Betriebsarten von Blockchiffren

Blocklänge ist fest und klein. Wie große Mengen an Daten verschlüsseln?

Blockchiffre geeignet verwenden:

- ECB Mode (Electronic Code Book)
- CBC Mode (Cipher Block Chaining)
- CFB Mode (Cipher Feedback)
- OFB Mode (Output Feedback)
- CTR Mode (Counter Mode)

Diese Betriebsarten (ohne CTR) wurden ursprünglich für DES entwickelt, können aber mit jedem Blockchiffre verwendet werden.

Sind standardisiert.

ECB – Electronic Code Book Mode

Einfachste Herangehensweise.

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.
Letzten Block durch (zufällige) Bits ergänzen, falls nötig (Padding).

Verschlüsselung durch $c = c_1 c_2 \cdots c_t$ mit $c_i = \mathcal{E}(k, m_i)$.
Entschlüsseln durch $m_i = \mathcal{D}(k, c_i)$.

ECB – Electronic Code Book Mode

Eigenschaften:

- $m_i = m_j$ dann $c_i = c_j$, also Regelmäßigkeiten und Wiederholungen übertragen sich.
- Unabhängige c_i, Übertragungsfehler auf Block beschränkt.

Beispiel: Bei Bildern bleiben häufig Konturen erkennbar!

Probleme:

- Chiffretext zu Klartext am Anfang/Ende von Nachrichten extrahierbar.
- Block replay: Mischen/Einfügen von bekanntem Chiffretext möglich.

Anwendung: Besser nicht (u.U. Verschlüsselung von Schlüsseln).

Padding

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.
Padding = Letzten Block durch (zufällige) Bits ergänzen.

Vom Standpunkt der Kryptographie ist egal, wie man ergänzt (da jeder Klartext sicher verschlüsselt werden soll).

Warum nicht nur Nullen anhängen?

Ansätze:

- Eine Eins und soviele Nullen anhängen, wie nötig.
- Zufällige Bytes und Anzahl zu entfernender Bytes hinten anhängen.

CBC – Cipher Block Chaining Mode

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.
Letzten Block durch (zufällige) Bits ergänzen, falls nötig.

Verschlüsselung durch $c = (c_0)c_1 c_2 \cdots c_t$ mit $c_0 = IV$ und
$c_i = \mathcal{E}(k, m_i \oplus c_{i-1})$ für $i \geq 1$.

Entschlüsseln durch
$m_i = \mathcal{D}(k, c_i) \oplus c_{i-1}$ für $i \geq 1$.

IV ist zufällig gewählt, oder wird aus m erzeugt (so daß es nur (!) für m vorkommt, z.B. die Verschlüsselung einer eindeutigen Nachrichtennummer).
Braucht nicht geheim gehalten zu werden.

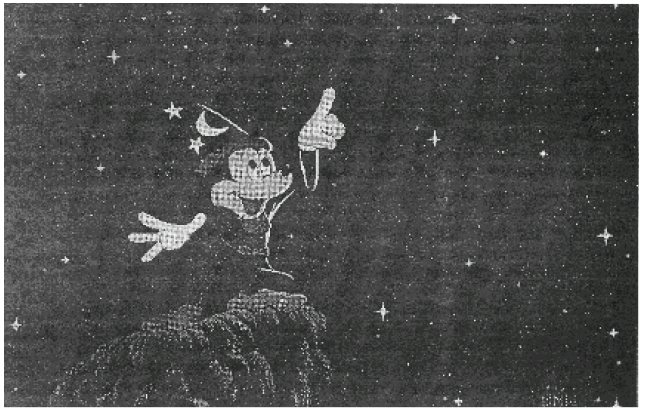
CBC – Cipher Block Chaining Mode

Eigenschaften:

- Kontextabhängig: c_i hängt von c_j mit $j < i$ ab.
- Regelmäßigkeiten und Wiederholungen werden (durch unterschiedliche IV) verwischt.
- Fehler in c_i betrifft nur m_i und lokal m_{i+1}.
- Block replay nicht möglich.

Anwendung: Ist der Standardmodus. Verschlüsseln langer Nachrichten.

Bild unverschlüsselt



(ausgeliehen von N. Smart, F. Vercauteren)

Bild verschlüsselt im ECB Mode

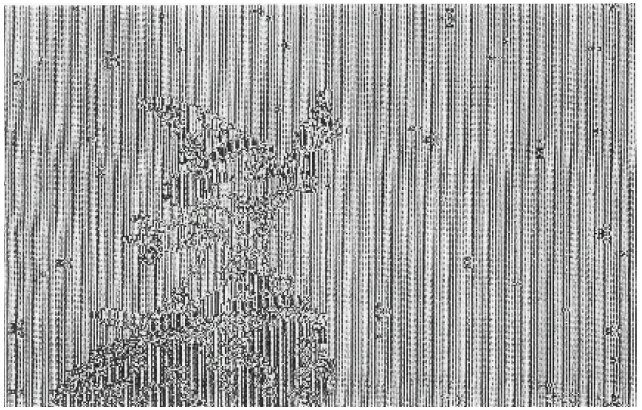
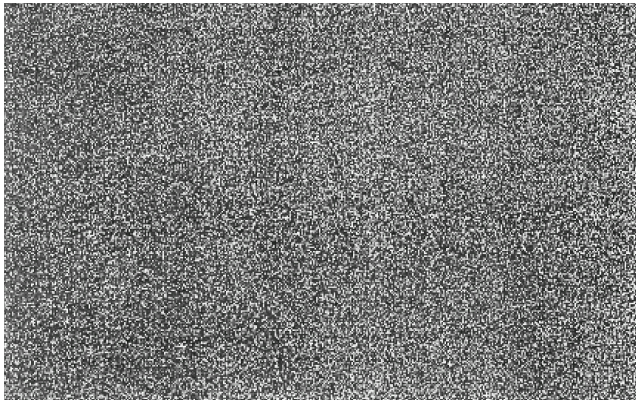


Bild verschlüsselt im CBC Mode



OFB – Output Feedback Mode

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.

Verschlüsselung durch $c = c_1 c_2 \cdots c_t$ mit
$k_0 = IV$ und $k_i = \mathcal{E}(k, k_{i-1})$ für $1 \leq i \leq t$,
$c_i = m_i \oplus k_i$ für $1 \leq i \leq t$.

Entschlüsseln durch
$k_0 = IV$ und $k_i = \mathcal{E}(k, k_{i-1})$ für $1 \leq i \leq t$,
$m_i = c_i \oplus k_i$ für $1 \leq i \leq t$.

IV wird wie bei CBC benutzt. Beim Ver- und Entschlüsseln von m_i bzw. c_i kann man auch nur einen Teil von k_i verwenden, wenn die Blocklänge kleiner als die Schlüssellänge ist.

OFB – Output Feedback Mode

Eigenschaften:
- Entschlüsseln = Verschlüsseln, nur $\mathcal{E}$ benutzt.
- Kein Padding notwendig.
- Fehler in c_i bleiben lokal.
- c_i hängt nicht von c_j für $j < i$ ab.
- Vergleichbar zum One-Time Pad.

Probleme:
- Gefahr: Gleiches IV, gleiche k_i (!).
- k_i periodisch.

Anwendung:
- Satellitenkommunikation (wegen der Fehler).
- Filesysteme/Datenbanken wegen wahlfreiem Zugriff.

CFB – Cipher Feedback Mode

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.

Verschlüsselung durch $c = c_1 c_2 \cdots c_t$ mit
$c_0 = IV$ und $k_i = \mathcal{E}(k, c_{i-1})$ für $1 \leq i \leq t$,
$c_i = m_i \oplus k_i$ für $1 \leq i \leq t$.

Entschlüsseln durch
$c_0 = IV$ und $k_i = \mathcal{E}(k, c_{i-1})$ für $1 \leq i \leq t$,
$m_i = c_i \oplus k_i$ für $1 \leq i \leq t$.

IV wird wie bei CBC benutzt. Beim Ver- und Entschlüsseln von m_i bzw. c_i kann man auch in geeigneter Weise nur einen Teil von k_i verwenden.

CFB – Cipher Feedback Mode

Eigenschaften:

- Entschlüsseln = Verschlüsseln, nur $\mathcal{E}$ benutzt.
- Kein Padding notwendig.
- c_i hängt von c_j für $j < i$ ab.
- Fehler in c_i erzeugt Fehler lokal in m_i und in m_{i+1}.

Anwendung:

- Stückweise anfallende kleinere Datenmengen (Ströme).

CTR – Counter Mode

Klartext m in Blöcke der passenden Größe aufteilen $m = m_1 m_2 \cdots m_t$.

Verschlüsselung durch $c = c_1 c_2 \cdots c_t$ mit

$k_i = \mathcal{E}(k, \text{Nonce} \oplus i)$ für $1 \leq i \leq t$,

$c_i = m_i \oplus k_i$ für $1 \leq i \leq t$.

Entschlüsseln durch

$k_i = \mathcal{E}(k, \text{Nonce} \oplus i)$ für $1 \leq i \leq t$,

$m_i = c_i \oplus k_i$ für $1 \leq i \leq t$.

Nonce ist eine „Number to be used once“ (!) und wird ähnlich wie IV verwendet.

CTR – Counter Mode

CTR hat nach Ferguson-Schneier im wesentlichen nur Vorteile gegenüber den anderen Modes.

Eigenschaften:

- Nur $\mathcal{E}$ erforderlich.
- Kein Padding notwendig.
- Parallelisierbar.
- Wahlfreier Zugriff.
- Fehler in c_i bleiben lokal.
- Vergleichbar zum One-Time Pad.

Robustheit: Die Nonce muß pro verschlüsselter Nachricht m eindeutig sein. Diesbezüglich ist CBC robuster als CTR.

Ist standardisiert für AES.

Bemerkungen

Bit twiddling Angriffe: Ändert man Bits in c_i, so ändern sich auch die entsprechenden Bits in m_i. Bei CBC und CFB wird darüberhinaus m_{i+1} zum Großteil gestört.

In den Modes daher nach Möglichkeit chiffrierte Prüfsummen (MAC) verwenden.

Informationleakage:

- CBC: $c_i = c_j \Leftrightarrow m_i \oplus c_{i-1} = m_j \oplus c_{j-1} \Leftrightarrow m_i \oplus m_j = c_{i-1} \oplus c_{j-1}$.
- CTR: nur $m_i \oplus m_j \neq c_i \oplus c_j$ zu erfahren.

Birthday Angriff bei kleiner Blocklänge: Ist die Blocklänge 2^n, so kann man nach ca. $2^{n/2}$ verschlüsselten Blöcken erwarten, daß zwei Chiffretexte gleich sind. Daher Anzahl mit gleichem Schlüssel verschlüsselter Blöcke auf z.B. $2^{n/4}$ beschränken.

Mehrfachverschlüsselung

Doppelte Verschlüsselung:

$c = \mathcal{E}(k_2, \mathcal{E}(k_1, m))$.

$m = \mathcal{D}(k_1, \mathcal{D}(k_2, c))$.

Es gelte $\#K = 2^n$.

Sicherheit bezüglich exhaustive search 2^{2n} statt vorher 2^n (?).

Problem, wenn $\mathcal{E}(k_2, \mathcal{E}(k_1, \cdot)) = \mathcal{E}(k_3, \cdot)$ für ein k_3.

(Die Permutationen $DES(k, \cdot)$ erzeugen eine Untergruppe von $S(\{0,1\}^{56})$ der Ordnung $\geq 10^{2499}$).

Unter Known-Plaintext Angriff Sicherheit nur 2^{n+1} statt 2^{2n}:

Meet-in-the-middle Angriff. Daher wird doppelte Verschlüsselung im allgemeinen nicht verwendet.

Meet-in-the-middle Angriff

Es gelte $\#K = \#M = \#C$.

Gegeben/bekannt: c_1, c_2, m_1, m_2 mit $c_1 = \mathcal{E}(k_2, \mathcal{E}(k_1, m_1))$, $c_2 = \mathcal{E}(k_2, \mathcal{E}(k_1, m_2))$.

1. Berechne und speichere $\mathcal{E}(k_1', m_1)$ für alle $k_1' \in K$.
2. Berechne $\mathcal{D}(k_2', c_1)$ für alle $k_2' \in K$.
3. Für $\mathcal{D}(k_2', c_1) = \mathcal{E}(k_1', m_1)$ teste $c_2 = \mathcal{E}(k_2', \mathcal{E}(k_1', m_2))$.
4. Liefert eine kleine Menge Z von (k_1', k_2') mit $(k_1, k_2) \in Z$.

(Größe von Z idealerweise erwartungsgemäß gleich 1.)

Benötigt Speicher der Größe $O(\#K)$, Laufzeit ebenfalls $O(\#K)$.

Abkürzung: MITM

Mehrfachverschlüsselung

Dreifache Verschlüsselung (EDE):

$c = \mathcal{E}(k_3, \mathcal{D}(k_2, \mathcal{E}(k_1, m)))$.

$m = \mathcal{D}(k_1, \mathcal{E}(k_2, \mathcal{D}(k_3, c)))$.

$k_1 = k_3$, k_2 unabhängig und zufällig:

CPA-MITM Angriff in Laufzeit $O(\#K)$ und Speicher $O(\#K)$.

Vermutlich trotzdem nützlich, wenn Anzahl Verschlüsselungen beschränkt.

k_1, k_2, k_3 unabhängig und zufällig:

Known Plaintext MITM Angriff in Laufzeit $O(\#K^2)$ und Speicher $O(\#K)$.

Modus erlaubt Rückwärtskompatibilität, wenn $k_1 = k_2$.

Whitening

$c = k_3 \oplus \mathcal{E}(k_2, m \oplus k_1)$.

Sicherheit maximal $O(\#K^2)$ unter Known-Plaintext Angriff:

1. Seien $c_i = k_3 \oplus \mathcal{E}(k_2, m_i \oplus k_1)$, $c_j = k_3 \oplus \mathcal{E}(k_2, m_j \oplus k_1)$.
2. Dann gilt $c_i \oplus c_j = \mathcal{E}(k_2, m_i \oplus k_1) \oplus \mathcal{E}(k_2, m_j \oplus k_1)$.
3. Definiere $dc = c_i \oplus c_j$, $dm = m_i \oplus m_j$, $m = m_i \oplus k_1$.
4. Betrachte Gleichung $dc = \mathcal{E}(x, y) \oplus \mathcal{E}(x, y \oplus dm)$.
5. Nach $\#K^2$ Tests ca. $\#K$ Lösungen x, y gefunden, darunter $x = k_2$ und $y = m_i \oplus k_1$ oder $y = m_j \oplus k_1$.
6. Dann $k_1 = y \oplus m_i$ und $k_3 = c_i \oplus \mathcal{E}(x, y)$ oder $k_1 = y \oplus m_j$ und $k_3 = c_j \oplus \mathcal{E}(x, y)$. Möglichkeiten an weiteren c_i, m_i testen.

Schlüssellänge auf $r \log_2(\#K)$ vergrößern mit Sicherheitszuwachs auf $\#K^r$ im allgemeinen schwierig (sonst mit $n = 1$ anwenden, liefert ...)