9 Schlüsseleinigung, Schlüsselaustausch

Ziel: Sicherer Austausch von Schlüsseln über einen unsicheren Kanal

- initiale Schlüsseleinigung für erste sichere Kommunikation
- Schlüsselerneuerung für weitere Kommunikation
  Forward Secrecy: Brechen eines alten Schlüssels soll folgende Nachrichten nicht kompromittieren

Hybridverfahren

- A will B eine vertrauliche Nachricht m übermitteln
- B besitzt ein Schlüsselpaar (pk, sk) (z.B. für das RSA-Verfahren)
- A vertraut dem öffentlichen Schlüssel pk (weiß, dass dieser B gehört)

$\boxed{A}$ öffentlicher Schlüssel: $pk = e$		$\boxed{B}$ geheimer Schlüssel: $sk = d$
choose random key k compute $k' = k^e \bmod n$ and $c = \mathrm{enc}(k, m)$	$\xrightarrow{k', c}$	compute $k = k'^d \bmod n$ and $m = \mathrm{dec}(k, c)$

Eigenschaften:

- Für jede Kommunikaiton neuer symmetrischer Schlüssel
- Keine Forward Secrecy: Brechen von sk führt zur Kompr. aller Schlüssel

Diffie-Hellman-Schlüsseleinigungsverfahren

Entwickelt von Diffie, Hellman und Merkle 1976
Sicherheit beruht auf Diskreten Logarithmusproblems

Also:

- Wir rechnen in $\mathbb{Z}_p^*$, p sehr große Primzahl
- Wir benötigen einen Erzeuger $g \in \mathbb{Z}_p^*$
  (um kleine Untergruppen auszuschließen)

$\boxed{A}$ Parameter (g, p)		$\boxed{B}$ Parameter (g, p)
choose random x		
compute $X = g^x \bmod p$	$\xrightarrow{X}$	
		choose random y
	$\xleftarrow{Y}$	compute $Y = g^y \bmod p$
comp. $Y^x = (g^y)^x = g^{xy} \bmod p$		comp. $X^y = (g^x)^y = g^{xy} \bmod p$

Abbildung 2: Diffie-Hellman-Schlüsseleinigungsverfahren (DH)

Aus g^{xy} lassen sich z.B. Schlüssel für Secure Messaging ableiten:

- $k_E := \text{Hash}(g^{xy}||0x00)$ Schlüssel für Verschlüsselung
- $k_A := \text{Hash}(g^{xy}||0x01)$ Schlüssel für MAC

Sicherheit des Diffie-Hellman-Schlüsseleinigungsverfahrens

Angreifer kennt Parameter (g, p) und sieht $g^x \bmod p$ und $g^y \bmod p$

Ziel: Bestimmung von g^{xy} (das gemeinsame Geheimnis)

- Einzige derzeit bekannte Möglichkeit: Bestimme x oder y
  (d.h. berechne $\log_g g^x$ oder $\log_g g^y$)

Forward Secrecy, wenn immer neue Zufallszahlen x, y gewählt werden

- Umgesetzt in tls (siehe Kapitel Internetsicherheit)
- Kürzel DHE (E für ephemral (flüchtig))

Aber folgender Angriff möglich:

$\boxed{A}$		$\boxed{O}$ (Angreifer)		$\boxed{B}$
choose random x		choose random z		
$X = g^x \bmod p$	$\xrightarrow{X}$	$Z = g^z \bmod p$	$\xrightarrow{Z}$	choose random y
	$\xleftarrow{Z}$		$\xleftarrow{Y}$	$Y = g^y \bmod p$
$Z^x = g^{xz} \bmod p$		$X^z = g^{xz} \bmod p$		
		$Y^z = g^{yz} \bmod p$		$Z^y = g^{yz} \bmod p$

Abbildung 3: Man-in-the-Middle Angriff

Nicht A und B berechnen Geheimnis, sondern A mit O und B mit O

Lösung: Authentisierung der Schlüsselanteile (MAC, Signatur, Passwort)

Übung: Geben Sie ein sicheres Schlüsselaustauschprotokoll unter Nutzung von Diffie-Hellman und Signaturverfahren an.

SPEKE (Simple Password Exponential Key Exchange)

Beispiel für ein Password-Authenticated Key Agreement Protocol.

- Parameter: q prim mit $p := 2q + 1$ prim und Hashfunktion H.
- A und B haben gemeinsames Passwort π.

$\boxed{A}$ Parameter (p, H, π)		$\boxed{B}$ Parameter (p, H, π)
Berechne $g = H(\pi)^2 \bmod p$		Berechne $g = H(\pi)^2 \bmod p$
choose random x		
compute $X = g^x \bmod p$	$\xrightarrow{X}$	
		choose random y
	$\xleftarrow{Y}$	compute $Y = g^y \bmod p$
comp. $Y^x = (g^y)^x = g^{xy} \bmod p$		comp. $X^y = (g^x)^y = g^{xy} \bmod p$

Abbildung 4: Simple Password Exponential Key Exchange (SPEKE)

- Wahl von $p = 2q+1$ prim: $\mathbb{Z}_p^* = \{1, \dots, p-1\}$ hat genau 2 Untergruppen
  - Eine der Ordnung 2, eine der Ordnung q
  - Ordnungen von Untergruppen teilen Ordnung der großen Gruppe
    Da $|\mathbb{Z}_p^*| = p - 1 = 2q$ und q prim, gibt es nur die Teiler 2 und q
  - $g = H(\pi)^2 \bmod p$: g ist Erzeuger der großen Untergruppe (Übung)

Needham-Schroeder-Protokoll

- Datenaustausch in dezentralen Netzen
- Schlüsselaustausch und Authentisierung
- Umgesetzt in Kerberos

Zwei Versionen: symmetrisch und asymmetrisch.

Symmetrische Version:

- Es wird ein Authentifizierungsserver AS benötigt
- A und B haben jeweils symmetrische Schlüssel K_A und K_B
  (z.B. für AES, diese sind AS bekannt)

$\boxed{A}$ K_A		$\boxed{AS}$ K_A, K_B		$\boxed{B}$ K_B
Wähle Nonce N_A	$\xrightarrow{A,B,N_A}$	Wähle Schlüssel K		
		$T_B = \text{enc}_{K_B}(K\|\|A)$		
	$\xleftarrow{T_A}$	$T_A = \text{enc}_{K_A}(N_A\|\|B\|\|K\|\|T_B)$		
$N_A\|\|B\|\|K\|\|T_B = \text{dec}_{K_A}(T_A)$			$\xrightarrow{T_B}$	$K\|\|A = \text{dec}_{K_B}(T_B)$
				Wähle Nonce N_B
			$\xleftarrow{T_{BA}}$	$T_{BA} = \text{enc}_K(N_B)$
$N_B = \text{dec}_K(T_{BA})$				
$T_{AB} = \text{enc}_K(N_B - 1)$			$\xrightarrow{T_{AB}}$	

Abbildung 5: Needham-Schroeder-Protokoll

Eigenschaften:

- A, B nutzen den von AS generierten Schlüssel K zur Kommunikation
- K wird verschlüsselt ausgetauscht
  (mit dem nur A, AS bzw. B, AS bekannten Schlüssel K_A bzw. K_B)
- A und B wissen, mit wem sie kommunizieren
  - Nur A und B können Schlüssel K entschlüsseln
  - Identitäten (d.h. A und B) werden mit verschlüsselt
    Damit kein MitM-Angriff möglich
- Replay-Attacken werden durch Nutzung von Nonces verhindert
  - Replay-Attacke: Einspielen einer zuvor abgehörten Verbindung
  - Nonce: number only used once
  - Beide nutzen Nonces (beide schließen Replay-Attacke aus)
    * Nur A und AS können T_A generieren
    * Nur A und AS können T_{AB} generieren

Kritik: Nur in der Vorlesung

Übung: Beschreiben Sie die asymmetrische Version des Protokolls.
Beschreiben Sie den von Lowe gefundenen Angriff auf diese Version[1].

[1] Gavin Lowe, An Attack on the Needham-Schroeder Public-Key Authentication Protocol (1995) (http://web.cs.wpi.edu/ cs564/f12/papers/lowe95.pdf)