Strukturelle Angriffe - Differentielle Kryptanalyse

Differentielle Kryptanalyse:

- Eingeführt von Biham und Shamir (1991).
- Idee: Spezifische Eingabedifferenzen Δ_x führen zu spezifischen Ausgabedifferenzen Δ_y.

Definition Differential

Sei F eine Blockchiffre mit Blocklänge n. Ein *Differential* $(\Delta_x, \Delta_y) \in \{0,1\}^n \times \{0,1\}^n$ hält mit Ws p, falls für zufällige k und $x_1, x_2 \in_R \{0,1\}^n$ mit $x_1 \oplus x_2 = \Delta_x$ gilt $\mathrm{Ws}[F_k(x_1) \oplus F_k(x_2) = \Delta_y] = p$.

- Für Pseudozufallsfunktionen sollte kein Differential mit Wahrscheinlichkeit signifikant größer als 2^{-n} halten.
- Für große Differentiale ist F offenbar keine Pseudozufallsfunktion.
- Mehrere große Differentiale führen oft zu einem CPA-Angriff auf k.
- Finden von (Δ_x, Δ_y) geschieht entweder durch Brute-Force Suche oder durch cleveres Ausnutzen der Blockchiffrenstruktur.

Differentielle Kryptanalyse von DES

Differentielle Kryptanalyse von DES:

- CPA-Angriff mit 2^{36} von insgesamt 2^{47} gewählten Klartexten.
- Benötigt Laufzeit 2^{37} und vernachlässigbaren Speicherbedarf.
- Theoretisch deutlich besser als Brute-Force, die Anzahl der benötigten gewählten Klartexte ist allerdings sehr groß.
- DES wurde als resistent gegen differentielle Krypanalyse designt.
- Die DES-Designer veröffentlichten allerdings den Angriff nicht.

Strukturelle Angriffe - Lineare Kryptanalyse

Lineare Kryptanalyse:

- Entwickelt von Matsui (1993). Methode verwendet lineare Beziehungen der Ein- und Ausgabebits.

Definition Bias

Sei F eine Blockchiffre mit Blocklänge n. Seien $I = \{i_1, \ldots, i_\ell\}$, $J = \{j_1, \ldots, j_\ell\} \subseteq \{1, \ldots, n\}$. Die Indexmengen I, J besitzen *Bias* p, falls für zufällige k und $x \in \{0,1\}^n$ mit $y = F_k(x)$ gilt:

$$\mathrm{Ws}[x_{i_1} \oplus \ldots \oplus x_{i_\ell} \oplus y_{j_1} \oplus \ldots \oplus y_{j_\ell} = 0] = p.$$

- Für echte Zufallsfunktionen sollte der Bias nie signifikant $> \frac{1}{2}$ sein.
- Matsui: Ein genügend großer Bias einer Blockchiffre führt dazu, dass der geheime Schlüssel rekonstruiert werden kann.
- Methode ist KPA, d.h. benötigt keine gewählten Klartexte.
- Liefert KPA-Angriff auf DES mit 2^{43} bekannten Paaren (x_i, y_i).
- Laufzeit ist 2^{43}, Speicherplatz ist vernachlässigbar.

Doppelte Verschlüsselung bringt wenig

Szenario: doppelte Verschlüsselung

- Sei F eine Blockchiffre mit Schlüssellänge n wie z.B. DES.
- Dann besitzt $F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x))$ Schlüssellänge $2n$.
- Leider liefert F' kein Sicherheitsniveau von 2^{2n}.

Algorithmus Meet-in-the-Middle Angriff auf doppelte Verschl.

EINGABE: (x_1, y_1), (x_2, y_2)

1. Für alle $k_1 \in \{0,1\}^n$, berechne $z := F_{k_1}(x_1)$. Speichere (z, k_1) in einer nach der ersten Komponente sortierten Liste L_1.
2. Für alle $k_2 \in \{0,1\}^n$, berechne $z := F^{-1}_{k_2}(y_1)$. Speichere (z, k_2) in einer nach der ersten Komponente sortierten Liste L_2.
3. Für alle z mit $(z, k_1) \in L_1$ und $(z, k_2) \in L_2$, speichere (k_1, k_2) in S.
4. Für alle $(k_1, k_2) \in S$: Verifiziere Korrektheit mittels (x_2, y_2).

AUSGABE: $k = (k_1, k_2)$

Doppelte Verschlüsselung bringt wenig

Korrektheit:

- Für korrektes (k_1, k_2) gilt $F_{k_2}(F_{k_1}(x)) = y$, d.h. $F_{k_1}(x) = F_{k_2}^{-1}(y)$.
- Ein falsches (k_1, k_2) erfüllt diese Identität mit Ws etwa 2^{-n}.
- D.h. wir erwarten $2^{2n} \cdot 2^{-n} = 2^n$ Elemente in der Menge S.
- Verifizieren mit (x_2, y_2) liefert den korrekten Schlüssel.

Laufzeit: Wir zählen Operationen auf einzelnen Schlüsseln mit Zeit/Platz $\mathcal{O}(1)$.

- Schritt 1 und 2: jeweils Zeit $\mathcal{O}(n \cdot 2^n)$ und Platz $\mathcal{O}(2^n)$.
- Schritt 3: Laufzeit und Platz $\mathcal{O}(2^n)$.
- Schritt 4 lässt sich in Laufzeit $\mathcal{O}(2^n)$ realisieren.
- D.h. wir erhalten Gesamtlaufzeit $\mathcal{O}(n \cdot 2^n)$ und Platz $\mathcal{O}(2^n)$.
- Damit erhöht sich die Laufzeit gegenüber einem Brute-Force Angriff bei einfacher Verschlüsselung nicht wesentlich.

Dreifache Verschlüsselung

Szenario: dreifache Verschlüsselung

1. Variante 1: $F'_{k_1,k_2,k_3}(x) := F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$
2. Variante 2: $F'_{k_1,k_2}(x) := F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$

Grund des Alternierens von F, F^{-1}, F: Für die Wahl von $k_1 = k_2 = k_3$ erhalten wir eine einfache Anwendung von $F_{k_1}(x)$.

Sicherheit der 1. Variante:

- Meet-in-the-Middle Angriff wie zuvor in Zeit $\mathcal{O}(n \cdot 2^{2n})$.
- Benötigt Speicherplatz $\mathcal{O}(2^{2n})$.

Sicherheit der 2. Variante:

- Bekannter CPA-Angriff mit $\mathcal{O}(2^n)$ gewählten Paaren.
- Zeitkomplexität beträgt ebenfalls $\mathcal{O}(2^n)$.

Triple-DES:

- Beide Varianten von Triple-DES finden in der Praxis Verwendung.
- Löste 1999 DES als Standard ab. Trotz Standardisierung von AES im Jahr 2002 ist Triple-DES auch heute noch weitverbreitet.

AES - Advanced Encryption Standard

NIST Wettbewerb: (National Institute of Standard and Technology)

- Jan 1997: Aufruf zum Konstruktions-Wettbewerb einer Blockchiffre
- Ursprünglich 15 Kandidaten eingereicht.
- Aug 1999: Auswahl von fünf AES-Finalisten

  MARS, RC6, Rijndael, Serpent und Twofish.
- Okt 2000: Auswahl von Rijndael der Autoren Rijmen und Daemen.

Struktur von AES (Rijndael):

- AES ist ein SPN und besitzt Blocklänge 128.
- Schlüssel mit 128, 192 und 256 Bit können verwendet werden.
- Eingaben $x \in \{0,1\}^{128}$ werden rundenweise in einer 4×4-Byte Matrix, der sogenannten Zustandsmatrix, modifiziert.
- Anzahl Runden: 10 für 128-Bit k, 12 für 192-Bit und 14 für 256-Bit.

Die vier Rundenoperationen von AES

Operation 1: AddRoundKey

- Leite aus k einen Rundenschlüssel $k_i \in \{0,1\}^{128}$ ab.
- XOR der Zustandsmatrix mit k_i.

Operation 2: SubByte

- Interpretiere jedes Byte der Zustandsmatrix als Element $x \in \mathbb{F}_8$.
- Ersetze x durch x^{-1} in $\mathbb{F}_8$ und 0^8 durch 0^8.
- Wende eine affine Transformation auf die Zustandsbytes an.
- Man beachte: Dieselbe S-Box wird für alle Bytes verwendet.

Operation 3: ShiftRow

- Verschiebe die 4 Zeilen der 4×4-Zustandsmatrix zyklisch.
- Lasse die 1. Zeile unverändert.
- Verschiebe die 2. Zeile um eine Position nach links, die 3. Zeile um 2 nach links und die 3. Zeile um 3 Positionen nach links.

Die vier Rundenoperationen von AES

Operation 4: MixColumn

- Sei $a_{0,j}, a_{1,j}, a_{2,j}, a_{3,j}$ eine Spalte der Zustandsmatrix.
- Betrachte die Spalte als Element aus $\mathbb{F}_8/(x^4+1)$, d.h.

$$a_{0,j} + a_{1,j}x + a_{2,j}x^2 + a_{3,j}x^3 \text{ mit } a_{i,j} \in \mathbb{F}_8.$$

- Multipliziere mit $c(x) = 2 + x + x^2 + 3x^3 \in \mathbb{F}_8/(x^4+1)$.
- Kodieren $a \in \mathbb{F}_8 = F_2[y]/\pi$, π irreduzibel mit Grad 8, wie folgt: Sei z.B. $a = y^5 + y + 1$. Wir schreiben $a = (00100011) = 36$.
- MixColumn entspricht Multiplikation mit einer Matrix $C \in \mathbb{F}_8^{4\times 4}$.
- D.h. eine Spalte x wird mittels $x \to Cx$ linear abgebildet.
- Menge aller (x, Cx) definiert einen linearen Code mit Distanz 5.
- D.h. unterscheiden sich zwei Spalten x, x' in nur einer Position, so unterscheiden sie sich nach MixColumn in allen 4 Positionen.
- Diese Eigenschaft führt zu einer schnellen Diffusion bei AES.