder mehreren Netzwerken kontrolliert
- Die Firewall ist immer auch ein Router, welcher die Pakete weiterleitet
- Grundlage hierfür ist eine Security Police
- Basierend auf der Security Police, wird eine oder mehrere Firewalls installiert

**Was kann man machen?**

- Zugriff kontrollieren auf die verschiedenen Internet-Seiten
- Zugriff kontrollieren vom Internet auf die internen Computer und Services
- Bösartigen Traffic unterbinden (bspw. HTTP-Traffic für SQL-Injection)
- Einer der wichtigsten Komponenten
- Blockieren von nicht-gewollten-Traffic
- Struktur von Aussen verstecken

**Wozu nicht?**

- Teilt implizit die Welt in Innen und Aussen auf, solange die Angreifer von Aussen sind, funktioniert. Sobald sie im Inneren des Netzwerk sind, schützen sie nicht mehr
- Schützt sie nicht vor Attacken auf dem Applikation Layer

## 5.2 Sie verstehen den Unterschied zwischen packet-filtering firewalls und application layer firewalls and wissen den grundsätzlichen Einsatz von beiden Typen

⇒ In der Praxis verwendet man beide Arten der Firewalls um eine möglichst gute Sicherheit zu gewährleisten

### 5.2.1 packet-filtering firewalls

- meist genutzte Typ
- operiert auf dem Network-Layer (manchmal auch auf den Transport-Layer)
- Die Firewall erhält ein Paket und bevor es zum anderen Netzwerk weitergeleitet wird, begutachtet es das Paket
- end-to-end Kommunikation wird nicht aufgespalten
- meistens werden die headers inspiziert
- *Beispiel* Erlaube jedem Host im Netzwerk 160.85.37.0/24 mit Host 160.85.215.20 zu kommunizieren, dies jedoch nur über Port 80
- **Vorteil** Sehr schnell, da es nur den Layer 3 und 4 die Protokol-Header überprüft werden muss
- **Limitationen** Man kann nur beschränkten, wer mit wem reden kann - jedoch kann man den Inhalt, welcher kommuniziert wird, nicht beschränken
- Gibt viele kommerzielle Produkte auf dem Markt

**Einsatzgebiet**

**Szenario I:**

- Firewall hat drei Netzwerk Interfaces, drei Netzwerke
- Internes Netzwerk mit Computern und internen Services (print server, file server etc.)
- Demilitarized Zone (DMZ) mit public services (web, email etc.)
- externes Netzwerk, alles andere
- **Vorteil** Nur eine Firewall notwendig
- **Nachteil** Firewall ist single point of failure und ein Angreifer muss nur eine Firewall überwinden

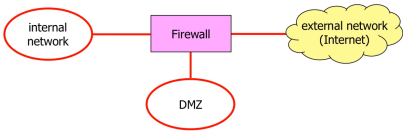


*Abbildung 5.1: Schema des ersten Packet Filterin Szenario*

**Szenario II:**

- Zwei Firewall, mit jeweils zwei Netzwerk interfaces, immer noch drei Netzwerk
- DMZ wird auf zwei Firewalls aufgeteilt
- **Vorteile**: Zwei Firewalls (idealerweise von unterschiedlichen Herstellern) vom externen zum internen Netzwerk
- **Nachteile**: Teurer und es braucht mehr Expertise für die Konfiguration


*Abbildung 5.2: Schema des zweiten Packet Filterin Szenario*

### 5.2.2 application layer firewalls

- Arbeitet auf dem Application Layer (oft auch Web-Applikation oder Proxy Firewall) → wird oft verwendet um die Webseite zu schützen
- spaltet die End-To-End-Kommunikation
- Inspektiert application layer daten gemäss den Regeln, sofern die Daten i.O. sind, wird es weitergeleitet
- *Beispiel* Überprüfen, ob HTTP-Request JavaScript-Code enthält und falls dem der Fall ist, dies blockieren
- **Vorteil** Man hat deep inspection für alle Daten die ausgetauscht werden
- **Limitationen** Relativ langsam, es gibt Schwierigkeiten für verschlüsselte Daten, Normalerweise für einen Typ von Applikation

**Einsatzgebiet**

**Web Application Firewall (WAF)**:

- zwei Firewalls mit drei Netzwerken
- um den Zugriff auf den Webserver(mit Weltkugel) zu reglementieren, wird FW2 verwendet um nur HTTP-Request für die WAF zuzulassen
- DNS muss so konfiguriert dass die Namensauflösung auf die WAF zeigt
- Zugriff vom externen ins interne Netzwerk wird mittels den blauen Linien dargestellt

## 5.3 Sie verstehen den Unterschied zwischen stateless und statefull firewalls

### 5.3.1 stateless Firewalls

- Jedes IP-Paket wird komplett isoliert von allen anderen behandelt
- die Firewall merkt sich nicht welche Verbindungen bestehen
- **Nachteil:** Firewalls sind meistens offener als man es braucht, nur sehr eingeschränkten Support für komplexe Protokolle (bspw. FTP)

### 5.3.2 Stateful Firewalls

- heutzutage sind beinahe alle Firewalls statefull
- Firewall merkt sich ob bestimmte Pakete zu bestimmten Sessions gehören
- typische Informationen: Protokoll, source und destination der IP-Adresse, Ports, Sessiondauer, Protokol-Phase (TCP)
- **Vorteile** Braucht weniger Regel (ca. die Hälfte), Return-traffic wird nur noch on-demand erlaubt, Support für komplexe Protokolle (bspw. FTP)
- netfilter und nftables unterstützt Stateful Packet-Filtering

## 5.4 Sie kennen das fundamentale Konzept von der netfilter/nftables-Architektur und können nftables anwenden um eine einfache Firewall zu konfigurieren

**netfilter** ist ein uralters Programm, welches sehr viele Möglichkeiten bietet:

- Erlaubt Packet-Filtering
- Erlaubt Network Adress Translation (NAT)
- erlaubt general packet mangling
- ist ein Mechanismus der den Zugriff auf die Pakete erlaubt und diese entsprechend analysiert, modifiziert, extrahiert oder löscht

**nftables** ist ein darauf aufbauendes Netzwerk Klassifizierungs-Tool

- Packet-Filtering Firewall kann damit implementiert werden
- Network Address Translation kann damit realisiert werden

**nft** ist der Name des Command Line Tools um nftables zu konfigurieren

### 5.4.1 Linux Netfilter

- der Kernel hat eine bestimmte Anzahl an Hooks, welche an unterschiedliche Punkte zum Zuge kommt
- bspw. vor dem INGRESS oder nach dem POSTROUTING, des Weiteren bei INPUT und OUTPUT

Es gibt gesamthaft acht Hooks:

1. INGRESS, da kommt da Paket frisch von der Netzwerkkarte
2. PREROUTING, Paket wurde entgegengenommen und Checksummen wurden geprüft aber noch keine Verarbeitung (ist noch nicht klar, ob das Paket an ein anderes Netzwerk weitergeleitet oder intern verwendet)
3. INPUT, falls bei der Routing Decision entschieden wird, dass das Paket für den lokalen Zweck ist
4. FORWARD, falls bei der Routing Decision entschieden wird, dass das Paket für ein anderes Paket notwendig ist
5. POSTROUTING, das Paket wird an die Netzwerkkarte ausgeliefert
6. OUTPUT, kommt vom lokalen Prozess und geht an die Netzwerkkarte

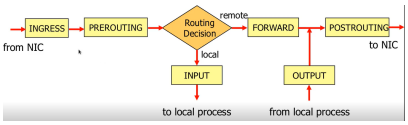


*Abbildung 5.3: Die 8 Hooks beim Netfilter*

### 5.4.2 nftables Rules

- nftables Regeln haben einen Classification Part und einen oder mehrere Action-Parts
- **Classification:** Auf welche Pakete soll diese Regel angewendet werden
- **Action:** Sagt was mit diesen Pakete gemacht werden soll
- accept-Action die Verarbeitung des Paket wird weitergeführt
- drop-Action Die Verarbeitung wurde gestoppt, jedoch wird auch nichts damit gemacht, sprich der Sender erhält keine Mitteilung
- reject-Action Verarbeitung des Pakets wurde gestopppt, sender erhält eine Meldung mittels ICMP-Paket → Standardtext *port unreachable*, diese Nachricht kann auch noch angepasst werden bspw. durch *... reject with icmp type host-prohibited*
- jump-Action Man möchte die Verarbeitung eines Pakets an einer anderen Stelle fortsetzen
- (häufige) Empfehlung bei Pakete welche man nicht verwenden möchte sollte man drop verwenden → Angreifer erkennen auch bei Drop, dass es eine Firewall gibt
- Sysadmin möchten jedoch eine Antwort erhalten und daher ist die drop dann nicht
- ip steht für IPv4 Pakete
- ip6 steht für IPv6
- daddr steht für destination adress

**Beispiel Befehle**

- *ip daddr 8.8.8.8 drop* → sämtliche Bedienungen müssen eintreffen, dass sie zum Zuge kommen
- *ip6 nexthdr tcp accept* → akzeptiert alle IPv6 Pakete mit einem TCP Header
- *ip6 nexthdr != tcp accept*
- *iifname eth2 accept* → Akzeptiert alle Pakete welche vom interface eth2 kommen
- *icmp type echo-request limit rate 10/second accept* → Maximal 10 ICMP echo-request (ping) limitieren

### 5.4.3 nftables chain

- Regeln sind in Chains organisiert
- Base-chain in der Table gibt es einen Hook
- non-base-chain nicht mit einem Hook
- Classification wird von open nach unten durchgeführt, bis eine Regel ausgeführt. Sobald eine zutrifft, werden die anderen nicht ausgeführt
- Falls keine Regel zutrifft, dann kommt eine policy zum Zuge
- default-policy ist accept
- Chains haben einen Typ, hier verwenden wir grundsätzlich *filter*
- Chains haben auch eine entsprechende Prioritäten

**Beispiel:**

```
chain ssh-traffic  type filter hook input priority 0; policy drop;
tcp dport ssh count accept
```

- *type filter*: Chain wird für Paket-Filtering verwendet
- *hook input*: ist zum INPUT-hook angegliedert (base chain)
- *priority 0*: hat die Priorität 0 (mittlerer Wert)
- *policy drop*: unklassifizierte Pakete werden gedropped

⇒ Nur Pakete an SSH

### 5.4.4 nftables Tables

- chains werden in TABLES zusammengefasst
- Tables fassen alle Chains der gleichen Art zusammengefasse
- diese Paket-Tyoes werden auch Adress families genannt (ip = nur IPv4; ip6 = nur IPv6; inet = IPv4 und IPv6)

### 5.4.5 nftables Ruleset and Scripting

```
#!/usr/sbin/nft -f
flush ruleset          # Clean all tables, chains
table inet myinput {
  chain tcp-traffic {
    type filter hook input priority 0; policy drop;
    tcp dport { https, http } jump http-traffic
  }
  chain http-traffic { # non-base-chain
    type filter; policy drop;
    count accept
  }
} The ruleset brings all tables together
```

*Abbildung 5.4: Beispielcode eines Rulesets*

- *#/usr/sbin/nft -f* wird für die Ausführung verwendet
- File soll ausführbar sein
- es gilt eine atomare Ausführung (entweder es wird alles ausgeführt oder nichts)
- *flush ruleset* gesamte Ruleset wird atomar ersetzt
- mittels *define* können Variable definiert werden, diese können dann im Code mittels $ + Variablenname verwendet werden

### 5.4.6 Vorgehen beim Aufbau eines nftables

1. Belegung von Variablen mittels *define*. Dabei haben wir die präfixe i für internes Netzwerk, d für DMZ, e für externes Netzwerk. Postfix ifc für interface, net für Netzwerk, addr = Addreess des Interfaces Richtung Firewall
2. INPUT, OUTPUT und FORWARD Hooks muss je eine Chain erarbeitet werden (myinput, myforward, myoutput) das gesamte resultiert in der table myfilter; Alle policies sind drop.

## 5.5 Sie verstehen die Absicht eines Port-Scanners und können diesen mittels port scan und nmap anwenden und interpretieren

- Ist eine Technik um herauszufinden welche Dienste auf einem Remote-Dienst alles laufen
- wird häufig von Angreifer verwendet um sich einen ersten Überblick zu verschaffen
- Man überprüft ob der Host verfügbar ist (Ping)

- Aufbauen einer TCP Verbindung, falls das funktioniert handelt es sich um einen offenen Port. Bei einer TCP RST Antwort handelt es sich um einen geschlossenen Map
- populärster Port-Scanner ist nmap

**Beispiel nmap**
Überprüfen ob der TCP Port 80 bei der ZHAW offen ist: *nmap -80 www.zhaw.ch*

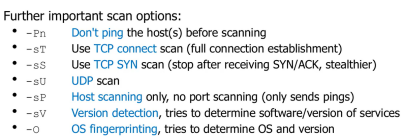
Further important scan options:
- `-Pn` Don't ping the host(s) before scanning
- `-sT` Use TCP connect scan (full connection establishment)
- `-sS` Use TCP SYN scan (stop after receiving SYN/ACK, stealthier)
- `-sU` UDP scan
- `-sP` Host scanning only, no port scanning (only sends pings)
- `-sV` Version detection, tries to determine software/version of services
- `-O` OS fingerprinting, tries to determine OS and version

*Abbildung 5.5: Weitere Befehle bei nmap*