Inhalt

Einleitung

- PublicKey Verfahren können auch dazu benutzt werden, um Dokumente zu signieren
- Alice signiert ein Dokument m, indem sie ihren geheimen Schlüssel d_A auf das Dokument anwendet. Das signierte Dokument ist dann das Paar $(m, d_A(m))$.
- Bob verifiziert die Signatur, indem er mit dem öffentlichen Schlüssel e_A von Alice $e_A(d_A(m))$ berechnet und prüft, ob der berechnete Wert mit m übereinstimmt.
- Aus Laufzeitgründen wird statt des gesamten Dokuments m nur ein Hash-Wert $h(m)$ davon signiert. Dabei ist h eine öffentliche kollisionsresistente Hashfunktion. Das signierte Dokument ist dann also das Paar $(m, d_A(h(m)))$.

Digitale Signatur mit RSA

Das Erstellen einer digitalen Signatur mit Hilfe des RSA Verfahrens folgt direkt aus den einleitenden Ausführungen

Aufgabe: Bestimme die RSA-Signatur einer Nachricht m mit dem Hashwert $h(m) = 12345$. Der RSA-Modul sei $n = 28829$ und der öffentliche Schlüssel von Alice laute $e_A = 59$.

Digital Signature Algorithm DSA (El Gamal)

- Im El Gamal Verfahren sind Verschlüsselung und Entschlüsselung nicht direkt vertauschbar. Deshalb braucht es leichte Modifikationen:

- Alice wählt einen öffentlichen Schlüssel (p, g, A). Ihr privater Schlüssel ist dabei eine zufällig gewählte Zahl $a \in \{2, \ldots, p-2\}$, wobei $A = g^a \pmod{p}$.
- Signatur von $h(m)$: Alice wählt eine Zufallszahl $k \in \{2, \ldots, p-2\}$ die zu $p-1$ teilerfremd ist.
- Sie berechnet:
  - $r = g^k \pmod{p}$
  - $s = k^{-1} \cdot (h(m) - a \cdot r) \pmod{p-1}$
- **Signatur:** (r, s)

Digital Signature Algorithm DSA (El Gamal)

- Verifikation der Signatur: Bob prüft, ob $1 \leq r \leq p - 1$ erfüllt ist. Falls nicht, so wird die Signatur zurückgewiesen.
- Sodann überprüft Bob, ob die Kongruenz

$$A^r \cdot r^s = g^{h(m)} \pmod{p}$$

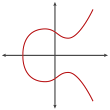
erfüllt ist. Wenn ja, so ist die Signatur gültig. Sonst nicht.

Digital Signature Algorithm DSA (El Gamal)

Aufgabe: Der öffentlichen Schlüssel von Alice laute $(p = 23, g = 7, A = 4)$. Dabei ist ihr geheimer Schlüssel $a = 6$. Alice möchte ein Dokument m mit dem Hashwert $h(m) = 7$ signieren. Bestimme und verifiziere die Signatur.

- Erstellung der Signatur des Dokuments m mit Hilfe des Hashwerts $h(m)$ und des Private Key
- Verifizieren durch Anwenden des Public Key
- Ein paar Worte der Warnung zur Anwendung des ECDSA

Digitale Signatur mit elliptischen Kurven

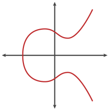


Schlüsselerzeugung

[Die farbigen Werte bilden ein Zahlen-Beispiel]

1. Wähle eine elliptische Kurve E über GF(p) für eine Primzahl p
   $E : y^2 = x^3 + 7$, $p = 67$,
2. Wähle einen Punkt P aus E, s.d. $n = |P|$ prim ist. P = (2,22), n = 79
3. Wähle einen **geheimen Schlüssel** d. 2
4. Gib **öffentlichen Schlüssel** $Q := d \cdot P$ bekannt.

Digitale Signatur mit elliptischen Kurven



Erstellen der Signatur

1. Wähle eine zufällige Zahl $k < n$. 3
2. Berechne $(x, y) := k \cdot P$
3. Setze $r := x \pmod{n}$
4. Setze $h(m) :=$ Hashwert des zu signierenden Dokuments m (dargestellt als Zahl) 17
5. Setze $s := (h(m) + r \cdot d) \cdot k^{-1} \pmod{n}$
6. Signatur von m: (r, s)

Verifikation der Signatur

Schritte, um zu überprüfen, ob (r, s) eine gültige Signatur ist:

1. Berechne $u := h(m) \cdot s^{-1} \pmod{n}$
2. Berechne $v := r \cdot s^{-1} \pmod{n}$
3. Berechne $(a, b) := u \cdot P + v \cdot Q$
4. Überprüfe, ob $r = a \pmod{n}$ gilt. (Sonst ist die Signatur nicht gültig.)

Digitale Signatur mit elliptischen Kurven

Nachweis der Funktionsweise

- $$\begin{aligned} uP + vQ &= h(m)s^{-1} \cdot P + rs^{-1}d \cdot P \\ &= (h(m)s^{-1} + rs^{-1}d) \cdot P \\ &= (h(m) + rd)s^{-1} \cdot P \end{aligned}$$
- Da $s := (h(m) + rd)k^{-1} \pmod{n}$, folgt $k = (h(m) + rd)s^{-}1 \pmod{n}$.
- Einsetzen ergibt: $uP + vQ = kP$
- Die x-Koordinate von kP ist r $\square$

Ein paar Worte der Warnung zu DSA und ECDSA

Umgang mit dem geheimen Schlüssel d und der Nonce k

- Nachdem wir die Funktionsweise von ECDSA jetzt kennen, ist es WICHTIG, folgende Punkte festzuhalten:
- Den geheimen Schlüssel d darf NUR die unterzeichnende Person kennen. Aber auch die Nonce (number used once) k, die bei der Erstellung der Signatur verwendet wird, darf NICHT veröffentlicht werden. Sonst kann daraus der geheime Schlüssel d berechnet werden:

Ein paar Worte der Warnung zu DSA und ECDSA

Umgang mit dem geheimen Schlüssel d und der Nonce k

- $s = (h(m) + r \cdot d) \cdot k^{-1} \pmod{n}$
- $k \cdot s = h(m) + r \cdot d \pmod{n}$
- $k \cdot s - h(m) = r \cdot d \pmod{n}$
- $d = (k \cdot s - h(m)) \cdot r^{-1} \pmod{n}$

Ein paar Worte der Warnung zu DSA und ECDSA

Umgang mit dem geheimen Schlüssel d und der Nonce k

- Der Unterzeichner muss also nicht nur den geheimen Schlüssel d, sondern auch JEDE irgendwann verwendete Nonce k geheim halten!
- Zusätzlich darf der Unterzeichner auch NIE dasselbe k für weitere Signaturen verwenden. Dies sieht ein Angreifer den betreffenden Signaturen sofort an, weil deren Komponenten r dann übereinstimmen.
- Ein Angreifer kann dann aus den Signaturen (r, s_1) und (r, s_2) die Nonce k leicht wie folgt berechnen (und damit dann den geheimen Schlüssel d wie oben berechnen):

Ein paar Worte der Warnung zu DSA und ECDSA

Umgang mit dem geheimen Schlüssel *d* und der Nonce *k*

- $s_1 = (h(m_1) + r \cdot d) \cdot k^{-1}$ und $s_2 = (h(m_2) + r \cdot d) \cdot k^{-1}$
- $s_1 - s_2 = (h(m_1) - h(m_2)) \cdot k^{-1}$
- $k \cdot (s_1 - s_2) = h(m_1) - h(m_2)$
- $k = (s_1 - s_2)^{-1} \cdot (h(m_1) - h(m_2))$