

# Computersystemsicherheit

*Lena Thuy Trang Vo*

*Wintersemester 2024/25*

## Inhaltsverzeichnis

---

<b>1</b>	<b>Thema 1: Einführung</b>	<b>3</b>
1.1	Themenübersicht . . . . .	3
1.2	Begriffsbedeutung . . . . .	3
1.2.1	Was bedeutet Sicherheit? . . . . .	3
1.2.2	Sicherheitseigenschaften . . . . .	3
1.2.3	Wie können wir uns schützen? Allgemeine Sicherheitsprinzipien . . . . .	4
<b>2</b>	<b>Thema 2: Einführung Kryptographie</b>	<b>6</b>
2.1	Themenübersicht . . . . .	6
2.2	Was ist Kryptographie? . . . . .	6
2.3	Klassische vs. moderne Kryptographie . . . . .	6
2.3.1	Klassische Kryptographie . . . . .	6
2.3.2	Moderne Kryptographie . . . . .	6
2.4	Ziele der Kryptographie . . . . .	6
2.5	Schlüssel . . . . .	7
2.6	Kerkhoff'sches Prinzip . . . . .	7
2.7	Klassische Chiffren . . . . .	7
2.8	Moderne Kryptographie . . . . .	7
2.8.1	Anwendung Heute . . . . .	7
2.8.2	Ansatz der modernen Kryptographie . . . . .	8
2.8.3	Was sind kryptographische Annahmen? . . . . .	8
2.8.4	Kryptographische Primitive und Konstruktionen . . . . .	8
<b>3</b>	<b>Thema 3: Symmetrische Kryptographie</b>	<b>8</b>
3.1	Themenübersicht . . . . .	8
3.2	Definition Symmetrischer Chiffren . . . . .	9
3.2.1	Schutzziel . . . . .	9
3.2.2	Funktionale Definition . . . . .	9
3.2.3	Sicherheitsdefinition . . . . .	9
3.2.4	Sicherheitsspiel (IND-CPA) . . . . .	10
3.2.5	Bietet IND-CPA die stärkste Sicherheit? . . . . .	10
3.2.6	Stärkere Sicherheit: IND-CCA . . . . .	10
3.2.7	Unterschied zwischen IND-CPA und IND-CCA . . . . .	11
3.3	One-Time-Pad Verschlüsselung . . . . .	11
3.3.1	One-Time-Pad . . . . .	11
3.3.2	Sicherheit . . . . .	12
3.3.3	Schlüssel nur einmal verwenden . . . . .	12
3.3.4	Venona-Projekt: Risiken von One-Time-Pads . . . . .	12
3.3.5	Nachteile von One-Time-Pad . . . . .	13
<b>4</b>	<b>Übung 1</b>	<b>13</b>
4.1	Aufgabe 1: Wissensfragen . . . . .	13
4.2	Aufgabe 2: IND-CPA vs. IND-CCA . . . . .	14
<b>5</b>	<b>Thema 3: Symmetrische Kryptographie - Teil</b>	<b>14</b>
5.1	Data Encryption Standard (DES) . . . . .	14
5.1.1	DES: Angriffe . . . . .	15
5.1.2	Triple-DES . . . . .	15
5.2	Advanced Encryption Standard (AES) . . . . .	15
5.2.1	Angriffe in der Praxis . . . . .	16
5.3	Modes of Operation . . . . .	16
5.3.1	Electronic Code Book (ECB) Modus . . . . .	16
5.3.2	Cipher Block Chaining (CBC) Modus . . . . .	17
5.3.3	Counter Modus (CTR) . . . . .	19

5.4	Kryptographische Hashfunktionen . . . . .	20
5.4.1	Hashfunktionen . . . . .	20
5.4.2	Sicherheitsdefinitionen . . . . .	20
5.5	Message Authentication Codes . . . . .	20
5.5.1	Sicherheitsdefinition . . . . .	21
5.5.2	CBC-MAC . . . . .	22

## Thema 1: Einführung

### Themenübersicht

- Begriffsbedeutung
- Warum ist Sicherheit wichtig?
- Fallbeispiele für Sicherheitsvorfälle
- Sicherheitsprinzipien
  - Kenne die Angreifer
  - Berücksichtige menschliche Faktoren
  - Sicherheit ist wirtschaftliche Abwägung
  - Detektieren falls nicht verhinderbar
  - Defense in depth (gestaffelte Verteidigung)
  - Fail-safe Standard

### Begriffsbedeutung

#### 1.2.1 Was bedeutet Sicherheit?

##### Betriebssicherheit / Safety

- Schutz gegen Fehler/Unfälle
- Fehler meist unabsichtlich verursacht
- Gegenmaßnahme: Verifikation, Testen

##### Angriffssicherheit/Security

- Schutz gegen worst-case Angreifer
- meist Schadabsicht
- Verifikation und Testen hilft wenig

#### Security und Safety können im Konflikt zueinander stehen

Beispiel Notausgang

- **Safety:** Im Notfall können Personen aus dem Gebäude
- **Security:** Für Gebäudeschutz am besten gar keine Tür

#### 1.2.2 Sicherheitseigenschaften

Sicherheit kann vieles bedeuten...

1. **Vertraulichkeit** von Daten/Nachrichten (z.B. von Whatsapp Nachrichten)
  - Sicherstellung, dass das System keine unautorisierte Informationsgewinnung ermöglicht
2. **Anonymität** von Benutzern (z.B. beim Surfen im Web)
3. **Integrität** von Daten/Berechnungen (z.B. bei Überweisungen im Online-Banking)
  - Gewährleistung, dass nicht autorisierte Subjekte ein Objekt nicht unbemerkt ändern können
4. **Authentizität** von Dateien (z.B. Software-Updates)
  - Echtheit und Glaubwürdigkeit eines Objektes, die kryptografisch überprüfbar ist
5. **Verfügbarkeit** von Diensten (z.B. des Stromnetzes)
  - Gewährleistung, dass autorisierte Subjekte nicht in der Funktionalität beeinträchtigt werden

### 1.2.3 Wie können wir uns schützen? Allgemeine Sicherheitsprinzipien

#### Sicherheitsprinzipien

1. Kenne die Angreifer
2. Berücksichtige menschliche Faktoren
3. wirtschaftliche Faktoren beeinflussen Sicherheit
4. Detektieren falls nicht verhinderbar
5. Defense in depth (gestaffelte Verteidigung)
6. Fail-safe Standards

#### 1. Kenne die Angreifer

Um ein effektives **Bedrohungsmodell** zu entwickeln, ist es wichtig, die **potenziellen Angreifer** und deren **Motivationen** zu verstehen.

##### Ressourcen:

- Individuum
- Organisierte Gruppen
- Terroristen
- staatlich geförderte Organisationen

##### Motivation:

- Geld
- politische Maßnahmen
- Vergeltung
- aus Spaß

#### Annahmen über Angreifer sind schwer zu treffen

- **rechtzeitiges Erkennen von Angriffen schwierig:** Angreifer kann unbemerkt mit dem System interagieren
- **Angreifer kennt das System:** Welches Betriebssystem wird verwendet, welche Hardware? (kennt Schwachstellen)
- **Kann Glück haben:** bei Chance 1:1.000.000 kann der Angreifer es 1.000.000 mal Probieren

#### 2. Berücksichtige menschliche Faktoren

Einschränkung der Sicherheit durch menschliches Verhalten möglich

- als **Benutzer:in**
  - neigen dazu, Sicherheitsmechanismen zu umgehen, wenn diese die Nutzung erschweren
  - Beispiel: Wahl einfacher und wiederverwendeter Passwörter
- als **Programmierer:in**
  - Programmierer können Fehler machen, die Sicherheitslücken schaffen
  - benutzen Tools, die erlauben Fehler zu machen (z.B. Sprache ohne Typsicherheit)
- als **Angreifer:in**

- Angreifer nutzen oft menschliche Eigenschaften wie Vertrauen oder Leichtgläubigkeit aus, um Informationen zu stehlen oder Zugang zu Systemen zu erlangen (Social Engineering)

Ergo: alle verwendeten Tools und Systeme sollten narrensicher sein.

### 3. wirtschaftliche Faktoren beeinflussen Sicherheit

- organisierte Cyberkriminalität nimmt zu
- Angriffsziele von organisierter Cyberkriminalität: **wirtschaftliche Interessen**
  - sei es zur direkten finanziellen Bereicherung oder um einem Wettbewerber oder Land zu schaden
- aus Sicht der angreifenden Partei:
  - Angriff teurer als Belohnung → kein Angriffsversuch
- aus Sicht der verteidigenden Partei:
  - viel Sicherheit kostet viel Geld
  - Abwägung zwischen Kosten-/Nutzen
    - \* Nutzen der Sicherheitsmaßnahmen proportional zu Kosten eines erfolgreichen Angriffs

### 4. Detektieren, falls nicht verhinderbar

1. **Abschrecken:** Einen Angriff abschrecken, bevor dieser stattfindet.
2. **Verhindern:** Falls Angriff stattfindet, verhindere dessen Erfolg
3. **Detektieren:** Stelle fest, falls ein Angriff stattgefunden hat
  - Falls nicht verhinderbar, dann wenigstens feststellen, dass ein Angriff stattgefunden hat
  - es ist essenziell, ihn schnell zu erkennen, um den Schaden zu minimieren
4. **Reagieren:** Reaktion auf stattgefundenen Angriff
  - Detektion ohne Reaktion ist nutzlos: Es ist entscheidend, nach der Erkennung eines Angriffs sofortige Maßnahmen zu ergreifen, um weitere Schäden zu verhindern.

### 5. Defense in Depth

- verschiedene Sicherheitsmaßnahmen implementieren
- schichtweiser Aufbau
  - Sicherheitsmaßnahmen übereinander legen, sodass ein Angreifer alle Schichten durchbrechen muss, um erfolgreich zu sein.
- Sicherheit ist oft weniger als die Summe aller Teile
  - Trotz der Vielzahl an Schutzmaßnahmen kann die Sicherheit oft nur so stark sein wie das schwächste Glied in der Kette.

### 6. Fail-Safe Standards

Dieses Prinzip sorgt dafür, dass ein System bei einem **Ausfall** oder einer Anomalie in einen Zustand übergeht, der den **geringstmöglichen Schaden** verursacht.

**Beispiele:**

**mechanisches Zugsignal:**

Bei einem mechanischen Zugsignal fällt das Signal auf "Halt", wenn das Zugseil reißt. Dies stellt sicher, dass Züge bei einem technischen Defekt automatisch gestoppt werden und keine Gefahr entsteht.

**elektronisches Nummernschloss:**

Bei einem elektronischen Nummernschloss ist es nicht immer einfach zu entscheiden, was der sichere Zustand ist. Bei einem Stromausfall könnte das Schloss entweder offen bleiben, um den Zugang zu ermöglichen, oder geschlossen bleiben, um unbefugten Zutritt zu verhindern. Die Entscheidung hängt von der spezifischen Anwendung und den damit verbundenen Risiken ab.

## Thema 2: Einführung Kryptographie

---

### Themenübersicht

- Was ist Kryptographie?
- Ziele der Kryptographie
- Klassische Chiffren
- Ansätze der modernen Kryptographie

### Was ist Kryptographie?

Kryptographie ist die Wissenschaft der **Verschlüsselung und Entschlüsselung** von Informationen. Sie dient dazu, Daten und Kommunikation vor unbefugtem Zugriff und Manipulation zu schützen.

- unzählige Anwendungen in der Praxis
  - grundlegender Baustein jedes Sicherheitssystems
  - z.B. ohne Kryptographie keine Sicherheit im Internet

### Klassische vs. moderne Kryptographie

#### 2.3.1 Klassische Kryptographie

- bezieht sich auf ältere Verschlüsselungsmethoden, die hauptsächlich zur **sicheren Kommunikation über unsichere Kanäle** verwendet wurden
- Hauptanwendung: Militär

#### 2.3.2 Moderne Kryptographie

- bietet **starke Sicherheitsgarantien für Daten und Berechnungen**, selbst in Anwesenheit eines Angreifers
- wird in nahezu jedem Lebensbereich angewendet

### Ziele der Kryptographie

1. **Vertraulichkeit:** Angreifer kann den Inhalt der Nachricht **nicht lernen**
  - nur autorisierte Parteien haben Zugang zu den Informationen, während unbefugte Dritte ausgeschlossen werden
2. **Integrität:** Angreifer kann Nachricht nicht ändern, ohne dass Änderung bekannt wird
3. **Authentizität:** Angreifer kann nicht die Nachricht von einer anderen Person stammen lassen
  - sicherstellen, dass der Absender einer Nachricht wirklich derjenige ist, für den er sich ausgibt

## Schlüssel

- **Kurzer Schlüssel:** Kryptoverfahren verwenden zufällig gewählten, kurzen Schlüssel
- **Symmetrische Kryptographie:** Gleicher Schlüssel zum Ver- und Entschlüsseln
- **Asymmetrische Kryptographie:**
  - öffentlicher Schlüssel: z.B. im Internet veröffentlicht
  - geheimer Schlüssel: nur einzelnen Nutzern eines Kryptoverfahren bekannt

## Kerckhoff'sches Prinzip

- ein Kryptoverfahren soll sicher bleiben, selbst wenn der Angreifer den Kryptoalgorithmus kennt
- alles ist öffentlich außer ein **kurzer Schlüssel  $k$** , der **zufällig** gewählt wurde

## Warum Kerckhoff?

- in kommerziell eingesetzten Produkten ist es schwarz, die Spezifikation geheim zu halten
  - **Reverse Engineering:** Algorithmen können rekonstruiert werden
- kurze Schlüssel sind einfacher zu **schützen**, zu **erzeugen** und **auszutauschen**
- die Sicherheit des Designs kann **öffentlich analysiert** werden

**Kerckhoff's Grundsatz verletzt**  $\implies$  Sicherheit bei Verschleierung

## Klassische Chiffren

### Shift-Chiffre

- zyklischer Shift jedes Buchstaben um  **$k$**  Stellen im Alphabet
- Caesars Chiffre:  **$k = 3$**
- alle Schlüssel ausprobieren bei Angriff  $\longrightarrow$  **Brute-Force-Angriff**

### Erweiterte Shift-Chiffre

Benutze **Wort als Schlüssel** und verschiebe jeden Buchstaben, um die durch den Schlüssel gegebene **Differenz**

### Substitutionschiffre

- ist eine Erweiterung der Shift-Chiffre, bei der jeder Buchstabe des Alphabets durch einen anderen Buchstaben ersetzt wird, basierend auf einer **beliebigen Permutation** des Alphabets
- keine feste Verschiebung
- Anzahl an Schlüsseln:  $26! \approx 2^{88} \implies$  zu viele Schlüssel für ausprobieren

## Moderne Kryptographie

### 2.8.1 Anwendung Heute

- Sichere Kommunikation im Internet
- Digitale Zertifikate
- Sichere Datenspeicherung, z.B. für die Cloud
- Zugriffskontrolle, z.B. als Autoschlüssel
- E-Commerce und Online-Banking
- Digitale Signaturen
- Hashfunktionen
- ...



### 2.8.2 Ansatz der modernen Kryptographie

#### 1. Formale Definition:

- **Ziel des Angreifers:** z.B. bei Verschlüsselung sollte Angreifer nichts über Klartext lernen
- **Angreifermodell:** Was kann der Angreifer tun und sehen (z.B. Angreifer sieht nur Chiffretexte)

#### 2. Konstruktion:

- z.B. Konstruktion komplexer Kryptoverfahren aus einfachen Kryptoprimitiven

#### 3. Sicherheitsbeweis:

- **Annahme hält**  $\implies$  Kryptoverfahren ist sicher gemäß der formalen Definition (z.B. zahlentheoretische Annahme)
- **Reduktionsbeweis:** Angreifer gegen Kryptoverfahren  $\implies$  Annahme hält nicht

### 2.8.3 Was sind kryptographische Annahmen?

- Kryptoverfahren nutzen Annahmen, auf denen Sicherheit basiert
  - stellt sich heraus, dass **Annahme falsch ist** oder **effizient gelöst** werden kann, so wäre das Verfahren **nicht mehr sicher**
- **Einwegfunktion:** in eine Richtung einfach zu berechnen, in die andere praktisch unmöglich
- Annahme, praktisch unmöglich, Eingabe aus Ausgabe zu berechnen

#### Häufig genutzte Annahmen:

- Primfaktorzerlegung großer natürlicher Zahlen ist schwer
- Berechnung des Diskreten Logarithmus ist schwer
- Allgemein: Schwierigkeit mathematischer Probleme

### 2.8.4 Kryptographische Primitive und Konstruktionen

#### Kryptographische Primitive

- ist eine **abstrakte, fundamentale Funktion** mit **spezifischen kryptographischen Eigenschaften**
- **Beispiele:** Blockchiffren, Hashfunktionen, Digitale Signaturen etc.

#### Kryptographische Konstruktion

- beschreibt, wie **kryptographische Primitive instanziiert** und miteinander kombiniert werden, um **komplexe kryptographische Systeme** zu erstellen
- **Beispiele:** AES, SHA-256, Schnorr-Signaturen etc.

## Thema 3: Symmetrische Kryptographie

### Themenübersicht

- Was ist symmetrische Kryptographie?
- Definition Symmetrische Chiffre
- One-Time-Pad
- Block-Chiffren
- Modes of Operation

- Kryptographische Hashfunktionen
- Message Authentication Codes (MACs)
- Authenticated Encryption

**Symmetrische Kryptographie:** Es gibt nur **einen** Schlüssel für alle Algorithmen

### Definition Symmetrischer Chiffren

#### 3.2.1 Schutzziel

Welches kryptographische Schutzziel möchten wir mit symmetrischen Chiffren erreichen?

- **Vertraulichkeit:** Angreifer kann den Inhalt der Nachricht nicht lernen

#### 3.2.2 Funktionale Definition

- beschreibt **Input/Output-Verhalten** der Algorithmen
- Algorithmen: Gen, Enc, Dec

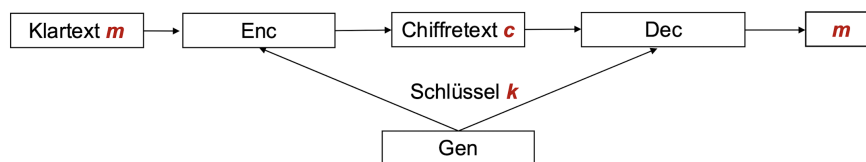


Abbildung 1: Funktionale Definition

- **Gen:** generiert einen zufälligen Schlüssel  $k$ , der später sowohl für die Verschlüsselung als auch für die Entschlüsselung verwendet wird
- **Enc (Verschlüsselung):** nimmt den Klartext  $m$  und Schlüssel  $k$  als Eingabe und erzeugt einen Chiffretext  $c$
- **Dec (Entschlüsselung):** nimmt den Chiffretext und denselben geheimen Schlüssel als Eingabe und stellt den ursprünglichen Text wieder her

#### Korrektheit:

Die Entschlüsselung eines gültigen Chiffretextes resultiert in die original verschlüsselte Nachricht

$$Dec(k, Enc(k, m)) = m \text{ für alle Nachrichten } m \text{ und Schlüssel } k \leftarrow Gen$$

**Effizienz:** Verschlüsselung und Entschlüsselung sind effizient ( 1 GB/s)

#### 3.2.3 Sicherheitsdefinition

- **Ziel des Angreifers:** Was ist ein erfolgreicher Angriff?
- **naive Option:** Angreifer lernt den Schlüssel  $k$  nicht
- in der **Kryptographie:** Angreifer lernt **nichts Neues** über  $m$

#### Angreifermodell:

- beschreibt die Fähigkeiten und Ressourcen des Angreifers
- Angreifer lernt **nur Chiffretext** (known ciphertext attack), z.B. durch Abhören des Kanals
- Angreifer lernt **Paare von Klartexten/Chiffretexten** (known plaintext/ciphertext attack), z.B. bestimmter Teil der verschlüsselten Nachricht kann bekannt sein
- Angreifer **wählt Klartexte und lernt zugehörige Chiffretexte** (chosen plaintext attack), z.B. Angreifer überzeugt Challenger davon, Nachrichten seiner Wahl zu verschlüsseln

### 3.2.4 Sicherheitsspiel (IND-CPA)

- Sicherheit wird in der Kryptographie durch sein **Spiel** zwischen **Angreifer** und **Challenger** definiert

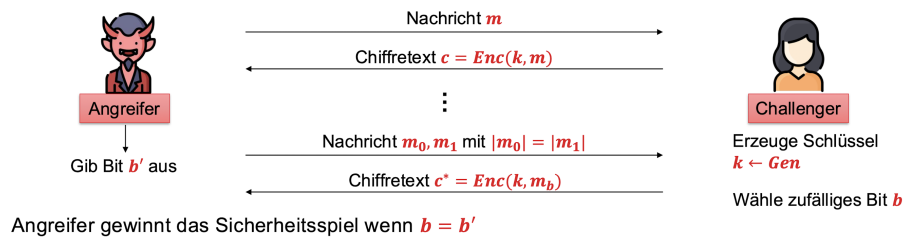


Abbildung 2: Sicherheitsspiel

#### Sicherheit:

Symmetrische Chiffre ist **IND-CPA sicher**, falls alle **effizienten** Angreifer das Sicherheitsspiel maximal mit der Wahrscheinlichkeit  $\approx \frac{1}{2}$  gewinnen können

#### Was bedeutet effizient?

- effizient**: Laufzeit **polynomiell** in der Schlüssellänge
- nicht effizient**: Laufzeit **exponentiell** in der Schlüssellänge

### 3.2.5 Bietet IND-CPA die stärkste Sicherheit?

- Nein, es gibt noch stärkere
- IND-CPA bietet grundlegenden Schutz gegen **passive Angriffe** (nur Zugriff auf Verschlüsselungen)
- Gefahr : **Chosen Ciphertext Angriff**
  - Angreifer hat auch Zugang zu **Entschlüsselung** von (bestimmten) Chiffretexten
  - Angreifer kann diese Informationen nutzen, um **sensitive Informationen** zu erlangen
- Beispiel: **Padding Orakel Angriff**
  - Angreifer kann durch gezielte **Entschlüsselungsanfragen** Informationen über den Klartext gewinnen

### 3.2.6 Stärkere Sicherheit: IND-CCA

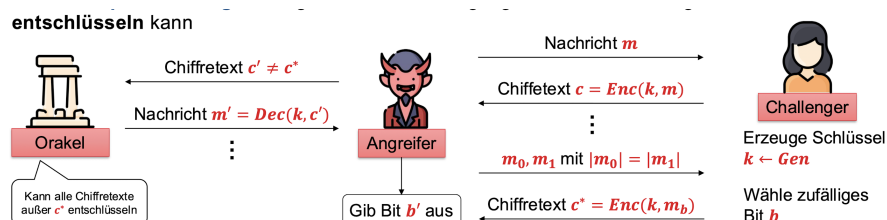


Abbildung 3: Sicherheitsspiel

- Chosen Ciphertext Angriff**
- geht einen Schritt weiter als IND-CPA und schützt auch vor Angreifern, die zusätzlich die Möglichkeit haben, **bestimmte Chiffretexte zu entschlüsseln**

### 3.2.7 Unterschied zwischen IND-CPA und IND-CCA

- **IND-CPA Sicherheit:**
  - schützt vor Angriffen, bei denen Angreifer **Verschlüsselungen** seiner Wahl erzeugen kann
  - reicht nicht aus, wenn Angreifer auch **Zugriff auf Entschlüsselungen** hat
- **IND-CCA Sicherheit:**
  - stärkere Sicherheit
  - schützt selbst dann, wenn Angreifer zusätzlich **Entschlüsselungen anfordern** kann (außer dem zu entschlüsselnden Chiffretext)
- **Fazit**
  - **IND-CCA Sicherheit ist notwendig**, wenn Angreifer auf **Entschlüsselungsoperationen zugreifen kann**

Wie zeigen wir (Un-)sicherheit?

- **Unsicheres Verfahren**
  - konstruiere effizienten Angreifer, der mit Wahrscheinlichkeit signifikant größer als  $\frac{1}{2}$
- **Sicheres Verfahren:**
  - Zeige, dass **alle Angreifer** das Sicherheitsspiel mit Wahrscheinlichkeit  $\approx \frac{1}{2}$  gewinnen
  - Reduktionsbeweis auf Annahmen

### One-Time-Pad Verschlüsselung

Wiederholung: XOR

#### ▪ Bit XOR Operationen

$x \oplus 0 = x$
$x \oplus x = 0$
$x \oplus y = y \oplus x$
$(x \oplus y) \oplus z = x \oplus (y \oplus z)$
$(x \oplus y) \oplus x = y$

#### Erweiterung auf Bitstrings

1	0	0	1	0	1	1	1
$\oplus$							
1	1	1	0	1	0	1	0
$=$							
0	1	1	1	1	1	0	1

Abbildung 4: XOR

### 3.3.1 One-Time-Pad

- auch häufig **Vernam-Chiffre** genannt
- symmetrisches Verschlüsselungsverfahren
- zur Verschlüsselung von **Bitstrings der Länge  $n$**

#### Funktionsweise

- **Gen:** in zufälliger **Schlüssel  $k$**  wird aus der Menge  $\{0,1\}^n$  erzeugt, wobei  $n$  die Länge des Klartextes  $m$  ist
  - Schlüssel muss mindestens so lang wie der Klartext sein und darf nur einmal verwendet werden

**Enc:** Verschlüsselung erfolgt durch eine **XOR-Operation** zwischen dem **Klartext**  $m$  und dem **Schlüssel**  $k$

$\text{Enc}(k, m) = k \oplus m \implies$  Ergebnis ist der **Chiffretext**  $c$

- **Dec:** um den Chiffretext zu **entschlüsseln** wird erneut eine **XOR-Operation** zwischen dem **Chiffretext**  $c$  und dem **Schlüssel**  $k$  durchgeführt

$\text{Dec}(k, c) = k \oplus c$

- da bein XOR die Operation **invertierbar** ist ( $x \oplus x = 0$ ), erhält man den ursprünglichen Klartext zurück.

### 3.3.2 Sicherheit

- beschränktes Sicherheitsspiel: Angreifer erhält **keinen Zugriff auf Chiffretexte**
- Angreifer gewinnt das Sicherheitsspiel, wenn  $b = b'$
- Analyse ergibt: Perfekte Sicherheit -  $c^*$  gibt keine Information über  $m_b$  preis

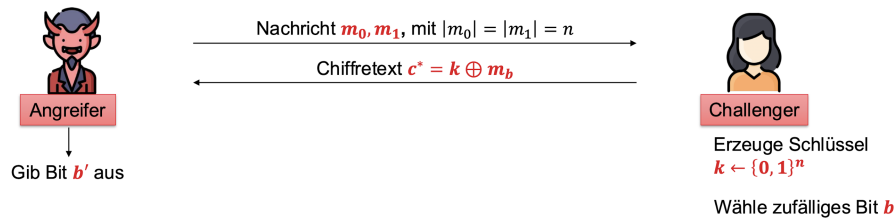


Abbildung 5: beschränktes Sicherheitsspiel

### 3.3.3 Schlüssel nur einmal verwenden

One-Time-Pad ist unsicher bei Wiederverwendung des gleichen Schlüssels

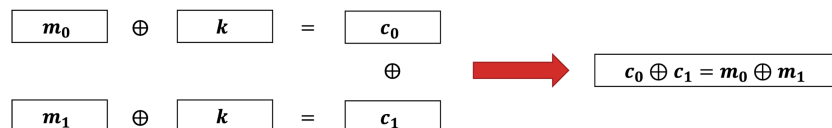


Abbildung 6: One-Time-Pad Schlüssel nur einmal verwenden

- Lernen des XORs kann nützliche Informationen preisgeben
- z.B. welche Bits von  $m_0$  und  $m_1$  gleich sind
- wenn  $m_0$  bekannt ist, dann ist auch  $m_1$  bekannt (und vice versa)
- **Moral:** zur Verschlüsselung jeder Nachricht muss ein neuer zufälliger Schlüssel gewählt werden

### 3.3.4 Venona-Projekt: Risiken von One-Time-Pads

- **Venona-Projekt (1943-1980)**
  - **Ziel:** Entschlüsselung sowjetischer Kommunikation durch USA und Großbritannien
  - Sowjetische Nachrichten wurden mit **One-Time-Pads verschlüsselt**

- Fehler: Schlüssel wurden unter Zeitdruck **mehrfach verwendet**
- **Risiken von One-Time-Pads**
  - **Mehrfachnutzung von Schlüsseln**: Führt dazu, dass Angreifer durch XOR der Chiffretexte Informationen über die Klartexte gewinnen können
  - US-Geheimdienste nutzten diesen Fehler, um sowjetische Nachrichten zu entschlüsseln
- **Lektion**: One-Time-Pad ist nur sicher, wenn jeder **Schlüssel einmalig** verwendet wird

### 3.3.5 Nachteile von One-Time-Pad

1. Schlüssel ist **so lang wie Nachricht**
  - für große Mengen von Daten müssen **lange zufällige Schlüssel** gespeichert und ausgetauscht werden
  - gute Zufälligkeit zu erzeugen, ist sehr aufwendig
2. Schlüssel kann **nur einmal benutzt** werden
  - mehrfache Verwendung kann etwa über Klartexte preisgeben
  - für viele Nachrichten benötigt man **viele Schlüssel**
3. Sicherheit im **beschränkten Angreifermodell**
  - Chiffretext-Only Angriffe

## Übung 1

### Aufgabe 1: Wissensfragen

- (a) Beschreiben Sie die Unterschiede zwischen Safety und Security.
- **Safety (Betriebssicherheit)**
    - Schutz gegen Fehler/Unfälle
    - Fehler meist unbeabsichtigt
    - mögliche Gegenmaßnahmen: Verifikation und Testen
  - **Security (Angriffssicherheit)**
    - Schutz gegen worst-case-Angreifer
    - meist Schadabsicht
    - Verifikation und Testen hilft wenig
- (b) Nennen und beschreiben Sie drei Sicherheitseigenschaften.
- **Vertraulichkeit** (Angreifer kann Nachricht nicht lernen)
  - **Anonymität** (Eigenschaft, dass eine Person oder Gruppe nicht identifiziert werden kann)
  - **Integrität** (Angreifer kann Nachricht nicht ändern, ohne dass Änderung erkannt wird)
  - **Authentizität** (Angreifer kann nicht behaupten, dass eine Nachricht von Alice kam, die diese nicht gesendet hat)
  - **Verfügbarkeit** (Eigenschaft eines Systems Zugriff für autorisierte Benutzer auf Daten und Dienste zu erlauben)
- (c) Beschreiben Sie das Sicherheitsprinzip der fail-safe Standards. Geben Sie ein Beispiel an.
- Dieses Prinzip sorgt dafür, dass ein System bei einem Ausfall oder einer Anomalie in einen Zustand übergeht, der den geringstmöglichen Schaden verursacht.
- Beispiel: mechanisches Zugsignal (aus VL)

- (d) Was sagt das Kerckhoffs'sche Prinzip aus?

Ein Kryptoverfahren soll sicher bleiben, selbst wenn der Angreifer den Kryptoalgorithmus kennt. Die Sicherheit des Kryptoverfahrens beruht auf der Geheimhaltung des Schlüssels.

### Aufgabe 2: IND-CPA vs. IND-CCA

- (a) **Szenario 1:** Sichere Kommunikation im Web: Um Nachrichten zwischen einem Webbrowser und einem Webserver abzusichern, wird ein Verschlüsselungsverfahren eingesetzt. Die Kommunikation soll dabei auch gegen einen Angreifer, der die Kommunikationskanäle kontrolliert, abgesichert werden.

In diesem Szenario hat der Angreifer möglicherweise die Kontrolle über den Kommunikationskanal. Das bedeutet, er könnte **nicht nur Nachrichten abfangen** (passiver Angriff), sondern auch aktiv Nachrichten modifizieren oder selbst verschlüsselte Nachrichten an den Server senden (**aktiver Angriff**). Da die IND-CPA jedoch nur vor passiven Angriffen schützt, empfiehlt sich hier eher die **IND-CCA Sicherheit**.

Musterlösung:

Ein Angreifer, der die Kommunikationskanäle kontrolliert, kann Chiffretexte abfangen, erneut senden oder modifizieren. Der Chiffretext wird dabei vom Empfänger entschlüsselt und der Empfänger führt mögliche Aktionen aus. In diesem Szenario ist IND-CPA Sicherheit nicht ausreichend und wir brauchen IND-CCA Sicherheit des Verschlüsselungsverfahrens. Der Empfänger stellt in diesem Szenario eine Art Entschlüsselungssorakel dar. Auch wenn der Empfänger die entschlüsselte Nachricht nicht an den Angreifer zurück schickt, kann der Angreifer möglicherweise Informationen aus den Aktionen des Empfängers ableiten. (z.B. der Webserver führt eine Transaktion aus). Um einen solchen Angriff zu verhindern, benötigt es eine stärkere Sicherheit als IND-CPA Sicherheit.

- (b) **Szenario 2:** Data at Rest: In einem Unternehmen werden sensible Daten auf einer Festplatte gespeichert. Diese Informationen sollen durch ein Verschlüsselungsverfahren abgesichert werden, sodass selbst bei einem Diebstahl der Festplatte die Daten geheim bleiben.

Musterlösung:

In diesem Szenario ist IND-CPA Sicherheit ausreichend. Sollte ein Angreifer im Besitz einer Festplatte kommen, dann hat er keinen Zugriff auf ein Entschlüsselungssorakel (Unter der Annahme, dass der Schlüssel nicht auf der Festplatte gespeichert ist). Der Angreifer kann lediglich versuchen die Klartexte zu erraten oder andere Online-Angriffe wie einen Bruce-Force-Angriff durchzuführen.

## Thema 3: Symmetrische Kryptographie - Teil

### Data Encryption Standard (DES)

- symmetrischer Verschlüsselungsalgorithmus, der in den 1970er Jahren von IBM im Auftrag des National Institute of Standards and Technology (NIST) entwickelt wurde
- **Blockchiffre**, die Daten in Blöcken von **64 Bit** verschlüsselt und entschlüsselt
- **Blocklänge:**  $n=64$  Bits
- **Schlüssellänge:**  $k=56$  Bits

### Verschlüsselung und Entschlüsselung

- Verschlüsselung erfolgt mit einem Schlüssel  $K \in \{0, 1\}^{56}$  und einer Nachricht  $M \in \{0, 1\}^{64}$
- das Ergebnis ist ein Chiffre  $C \in \{0, 1\}^{64}$
- die Entschlüsselung funktioniert ähnlich, wobei derselbe Schlüssel  $K$  verwendet wird, um das Chiffre  $C$  wieder in die ursprüngliche Nachricht  $M$  zu konvertieren

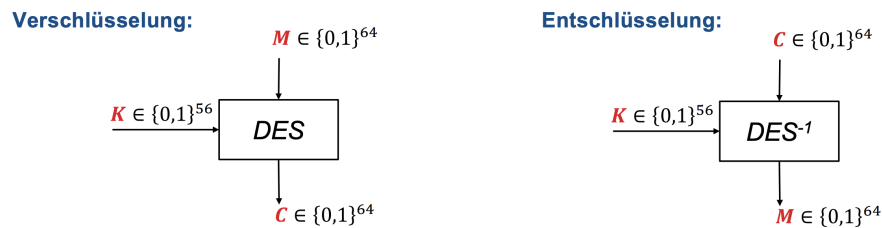


Abbildung 7: Des

### 5.1.1 DES: Angriffe

#### Theoretische Angriffe:

- **Differenzielle Kryptoanalyse**
- **Lineare Kryptoanalyse**
- DES bietet Schutz gegen Differenzielle Kryptoanalyse
- **Hauptschwachpunkt:** kurzer Schlüssel (nur 56 Bits)
  - Brute-Force Angriff ist möglich
  - DES Cracker: bricht DES in wenigen Tagen (in 1988)
- **Erweiterungen:** Mehrfachanwendung von DES zur Verlängerung des Schlüssels (z.B. Triple DES)

### 5.1.2 Triple-DES

- DES-Algorithmus wird dreimal hintereinander auf einen Datenblock angewendet, um die Sicherheit zu erhöhen
- dabei werden **drei 56-Bit-Schlüssel** verwendet, was zu einer **nominalen Schlüssellänge** von **168 Bits** führt

#### Meet-in-the-Middle Angriff

- nutzt den Umstand aus, dass der Angreifer sowohl den Klartext als auch den Chiffretext kennt und durch **Berechnung von Zwischenergebnissen** den Schlüsselraum effizienter durchsuchen kann
- funktioniert folgendermaßen:
  1. Ein Angreifer berechnet für alle möglichen Schlüssel  $K_0$  das Ergebnis  $X = DES(M, K_0)$ , wobei  $M$  die bekannte Nachricht ist ( $2^{56}$  Möglichkeiten)
  2. danach berechnet er für alle möglichen Schlüssel  $K_1$  das Ergebnis  $Y = DES^{-1}(X, K_1)$  ( $2^{56} \cdot 2^{56}$  Möglichkeiten)
  3. schließlich berechnet er für alle möglichen Schlüssel  $K_2$  das Ergebnis  $Z = DES^{-1}(C, K_2)$ , wobei  $C$  der bekannte Chiffretext ist
  4. Der Angreifer vergleicht nun alle Werte von  $Y$  und  $Z$ . Wenn eine Übereinstimmung gefunden wird, hat er potenziell den richtigen Schlüssel gefunden
- dieser Angriff reduziert den Aufwand für das Knacken von Triple-DES auf etwa  $2^{112}$  Operationen anstatt der erwarteten  $2^{168}$  Operationen

#### Advanced Encryption Standard (AES)

- Schlüsselgröße: **128, 192** oder **256 Bit**
- Block-Größe: **128 Bit**
- gilt als unangebrochen: in den USA Zulassung für **höchste Geheimhaltungsstufe**



### 5.2.1 Angriffe in der Praxis

#### 1. Seiten-Kanal-Angriffe

- **passive Angriffe**, bei denen der Angreifer **physikalische Informationen** während der Ausführung einer kryptographischen Operation beobachtet
- Messe die **Zeit**, die für kryptographische Operationen benötigt wird
- Messe den **Stromverbrauch**

#### 2. Fehlerangriffe

- **aktive Angriffe**, bei denen der Angreifer gezielt **Fehler in den Berechnungsprozess** einführt
- können durch **physikalische Manipulationen**, wie Laserstrahlen, elektromagnetische Pulse oder Spannungsschwankungen verursacht werden
- Ziel ist es durch die **Analyse der fehlerhaften Ausgaben** Informationen über den geheimen Schlüssel zu gewinnen

### Blockchiffren im Einsatz

- **nicht IND-CPA sicher**
- Nachrichten beliebiger Länge können nicht direkt verschlüsselt werden
- Merke: Deterministische Verschlüsselung kann nicht IND-CPA sicher sein

### Modes of Operation

- Ziel: Verschlüsselung von Nachrichten mit **beliebiger Länge**

#### 1. Electronic Codebook (ECB) Modus

- nicht IND-CPA sicher

#### 2. Cipher-Block-Chaining (CBC) Modus

#### 3. Counter (CTR) Modus

### 5.3.1 Electronic Code Book (ECB) Modus

- jeder Klartextblock wird **unabhängig** voneinander mit **demselben Schlüssel** verschlüsselt
- bedeutet, dass identische Klartextblöcke immer zu identischen Chiffretextblöcken führen → erhebliche Schwäche

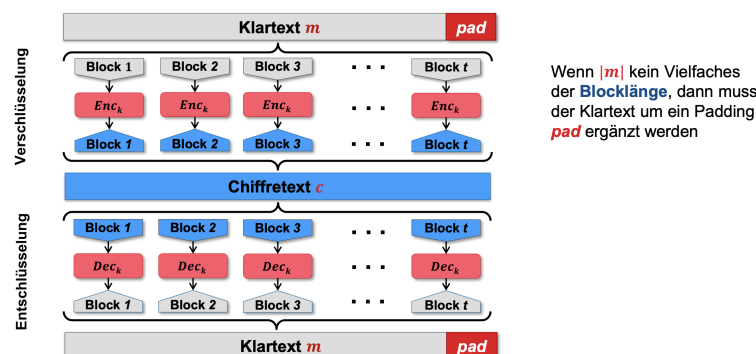


Abbildung 8: ECB-Modus

Funktionsweise:

## 1. Aufteilung des Klartextes in Blöcken

- Klartext  $m$  wird in Blöcke der Größe der Blockchiffre unterteilt
- wenn Länge des Klartextes  $|m|$  **kein Vielfaches** der Blockgröße ist, wird der letzte Block durch **Padding** (Auffüllen) ergänzt, um die Blockgröße zu erreichen

## 2. Verschlüsselung:

- jeder Klartextblock wird **unabhängig** voneinander mit dem **gleichen Schlüssel**  $k$  verschlüsselt

$$c_i = Enc_k(m_i)$$

## 3. Entschlüsselung:

- jeder Chiffretextblock wird unabhängig voneinander mit dem gleichen Schlüssel entschlüsselt

$$m_i = Dec_k(c_i)$$

- die entschlüsselten Blöcke werden zu einem **Klartextblock kombiniert**

## Schwächen des ECB-Modus:

- da jeder Block unabhängig verschlüsselt wird, gibt es **keine Zufälligkeit oder Verknüpfung** zwischen den Blöcken
  - führt dazu, dass ein Angreifer, der einen Teil des Klartextes kennt, leicht **Rückschlüsse** auf den Chiffretext ziehen kann

## 5.3.2 Cipher Block Chaining (CBC) Modus

- jeder Klartextblock wird vor der Verschlüsselung mit dem **vorherigen Klartextblock verknüpft**, was sicherstellt, dass gleiche Klartextblöcke nicht zu gleichen Chiffretextblöcken führen
  - dadurch wird eine Form der **Randomisierung** eingeführt

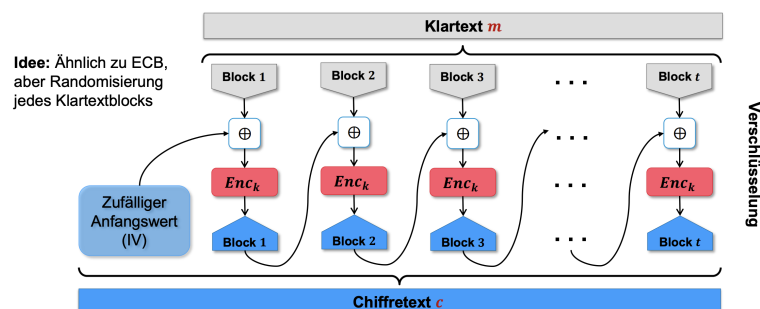


Abbildung 9: CBC-Verschlüsselung

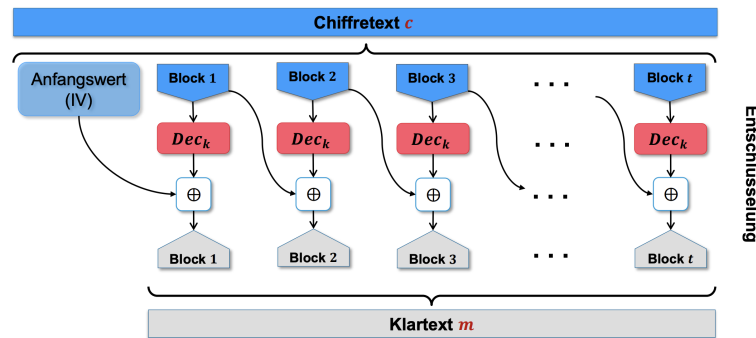


Abbildung 10: CBC-Entschlüsselung

### Funktionsweise des CBC-Modus:

1. **Initialisierungsvektor (IV):**
  - beginnt mit einem zufälligen Initialisierungsvektor, der **genau so lang ist wie ein Block**
2. **Verknüpfung der Blöcke:**
  - jeder Klartextblock wird vor der Verschlüsselung mit dem **Chiffretext des vorherigen Blocks** XOR verknüpft
3. **Entschlüsselung:**
  - jeder Chiffretext wird entschlüsselt und dann mit dem vorherigen Chiffretextblock XOR verknüpft, um den ursprünglichen Klartext wiederherzustellen

### CBC-Modus: Sicherheit

- ist **IND-CPA sicher**, wenn er korrekt verwendet wird

### Padding Angriffe auf CBC

- Chosen-Ciphertext Angriffe können Informationen über Bits der Nachricht liefern
- Bitänderungen in Chiffretext Block 1 führen zu Änderungen in Klartext Block 2
- wenn angreifer lernt ob bei der Entschlüsselung das Padding ungültig war, können **gezielte Modifikationen** in Block 1 **Informationen** über Klartext in Block 2 liefern

### Angriffsszenario:

- Angreifer hat Zugriff auf verschlüsselten Chiffretext, der im CBC-Modus erstellt wurde
- **kennt den Schlüssel nicht**, hat jedoch die Möglichkeit, den Chiffretext zu **modifizieren** und an ein **System zu senden**, das die Entschlüsselung durchführt
- System gibt Rückmeldungen darüber, ob das **Padding nach der Entschlüsselung korrekt** war (z.B. durch Fehlermeldungen)

### Sicherheit gegen Padding-Angriffe vs. IND-CPA/IND-CCA

- Sicherheit gegen Padding Angriffe ist **stärker** als IND-CPA Sicherheit
  - Bsp: CBC-Mode ist IND-CPA sicher, aber nicht gegen Padding Angriffe
- im IND-CCA Spiel hat Angreifer Zugriff auf **Entschlüsselungsrakel**
  - kann alles, was Padding Orakel auch kann (aber noch mehr)
  - ist mächtiger als ein Padding Orakel!
  - IND-CCA bietet **stärkere Sicherheit**

→ Sicherheit gegen Padding Angriffe liegt **zwischen IND-CPA und IND-CCA Sicherheit**

### 5.3.3 Counter Modus (CTR)

- Idee: Nutze Ausgabe der Blockchiffre als One-Time-Pad
- anstatt wie beim One-Time-Pad einen großen zufälligen Schlüssel zu verwenden, wird beim CTR-Modus ein **Zählerwert(Counter)** verwendet
- dieser Zähler wird bei jeder Verschlüsselung **inkrementiert** und zusammen mit einem festen Wert verschlüsselt

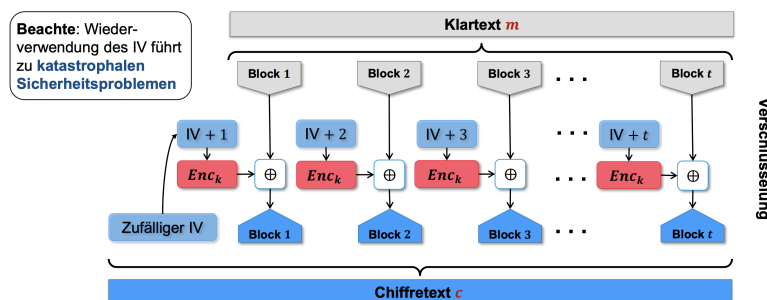
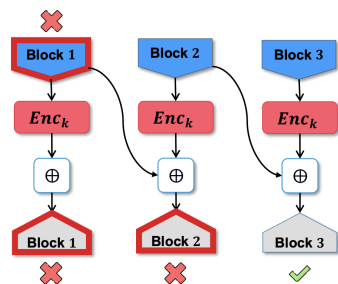


Abbildung 11: Counter Modus

### Fehlerresistenz

#### CBC Modus

- Fehler im Block  $c_1$  betrifft nur Blöcke  $m_1$  und  $m_2$



#### CRT Modus

- Fehler im Block  $c_1$  betrifft nur Block  $m_1$

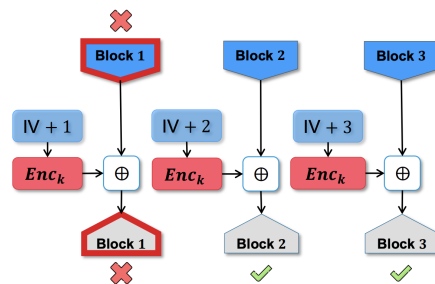


Abbildung 12: Fehlerresistenz

### Vergleich der Modi

#### CBC Modus

##### Fehlerresistenz

- Fehler verbreiten sich nicht über mehr als zwei Blöcke

##### Parallelisierung

- Verschlüsselung: **Nein**
- Entschlüsselung: **Ja**

##### IND-CPA Sicherheit

- **Ja**, aber Achtung bei Wahl des Paddings

##### IND-CCA Sicherheit

- **Nein**

#### CTR Modus

##### Fehlerresistenz

- Fehler wirken sich nur auf den aktuellen Block aus

##### Parallelisierung

- Verschlüsselung: **Ja**
- Entschlüsselung: **Ja**

##### IND-CPA Sicherheit

- **Ja**, aber Achtung auf Zufälligkeit des IV

##### IND-CCA Sicherheit

- **Nein**

Abbildung 13: Vergleich der unterschiedlichen Modi

## Kryptographische Hashfunktionen

### 5.4.1 Hashfunktionen

- Eingabe von Nachrichten mit **beliebiger Länge**
- Ausgabe mit **fixer Länge** (message digest")
- Determinismus: selbe Eingabe erzeugt immer denselben Hashwert

$H : \{0, 1\}^* \rightarrow \{0, 1\}^n$  bildet von großer Definitionsmenge auf kleineren Bildbereich ab

### 5.4.2 Sicherheitsdefinitionen

Es existieren 3 wichtige Sicherheitsdefinitionen für Hashfunktionen

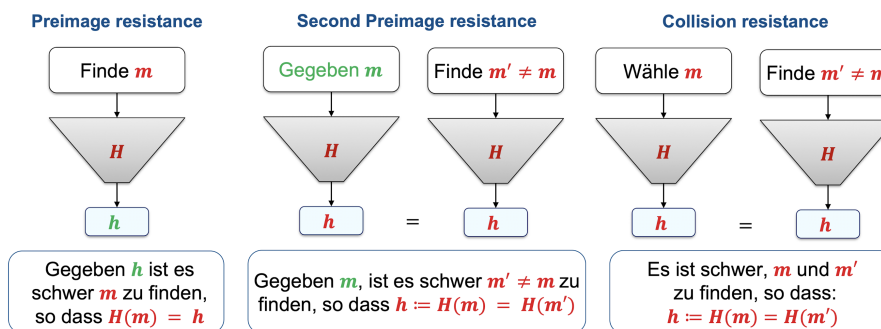


Abbildung 14: Sicherheitsdefinitionen

#### Warum diese Sicherheitsdefinitionen?

##### Preimage Resistance (Urbildresistenz):

- diese Eigenschaft stellt sicher, dass es **extrem schwierig** ist, die ursprüngliche Nachricht (oder Eingabe) aus einem gegebenen Hashwert zu **rekonstruieren**
- **Nutzen:** schützt vor Rekonstruktion aus Hash-Wert, z.B. um Vertraulichkeit von Passwörtern zu sichern

##### Second Preimage Resistance (Zweite Urbildresistenz):

- diese Eigenschaft schützt davor, dass ein Angreifer, der eine Nachricht  $m$  kennt, eine andere Nachricht  $m'$  findet, die denselben Hash-Wert hat wie  $m$
- **Nutzen:** Verhindert Finden verschiedener Daten mit gleichem Hash-Wert, z.B. schwierig zwei gültige Zertifikate zu erzeugen

##### Collision Resistance (Kollisionsresistenz):

- diese Eigenschaft stellt sicher, dass es extrem schwierig ist, zwei beliebige unterschiedliche Nachrichten  $m$  und  $m'$  zu finden, die denselben Hash-Wert haben
- **Nutzen:** Kollisionen sind ausgeschlossen und damit für beliebige Anwendungen anwendbar

## Message Authentication Codes

- kryptografisches Verfahren, das sicherstellt, dass eine Nachricht während der Übertragung **nicht manipuliert wurde und tatsächlich vom angegebenen Absender** stammt
- dient dazu, die **Integrität und Authentizität** einer Nachricht zu garantieren, was für sichere Kommunikation essenziell ist

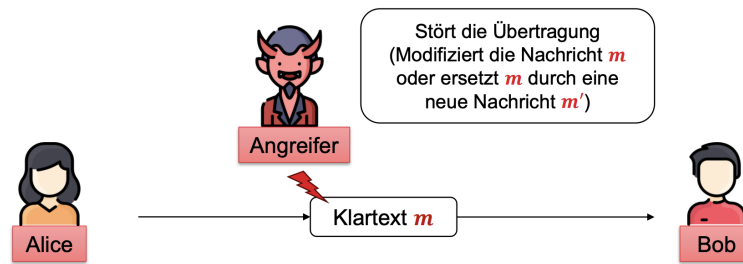


Abbildung 15: Message Authentication

- Im Allgemeinen garantiert Verschlüsselung allein **nicht die Integrität** einer Nachricht
- schützt lediglich die Vertraulichkeit der Daten, jedoch nicht deren Unverfälschtheit

#### Was wird benötigt?

- **Gen (Key-Generation)**
  - generiert einen geheimen Schlüssel  $k$  basierend auf einem Sicherheitsparameter
- **Mac (Tag-Generierung)**
  - berechnet einen Authentifizierungstag  $t$  basierend auf der Nachricht  $m$  und dem Schlüssel  $k$
- **Vrfy (Verifikation)**
  - überprüft die Gültigkeit eines Tags  $t$  für die Nachricht  $m$  und den Schlüssel  $k$

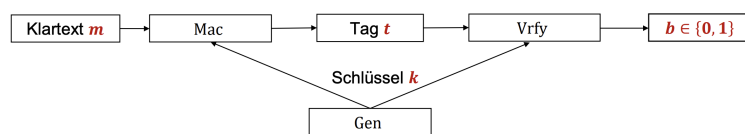


Abbildung 16: Message Authentication Code

**Korrektheit:** Die Verifizierung eines gültigen Tags  $t$  für eine Nachricht  $m$  und dem Schlüssel  $k$  muss korrekt sein. Das bedeutet:

$$\text{Vrfy}_k(m, \text{Mac}_k(m)) = 1 \quad (1)$$

**Effizienz:** Authentifizierung und Verifizierung sind effizient ( $>10$  GB/s)

#### 5.5.1 Sicherheitsdefinition

- Ziel des Angreifers: Was ist ein erfolgreicher Angriff?
  - naive Option: Angreifer lernt den Schlüssel  $k$  nicht
  - in der Kryptographie: Angreifer kann **keinen validen Tag  $t$**  für  $m$  erzeugen

#### Arten von MACs

1. **Informationstheoretisch sichere MACs**  $\rightarrow$  **nicht effizient** für Authentifizierung von vielen Nachrichten
2. **Komplextheoretisch sichere MACs**
  - für Nachrichten **fixer** Länge
  - für Nachrichten mit **beliebiger** Länge, z.B. **CBC-MAC** und **HMAC**

### 5.5.2 CBC-MAC

- ist ein Verfahren zur Berechnung eines **Message Authentication Codes (MAC)** auf Basis einer Blockchiffre
- verwendet den **CBC-Modus**, um eine Nachricht zu authentifizieren
- der resultierende Tag (MAC-Wert) wird aus dem letzten Block des **CBC-Verschlüsselungsprozesses** abgeleitet

#### Funktionsweise von CBC-MAC:

1. Sei  $F_k : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$  eine Blockchiffre mit Schlüssel  $k$ , die Blöcke der Länge  $n$  verarbeitet
2. die Nachricht  $m$  wird in **gleich große Blöcke** aufgeteilt:  $m = (m_1, m_2, \dots, m_t)$ , wobei jeder Block  $m_i \in \{0,1\}^n$  ist
3. der **Initialisierungsvektor (IV)** wird auf  $0^n$  gesetzt (eine Nullfolge der Länge  $n$ )
4. die Blöcke der Nachricht werden **iterativ** mit der Blockchiffre im CBC-Modus verarbeitet
  - der letzte Block wird als Tag ausgegeben
5. Prüfen ob Tag gültig ist:  $\text{Vrfy}_k(m, t) : \tilde{t} = \text{CBC-MAC}_k(m)$ 
  - gib 1 aus, falls  $t = \tilde{t}$ , ansonsten 0

**Wichtig:** CBC-MAC ist nur **sicher für Nachrichten mit fester Länge**

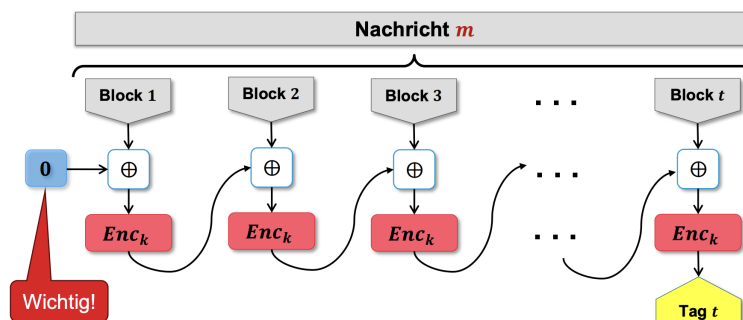


Abbildung 17: CBC-MAC

#### CBC-MAC mit Nachrichten unterschiedlicher Länge

- Nachrichten  $m$  mit  $\text{CBC-MAC}_k(m) = t$  und  $m'$  mit  $\text{CBC-MAC}_k(m) = t$

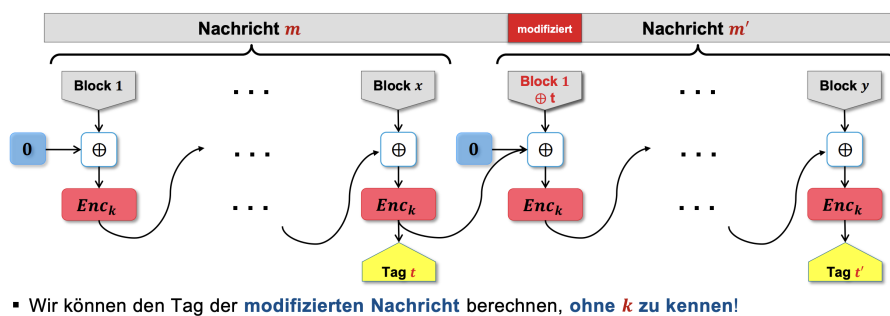


Abbildung 18: CBC-MAC - unterschiedliche Länge