Digitale Unterschriften mit ElGamal

Seminar Kryptographie und Datensicherheit

Institut für Informatik

Andreas Havenstein

Inhalt

- Einführung
- Signatur Systeme
- RSA Signatur Systeme
- Angriffe auf Signatur Systeme
- Signaturen und Verschlüsselung
- ElGamal Signatur Systeme
- Ausblick
- Quellen

Einführung

- konventionelle Unterschrift (handschriftlich) werden benutzt um Dokumente mit Personen zuzuorden
- Personen erklären sich verantwortlich für Dokumente (z.B. Verträge, Überweisungen, Briefe)
- Signature Systeme ermöglichen digitale Unterschriften

Einführung

- Dokumente mit einer digitalen Unterschrift versehen.
- Unversehrtheit der übermittelten Nachricht (Integrität) sichergestellt
- Nachricht wurde vom richtigen Absender (Authentizität) versandt

Einführung

- konventionelle Unterschrift ist Teil eines physischen Dokuments
- einfach (aber nicht sicher) durch Vergleich zu überprüfen
- kopieren möglich aber schwer und nicht perfekt

- digitale Unterschriften sind nicht an ein Dokument gebunden
  - müssen gebunden werden
- können sehr sicher mit Verifikationsalgorithmen überprüft werden
  - Verifikationsalgorithmus ist öffentlich
- Nachricht und Unterschrift lassen sich sehr leicht, perfekt kopieren
  - Nachricht muss zusätzliche Informationen enthalten

Signatur Systeme

- x - Nachricht
- y - Signatur (Unterschrift)
- (x,y) - Nachricht und Signatur bilden die signierte Nachricht
- $y = sig(x)$ - Signierungsalgorithmus erzeugt Signatur
- $ver(x,y)$ - Verifizierungsalgorithmus prüft ob Nachricht und Signatur zusammen passen

Definition

Ein **Signatur System** ist ein 5-Tupel $(\boldsymbol{P},\boldsymbol{A},\boldsymbol{K},\boldsymbol{S},\boldsymbol{V})$ wobei folgende Bedingungen erfüllt sind:

- $\boldsymbol{P}$ ist endliche Menge an möglichen Nachrichten.
- $\boldsymbol{A}$ ist endliche Menge an möglichen Signaturen.
- $\boldsymbol{K}$ ist endliche Menge der möglichen Schlüssel.
- Für jedes $k \in \boldsymbol{K}$ gibt es einen signier Algorithmus $sig_k \in \boldsymbol{S}$ und einen vertifizier Algorithmus $ver_k \in \boldsymbol{V}$.
- Mit $sig_k : \boldsymbol{P} \rightarrow \boldsymbol{A}$ und $ver_k : \boldsymbol{P} \times \boldsymbol{A} \rightarrow \{true, false\}$ gilt für alle Nachrichten $x \in \boldsymbol{P}$ und für alle Signaturen $y \in \boldsymbol{A}$

  $ver(x,y) = true$, wenn $y = sig(x)$

  $ver(x,y) = false$, wenn $y \neq sig(x)$.
- Ein Paar (x,y) mit $x \in \boldsymbol{P}$ und $y \in \boldsymbol{A}$ wird signierte Nachricht genannt.

Einwegfunktion

- Eine Einwegfunktion ist eine mathematische Funktion, die im Sinne der Komplexitätstheorie "schwer" umzukehren ist.
- Die Berechnung des Funktionswerts $y = f(x)$ ist „einfach“
- Die Berechnung des Inversen zu einem bekannten Funktionswert y, so dass $f(x) = y$, ist "schwer"
- Trapdoor-Einwegfunktionen: Invertierung wird mit zusätzlichen geheimen Informationen leichter.
  - z.B.: bei asymmetrischen Verschlüsselungsverfahren der private Key
- Beispiel: bei RSA: Primfaktorzerlegung
  bei Elgamal: diskrete Logarithmus

Definition

RSA Signatur System

Sei $n = pq$, wobei p und q Primzahlen. Sei $\boldsymbol{P} = \boldsymbol{A} = \mathbb{Z}_n$ und

$$\boldsymbol{K} = \{(n,p,q,a,b) : n = pq,\ p, q \text{ prim},\ ab \equiv 1\ (mod\ \varphi(n))\}$$

n, b sind public key; p, q, a sind private key

Für $k = (n,p,q,a,b)$, sei

$$sig_k(x) = x^a\ mod\ n$$

und

$$ver_k(x,y) = true \Leftrightarrow x \equiv y^b\ mod\ n$$

$(x, y \in \mathbb{Z}_n)$.

Beispiel RSA Signatur

$p = 11$ $q = 13$ $N = pq = 143$ $\varphi(n)=(p-1)(q-1) = 120$

$b = 23$ $a = 47$

$x = 100$

$$x^a \bmod n = y$$

$$100^{23} \bmod 143 = 133$$

$$y^b \bmod n = x$$

$$133^{47} \bmod 143 = 100$$

$x = 100$

Signaturen und Hashfunktionen

- Hashfunktion $z = h(x)$ wandelt Nachricht x in eine Kurzfassung z
- Signaturfunktion $y=sig_k(z)$ signiert Kurzfasung z
- vertifizier Funktion $ver_k(z,y)$ z wird noch mal vom Empfänger mit $h(x)$ berechnet

Nachricht	x
	$\downarrow$
Nachricht- Kurzfassung	$z = h(x)$
	$\downarrow$
Signatur	$y=sig_k(z)$
	$\downarrow$
Vertifizierung	$ver_k(z,y)$

Angriffe auf Signatur Systeme

- „key-only“ Angriff
  - Angreifer Oscar benutzt Alice's public key und ver_k
  - Oscar kann mit genügend Zeit eine gültige Signatur zu einer gegebenen Nachricht erzeugen (alle mögliche Signaturen testen)
- „known message“ Angriff
  - Oscar besitzt eine Liste von Alice signierten Nachrichten $(x_1,y_1),(x_2,y_2)$, ...
  - Oscar kann mit genügend Zeit und Rechenleistung $y_i=sig(x_i)$, i = 1, 2, ... bestimmen
- „chosen message“ Angriff
  - Oscar fordert für eine Liste von Nachrichten x_1,x_2, ... Signaturen von Alice $y_i=sig(x_i)$, i =1, 2, ... an

Mögliche Ziele von Angriffen

- „total break“
  - Angreifer ist ist der Lage Alice's private key zu bestimmen und somit jede Nachricht mit einer gültigen Signatur zu versehen
- „selective forgery“ (ausgewählte Fälschung)
  - mit einer Wahrscheinlichkeit (≥ 0) ist der Angreifer in der Lage eine Nachricht mit einer gültigen Signatur zu versehen
- „existential forgery“ (existentiell Fälschung)
  - Angreifer ist in der Lage für mindestens 1 Nachricht eine gültige Signatur zu erzeugen

Signaturen und Verschlüsselung

*Erst signieren oder **erst verschlüsseln**?*

- Alice möchte Bob eine signierte und verschlüsselte Nachricht schicken
  - verschlüsselt x mit $z = e_{Bob}(x)$
  - signiert mit $y=sig_{Alice}(z)$
- Bob erhält (z,y) von Alice
  - überprüft y mit $ver_{Alice}(z,y)$
  - entschlüsselt z mit $d_{Bob}(z)= x$
- Oscar (Man in the Middle) fängt Alice's Sendung (z,y) an Bob ab
  - signiert mit $y' = sig_{Oscar}(z)$, schickt (z,y') an Bob
  - Bob überprüft y' mit $ver_{Oscar}(z,y')$ glaubt ***z* stammt von Oscar**

Signaturen und Verschlüsselung

Erst signieren *oder erst verschlüsseln?*

- Alice möchte Bob eine signierte und verschlüsselte Nachricht schicken
  - signiert mit $y=sig_{Alice}(x)$
  - verschlüsselt dann beides (x,y) mit $z = e_{Bob}(x,y)$
- Bob erhält z von Alice
  - entschlüsselt z mit $d_{Bob}(z)=(x,y)$
  - überprüft dann y mit $ver_{Alice}(x,y)$
- Oscar hat keine Chance :-)

Definition

ElGamal Signatur System

Sei p prime und das diskrete Logarithmusproblem in $\mathbb{Z}_p$ sei nicht lösbar, und sei

$\alpha \in \mathbb{Z}_p^*$ ein primitive Element. Sei $\boldsymbol{P} = \mathbb{Z}_p^*$, $\boldsymbol{A} = \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$ und sei

$$\boldsymbol{K} = \{(p, \alpha, a, \beta) : \beta \equiv \alpha^a \ (mod\ p)\}.$$

p, α, β sind puplic key; a ist private key

Für $k = (p, \alpha, a, \beta)$ und für eine geheime Zufallszahl $r \in \mathbb{Z}_{p-1}^*$, sei

$$sig_k(x,r) = (\gamma,\delta),$$

wobei

$$\gamma = \alpha^r \ mod\ p \text{ und } \delta = (x - a\gamma)r^{-1} \ mod\ (p - 1)$$

Für $x, \gamma \in \mathbb{Z}_p^*$ und $\delta \in \mathbb{Z}_{p-1}$, sei

$$ver_k(x, (\gamma,\delta)) = true \quad \Leftrightarrow \quad \beta^\gamma \gamma^\delta \equiv \alpha^x \ (mod\ p).$$

Beispiel Elgamal Signatur

$p = 467$ $\alpha = 2$ $a = 127$

$\beta = \alpha^a \bmod p$ $\beta = 2^{127} \bmod 467$ $\beta = 132$

$x = 100$ $r = 213$ $sig(x,r) = (\gamma,\delta)$

$\gamma = \alpha^r \bmod p$ $\gamma = 2^{213} \bmod 467$ $\gamma = 29$

$\delta = (x - a\gamma)r^{-1} \bmod (p - 1)$

$\delta = (100 - 127 \times 29)\ 431 \bmod 466$ $\delta = 51$

$ver_k(x, (\gamma,\delta)) = true \Leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$.

$132^{29} \times 29^{51} \equiv 2^{100} \pmod{467}$

$189 \equiv 189 \pmod{467}$

Sicherheit des ElGamal Signatur Systems

- analog zur Sicherheit des ElGamal Krypto Systems
- möchte man für eine gegebene Nachricht *x* eine Signatur berechnen ohne das man *a* (private key) kennt
  - wählt man γ muss man um δ zu finden den diskreten Logarithmus $\log_{\gamma} \alpha^x \beta^{\gamma}$ berechnen
  - wählt man δ muss man um γ zu finden die Gleichung $\beta^{\gamma} \gamma^{\delta} \equiv \alpha^x \ (mod\ p)$ lösen
  - man kann versuchen γ δ gleichzeitig zu berechnen
- für keinen Weg wurde bisher eine Lösung gefunden, aber es wurde auch nicht bewiesen das es keine Lösung gibt

Beispiel eines „key only“ Angriffs

- 2 integer Werte *i* und *j* so dass gilt: $0 \leq i,j \leq p - 2$
- angenommen wir drücken γ in der Form $\gamma = \alpha^i \beta^j \bmod p$ aus
- eingesetzt in $\beta^\gamma \gamma^\delta \equiv \alpha^x \ (mod\ p)$ liefert:

$$\beta^\gamma (\alpha^i \beta^j)^\delta \equiv \alpha^x \ (mod\ p)$$
$$\beta^{\gamma + j\delta} \equiv \alpha^{x - i\delta} \ (mod\ p)$$

- letzte Konvergenz wird erfühlt durch:

  $x - i\,\delta \equiv 0 \ (mod\ p\text{-}1)$ und $\gamma - j\,\delta \equiv 0 \ (mod\ p\text{-}1)$

- Konvergenz durch gegebenes i und j aufgelöst liefert:

  $\delta = -\gamma j^{-1} \ mod\ (p\text{-}1)$, und

  $x = -\gamma i j^{-1} \ mod\ (p\text{-}1)$

Beispiel eines „key only“ Angriffs

- 2 integer Werte *i* und *j* so dass gilt: $0 \leq i,j \leq p - 2$
- angenommen wir drücken γ in der Form $\gamma = \alpha^i \beta^j \bmod p$ aus
- eingesetzt in $\beta^\gamma \gamma^\delta \equiv \alpha^x \ (mod\ p)$ liefert:

$$\beta^\gamma (\alpha^i \beta^j)^\delta \equiv \alpha^x \ (mod\ p)$$
$$\beta^{\gamma + j\delta} \equiv \alpha^{x - i\delta} \ (mod\ p)$$

- letzte Konvergenz wird erfühlt durch:

  $x - i\,\delta \equiv 0 \ (mod\ p\text{-}1)$ und $\gamma - j\,\delta \equiv 0 \ (mod\ p\text{-}1)$

- Konvergenz durch gegebenes i und j aufgelöst liefert:

  $\delta = -\gamma j^{-1} \ mod\ (p\text{-}1)$, und

  $x = -\gamma i j^{-1} \ mod\ (p\text{-}1)$

Beispiel eines „key only“ Angriffs

- public key: $p = 467$, $\alpha = 2$, $\beta = 132$
- wähle i und j: i = 99 j = 179
- $\gamma = \alpha^i \beta^j \bmod p$

  $\delta = -\gamma j^{-1} \; mod \; (p\text{-}1)$

  $x = -\gamma ij^{-1} \; mod \; (p\text{-}1)$

- $\gamma = 2^{99}\, 132^{179} \; mod \; 467 = 117$

  $\delta = -\, 117 \times 151 \; mod \; 466 = 41$

  $x = \; 99 \times 41 \; mod \; 466 = 331$

- (117, 41) *sollte* eine gültige Signatur für die Nachricht 331 sein

Beispiel eines „key only“ Angriffs

- $\gamma = 2^{99}\ 132^{179}\ mod\ 467 = 117$
  $\delta = -\ 117 \times 151\ mod\ 466 = 41$
  $x = \ 99 \times 41\ mod\ 466 = 331$
- Verifizierung:

  $ver_k(x, (\gamma,\delta)) = true \quad \Leftrightarrow \quad \beta^{\gamma}\ \gamma^{\delta} \equiv \alpha^{x}\ (mod\ p).$
  $132^{117} \times 117^{41} \equiv 2^{331} \pmod{467}$
  $303 \equiv 303 \pmod{467}$

- (117, 41) ist eine gültige Signatur für die Nachricht 331
- Fälschung einmal gelungen („existential forgery“)

Varianten des ElGamal Signature Systems

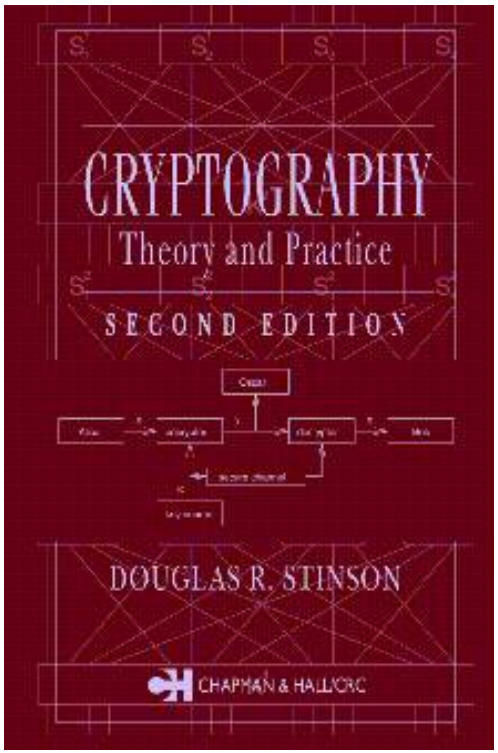
- Schnorr Signatur System
- DSA
- Elliptic Curve DSA

Ausblicke

- One- Time- Signatur System
  - z.B.: Lamport Signatur System
  - signieren von nur einer Nachricht verwendet wird
- Full Domain Hash Signatur System
  - Empfänger kann überprüfen, ob die Nachricht, durch einen Dritten verändert wurde
- Undeniable Signatur System
  - die eine eindeutige Zuordnung einer Person ermöglichen bzw. gefälschte Signaturen erkennen
  - benötigt zur Verifikation immer die Mitarbeit des Erzeugers der Signatur
- Fail- Stop- Signaturen
  - Unterzeichner kann beweisen das seine Signatur gefälscht ist

Quellen

- Douglas R. Stinson: **Cryptography: Theory and Practice,** 2nd Edition, Chapman & Hall/CRC 2002
- Menezes, Alfred J., van Oorschot, Paul C., Vanstone, Scott A., **Handbook of Applied Cryptography**, 5th Printing, CRC Press, 2001
- `www.wikipedia.de`



VIELEN DANK!

FRAGEN?