Cheatsheet: Computersystemsicherheit

Version vom 12. September 2016

CBC-Mode

Nachricht in n Teile gliedern.
Schlüssel ⇒ Permutationsmatrix mit Einheitsvektoren

$$\pi = \begin{pmatrix} 1 & \cdots & n \\ a & \cdots & z \end{pmatrix} \Rightarrow P_\pi = \begin{pmatrix} e_a \\ \vdots \\ e_z \end{pmatrix}$$

Verschlüsselung:
$c_0 = IV$
$c_i = E(\pi, c_{i-1} \oplus m_i) = P_\pi \cdot (c_{i-1} \oplus m_i)$
$\Rightarrow \quad c_1, \ldots, c_n$
Entschlüsselung:
$m_i = D(\pi^{-1}, c_i) \oplus c_{i-1} = (P_{\pi^{-1}} \cdot c_i) \oplus c_{i-1}$
mit $\pi^{-1} = \pi'$ (transponiert)

CFB-Mode

Verschlüsselung:
$c_i = E(\pi, c_{i-1}) \oplus m_i$
Entschlüsselung:
$m_i = E(\pi, c_{i-1}) \oplus c_i$

CTR-Mode

Verschlüsselung:
$c_i = E(\pi, IV + (i-1)) \oplus m_i$
(binäre Addition, Überträge verwerfen)
Entschlüsselung:
$m_i = E(\pi, IV + (i-1)) \oplus c_i$

Hashfunktion

Nicht injektive Abbildung, die Urbildbildbereich auf erheblich kleineren Bildbereich abbildet.
Speicherung von Passwörtern, Dateivalidierung

Message Authentication Code (MAC)

Hashfunktion mit geheimen Schlüssel zur Integritätsprüfung von Nachrichten. Ermöglicht kein non-repudiation, daher nicht als digitale Unterschrift geeignet.

Kollisionsresistenz

Es ist schwierig zwei Werte x und y mit $H(x) = H(y)$ zu bestimmen

Schwache Kollisionsresistenz

Es ist schwierig zu geg. Wert x ein x' mit $H(x) = H(x')$ zu bestimmen

Shamir Secret-Sharing

t von n Stakeholdern sind nötig, um Geheimnis $k = f(0)$ zu entschlüsseln. Außerdem gegeben: Primzahl $p > n, k$ und vom Dealer gewähltes Polynom f(x) vom Grad $t - 1$
Schlüssel $s_i = f(i) \mod p$
Secret Recovery:
$k = f(0) = \sum s_i \cdot l_i(0) \quad l_i(0) := \left[\prod_{i=1, j \neq i} \frac{j}{j-i}\right] \mod p$ Beim Berechnen von $l_i(0)$ die Nenner zu $(a)^{-1}$ zusammenfassen und als inverses Element berechnen.

Erweiterter Euklidischer Algorithmus (EEA)

$ggT(a,b)$ und $x \cdot a + y \cdot b = d$

q	r	x	y	a	b	x_2	x_1	y_2	y_1
X	X	X	X	a	b	1	0	0	1
a/b	$a \mod b$	$x_2 - qx_1$	$y_2 - qy_1$	b	r	x_1	x	y_1	y
				$\cdots$					
?	0	?	?	**=d**	**0**	**=x**	?	**=y**	?

Inverses Element $(a)^{-1}$ berechnen

Es gilt: $\left[a \cdot (a)^{-1}\right] \mod k = 1$
$ggT(a,k)$ mit EEA durchführen.

$$(a)^{-1} = \begin{cases} (x+k) \mod k & \text{wenn } x < 0 \\ x \mod k & \text{sonst} \end{cases}$$

iptables

Chains: Pakete von...

INPUT	Außen an System mit Firewall
FORWARD	Außen an System innerhalb des geschützten Bereichs
OUTPUT	Innen nach Außen

Targets:

ACCEPT	Akzeptieren und Weiterleiten
DROP	Verwerfen ohne Info an Absender
REJECT	Verwerfen mit Info

Parameter:

Param	Argumente	Erklärung
-P	Chain Target	Policy für Chain
-A	Chain	Append Regel
-p	Protokoll	z.B. tcp, icmp
-j	Target	Gibt Target an
-F	Chain	Flush, löscht alle Regeln für Chain außer Standardregeln mit -P
-i/-o	Interface	in-interface bzw. out-interface
-dport/sport	port	Destination/Source port

Reliability

Wahrscheinlichkeit, dass ein System über gewissen Zeitraum korrekt funktioniert.
Reihenschaltung: $R_{ges}(t) = \prod_{i=1}^{n} R_i$
Falls gleiches Modell $R_i(t) = e^{-\lambda_i t}$ gilt: $R_{ges}(t) = e^{-\lambda t}$ mit $\lambda = \sum_{i=1}^{n} \lambda_i$
Parallelschaltung: $R_{ges}(t) = 1 - \prod_{i=1}^{n}(1 - R_i(t))$

Availability

Verfügbarkeit eines Systems in %:
A=Uptime/(Downtime+Uptime) = MTTF/(MTTF+MTTR)
MTTF (Mean Time to Failure): $MTTF = \int_0^\infty R(t)dt \overset{\exp}{=} \frac{1}{\lambda}$
MTTR (Mean Time to Recovery):
Wahrscheinlichkeit $F(t)$, dass System bis t fehlerhaft wird.
Exp. Modell: $F(t) = 1 - e^{-\lambda t}$
Reliability Funktion: $R(t) = 1 - F(t)$ exp. Modell: $R(t) = e^{-\lambda t}$

Sicherheitsziele

Safety		Betriebssicherheit, Ablauf- und Ausfallsicherheit
Security		Angriffssicherheit
	Authenticity	Authentizität, Echtheit, Glaubwürdigkeit
	Integrity	Integrität der Daten
	Confidentiality	Vetraulichkeit, keine unautorisierte Datengewinnung
	Availability	Verfügbarkeit, keine Funktionsbeeinträchtigungen
	Non repudiation	Verbindlichkeit und Zuordenbarkeit

Kasiski-Test

1. Doppelt vorkommende N-Gramme und deren Abstand bestimmen
2. Primfaktorzerlegung
3. Gemeinsame Faktoren entsprechen Schlüssellänge

OTP – Beweis perfekte Sicherheit

zu zeigen: $Pr[Enc(k, m_1) = c | k \xleftarrow{\$} \mathcal{K}] = Pr[Enc(k, m_2) = c | k \xleftarrow{\$} \mathcal{K}]$

$$Pr[Enc(k,m) = c | k \xleftarrow{\$} \mathcal{K}] = Pr[m \oplus k = c | k \xleftarrow{\$} \mathcal{K}] \tag{1}$$
$$= Pr[k = c \oplus m | k \xleftarrow{\$} \mathcal{K}] \tag{2}$$
$$= Pr[k = k^* | k \xleftarrow{\$} \mathcal{K}] = \frac{1}{2^l} \tag{3}$$

Cheatsheet: Computersystemsicherheit

Version vom 12. September 2016

Satz von Euler

$n > 1 \Rightarrow a^{\phi(n)} \equiv 1(\mod n) \forall a \in \mathbb{Z}_n^*$

Spezialfall: kleiner Satz von Fermat

Primzahl $p > 1$ so gilt für jede Zahl x mit $ggT(x,p) = 1$:

$$x^{p-1} \equiv 1(\mod n)$$

In jeder endl. Gruppe M gilt: $x^{ord(M)} = 1 \forall x \in M$

Korrektheit von ElGamal

Bekannt: $E[(g,h),m] = (c_1,c_2) := (g^k,mh^k)$
Zu zeigen: $D(E(m)) = m \quad \forall m \in \mathbb{Z}_p^*$
Beweis:

$$\begin{aligned}
D(E(m)) &= D(g^k,mh^k) && (4)\\
&\equiv mh^k \cdot (g^k)^{-a} \mod p && (5)\\
&\equiv mh^k \cdot g^{k(p-1-a)} \mod p && (6)\\
&\equiv mg^{ak} \cdot g^{k(p-1)-ak} \mod p && (7)\\
&\equiv m \cdot g^{k(p-1)} \mod p && (8)\\
&\equiv m \mod p && (9)
\end{aligned}$$

RSA

Allgemein:
1. Wähle zufällig große Primzahlen p,q
2. Setze $n = p \cdot q$
3. Wähle zufällig (e,d) mit $ed \equiv 1 mod(\phi(n))$, mit $\phi(n) = (p-1)(q-1)$

Public key: $pk = (e,n)$
Private key: $sk = (d,n)$
(Public und private key sind invers zueinander) Ver-/Entschlüsselung:
$E(pk,m) = m^e \mod n$
$D(sk,c) = c^d \mod n$
RSA Signatur:
$sign(sk,m) = h(m)^d \mod n$
$verify(pk,m,s) : [s^e = h(m)?]$

Shamir Secret-Sharing

Erweiterter Euklidischer Algorithmus (EEA)

Inverses Element $(a)^{-1}$ berechnen

iptables

Reliability

Availability

Sicherheitsziele

Kasiski-Test

OTP – Beweis perfekte Sicherheit