Kryptographie und Komplexität

Einheit 5

Kryptosysteme auf der Basis diskreter Logarithmen

1. El Gamal Systeme
2. Angriffe auf Diskrete Logarithmen
3. Elliptische Kurven

Schwächen des RSA

- **Schlüssel müssen sehr groß werden**
  - Faktorisierungsalgorithmen können Schlüssel bis 1024 Bit angreifen
  - Blockgröße muß auf 2048 Bit oder größer anwachsen
  - Wachsende Blockgröße macht Verschlüsselung ineffizient
    · Potenzierung modulo n benötigt $\mathcal{O}(\|n\|^3)$ Schritte
    · Zeit für Verschlüsselung langer Nachrichten wächst quadratisch
  - Verschlüsselung braucht neue algebraische Probleme als Fundament
    Schwer zu brechende kleine Schlüssel oder effizientere Verschlüsselung
- **Semantische Sicherheit nicht sichergestellt**
  - Zahlentheoretisches Verfahren enthält keine Randomisierung
  - Gleiche Nachrichten werden immer auf gleiche Art verschlüsselt
  - Verschlüsselungsprotokoll sollte Zufall mit einbauen

Diskrete Logarithmen

- **Umstellung der RSA Ver-/Entschlüsselung**
  - RSA: Gegeben $y = x^e \bmod n$ bestimme $x = \sqrt[e]{y} \bmod n$
  - DL: Gegeben $y = e^x \bmod n$ bestimme $x = \log_e y \bmod n$
  - Potenzierung mit x ist formulierbar für beliebige zyklische Gruppen
    e muß keine Zahl sein sondern nur Gruppenelement der Ordnung n
- **Algebraische Formulierung des Problems**
  - Sei $(G, \cdot)$ multiplikative Gruppe, g Element der Ordnung n
    Für $y \in \langle g \rangle$ ist der **diskrete Logarithmus von y zur Basis g**
    (bezeichnet als $\boldsymbol{x = \log_g y}$) die eindeutige Zahl $x<n$ mit $y = g^x$
- **Welche Gruppen sind geeignet?**
  - Prime Restklassen modulo einer Primzahl $(\mathbb{Z}_p^*, \cdot)$
  - Punktgruppe einer elliptischen Kurve über endlichen Körpern
  - Hyperelliptische Kurven, Gruppen imaginär-quadratischer Ordnungen...

  Verzicht auf numerische Struktur macht Logarithmen z.T. erheblich schwerer zu berechnen als Wurzeln über $\mathbb{Z}_n$ oder Faktorisierung

Diskrete Logarithmen am Beispiel

- **Logarithmen zur Basis 2 modulo 13**
  - Berechne Potenzen von 2 zur Basis 13 mit Elementen aus $\mathbb{Z}_{13}^*$

x	1	2	3	4	5	6	7	8	9	10	11	12
2^x	2	4	8	3	6	12	11	9	5	10	7	1

  - Umstellung nach Logarithmen (Ordnung von 2 ist $n = 12$)

y	1	2	3	4	5	6	7	8	9	10	11	12
$\log_2 y$	0	1	4	2	9	5	11	3	8	10	7	6

  - Logarithmus ist eine Zahl $x<n$, kein Gruppenelement

- **Logarithmen zur Basis 5 modulo 19**
  - Berechne Potenzen von 5 zur Basis 19 mit Elementen aus $\mathbb{Z}_{19}^*$

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
5^x	5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1

  - Ordnung von 5 ist nur $n = 9$: $\langle 5\rangle = \{1; 4; 5; 6; 7; 11; 16; 17\}$
  - Umstellung nach Logarithmen für die Elemente von $\langle 5\rangle$

y	1	4	5	6	7	9	11	16	17
$\log_5 y$	0	8	1	2	6	5	3	7	4

Kryptographie und Komplexität

Einheit 5.1

ElGamal Systeme

1. Diffie Hellman Schlüsselaustausch
2. ElGamal Verschlüsselungsverfahren
3. Korrektheit und Komplexität
4. Sicherheitsaspekte

Diffie Hellman Schlüsselaustausch

Sicherer Austausch von Schlüsseln

- **Basisprotokoll verwendet diskrete Logarithmen über $\mathbb{Z}_p$**
  - Wähle Primzahl p und Erzeuger g von $\mathbb{Z}_p$ mit $2 \leq g \leq p-2$
    p und g werden nicht geheim gehalten
  - Alice wählt zufällige Zahl $a \in \{0, .., p-2\}$ und berechnet $A = g^a \bmod p$
    Alice hält a geheim und schickt A an Bob
  - Bob wählt zufällige Zahl $b \in \{0, .., p-2\}$ und berechnet $B = g^b \bmod p$
    Bob hält b geheim und schickt B an Alice
  - Alice berechnet $B^a \bmod p = g^{a \cdot b} \bmod p$
    Bob berechnet $A^b \bmod p = g^{a \cdot b} \bmod p$
  - Gemeinsamer Schlüssel $K = g^{a \cdot b} \bmod p$ ist nur Alice und Bob bekannt
- **Beispiel für n=17 und g=3**
  - Alice wählt a=7 und berechnet $A = g^a \bmod 17 = 2187 \bmod 17 = 11$
  - Bob wählt b=4 und berechnet $B = g^b \bmod 17 = 81 \bmod 17 = 13$
  - Der gemeinsame Schlüssel ist $K = A^b \bmod 17 = 14641 \bmod 17 = 4$

Sicherheit des Diffie Hellman Schlüssels

- **Angreifer kennt p, g, A und B**
  - p, g, A und B wurden über unsichere Kanäle ausgetauscht
  - Methode zur Bestimmung des gemeinsamen Schlüssels K ist bekannt
  - Um $K = A^b \bmod p = B^a \bmod p$ zu berechnen, müsste Angreifer entweder a oder b bestimmen können
- **Angreifer muß diskreten Logarithmus lösen**
  - Um K zu bestimmen muß Angreifer entweder $a = \log_g A$ oder $b = \log_g B$ ausrechnen können
  - Andere Methode, gemeinsamen Schlüssel zu brechen ist nicht bekannt
  - Äquivalenz des Diffie-Hellman Problems (bestimme $g^{a \cdot b}$ aus g^a und g^b) zum Problem des diskreten Logarithmus (berechne $\log_g A$) nicht bewiesen
- **Berechnung diskreter Logarithmen ist schwer**
  - Beste bekannte Verfahren für $\mathbb{Z}_p$ liegen in $L_n[1/3,\ 1.92]$
    Effizienteste Verfahren sind nur auf numerische Gruppen anwendbar

Diffie Hellman auf beliebigen Gruppen

- **Protokoll nahezu identisch**
  - Wähle Erzeuger g der Gruppe G mit Ordnung n
  - Alice wählt zufällige Zahl $a \in \{1, .., n-1\}$ und berechnet $A = g^a \in G$
  - Bob wählt zufällige Zahl $b \in \{1, .., p-1\}$ und berechnet $B = g^b \in G$
  - Alice berechnet $B^a = g^{a \cdot b}$ – Bob berechnet $A^b = g^{a \cdot b}$
  - Gemeinsamer Schlüssel $K = g^{a \cdot b}$ ist nur Alice und Bob bekannt
- **Rahmenbedingung an Auswahl der Gruppen**
  - Gruppen müssen zyklisch sein und erzeugende Elemente haben
  - Multiplikation und Potenzierung muß effizient implementierbar sein
  - Diffie-Hellman Problem muß schwer zu lösen sein
    $(\mathbb{Z}_p, +)$ ist ungeeignet, da $\log_g A = A \cdot g^{-1}$ leicht zu berechnen
- **Vorteil nichtnumerischer Gruppen**
  - Gruppenoperationen sind komplexer und schwerer zu invertieren
  - Lösung des Diffie-Hellman Problems kann sich nicht (nur) auf zahlentheoretische Zusammenhänge stützen

Größere Sicherheit bei geringerer Schlüssellänge möglich

Das ElGamal Verschlüsselungsverfahren

- **Public-Key Verfahren von Taher ElGamal (1985)**
  - Sicherheit basiert auf Schwierigkeit des DL Problems

    Berechnung diskreter Logarithmen ist nicht in akzeptabler Zeit möglich
  - Eng verwandt mit Diffie-Hellman Schlüsselaustauschprotokoll
- **Verwendet Potenzierung von Gruppenelementen**
  - Nachricht ist Exponent der Potenzierung (anstelle der Basis)
- **Protokoll enthält Randomisierung**
  - Empfänger legt Diffie-Hellman Teilschlüssel offen
  - Absender verwendet Zufallszahlen zur Verschlüsselung
  - Nachricht enthält Diffie-Hellman Teilschlüssel des Absenders

Effiziente Ver-/Entschlüsselung und semantische Sicherheit

Das ElGamal Verfahren im Detail

- **Schlüsselerzeugung**
  - Wähle eine große Zahl p und ein Gruppenelement g der Ordnung p
  - Wähle ein zufälliges $a \in \{0, .., p-1\}$ und berechne $A = g^a$
  - Lege p, g, A offen, halte a geheim
- **Verschlüsselung**
  - Gesamtschlüssel ist $K := (p, g, a, A)$, wobei p, g, A öffentlich
  - Text wird in Blöcke der Länge $\log_2 p/8$ zerlegt (ein Byte pro Buchstabe)
    Ein Textblock wird als Darstellung eines Gruppenelements x interpretiert
  - Absender wählt zufälliges $b \in \{0, .., p-1\}$ und berechnet $B := g^b$
  - Absender verschlüsselt x zu $y := x \cdot A^b$
  - Erzeugter Schlüsseltext ist $\boldsymbol{e_K(x, b)} = (B, y)$
- **Entschlüsselung**
  - Empfänger berechnet $\boldsymbol{d_K(B, y)} = y \cdot (B^a)^{-1}$

Das ElGamal Verfahren am Beispiel

- **Einfaches Beispiel in Restklassengruppen $(\mathbb{Z}_p^*, \cdot)$**
  - Alice wählt $p = 23$, $g = 7 \in \mathbb{Z}_p^*$, $a = 6$ und berechnet $A = 7^6 \bmod 23 = 4$
  - Bob wählt $b = 3$ und berechnet $B = 7^3 \bmod 23 = 21$ und $A^b \bmod 23 = 18$
  - Verschlüsselung der Nachricht $x = 7$ gibt Schlüsseltext $(B, y) = (21, 11)$
  - Alice erhält (B, y) und berechnet $(B^a)^{-1} = B^{p-1-a} \bmod 23 = 9$
  - Entschlüsselung von y liefert $11 \cdot 9 \bmod 23 = 7$
- **Berechnungen sind einfacher als bei RSA**
  - Bob kann $B := g^b$ und A^b im Voraus berechnen
  - Alice kann Invertierung von B^a in $\mathbb{Z}_p^*$ als $B^{p-1-a} \bmod p$ berechnen
- **Gute statistische Streuung der Schlüsseltexte**
  - Verschlüsselung von 4 5 6 7 8 9 mit $b = 3$ ergibt y-Werte 3 21 16 11 6 1
  - $b = x$ ergibt (B, y)-Paare (9, 12), (17, 14), (4, 12), (5, 10), (12, 3), (15, 2)
  - Verschlüsselung mit Zufallswerten für b ergibt “beliebige” (B, y)-Paare

ElGamal Verfahren mit 32-Bit Zahlen

- **Schlüsselerzeugung**
  - Alice wählt $p = 3013183829$, $g = 2719263871 \in \mathbb{Z}_p^*$, $a = 1000000000$
  - Alice berechnet $A = g^a \bmod p = 2006813696$ und veröffentlicht p, g, A
- **Verschlüsselung**
  - Bob wählt $b = 50000000000$ und berechnet
    $B = g^b \bmod p = 1948493095$ und $A^b \bmod p = 2537054755$
  - Klartext ist ‘Test’, codiert als Zahl $x = 1415934836$
    Schlüsseltext ist $(B, y) = (B, x{\cdot}A^b \bmod p) = (1948493095, 811008022)$
- **Entschlüsselung**
  - Alice berechnet $(B^a)^{-1} = B^{p-1-a} \bmod p = 475183925$
  - Multiplikation mit y liefert $811008022{\cdot}(B^a)^{-1} \bmod p = 1415934836$
  - Konversion in 4-Buchstaben-Block liefert ursprünglichen Klartext
- **Realistische Blocklänge ist 512 oder 1024 Bit**
  - Schnelle Primzahltests/Potenzierungsalgorithmen wie bei RSA nötig

Korrektheit und Komplexität von ElGamal

- **Korrektheit: Ver-/Entschlüsselung sind invers**
  - Es ist $d_K(B,y) = y\cdot(B^a)^{-1}$ wobei $e_K(x,b) = (B,y) = (g^b, x\cdot A^b)$
  - Für beliebige b ist $d_K(e_K(x,b)) = x\cdot A^b\cdot(B^a)^{-1} = x\cdot g^{a\cdot b}\cdot(g^{b\cdot a})^{-1} = x$
- **Aufwand für Auswahl des Schlüssels (einmalig)**
  - Erzeugung der Primzahlen p (z.B. mit Miller-Rabin) $\mathcal{O}(\|p\|^3)$
  - Wahl von g, a und Berechnung von $A = g^a$ in $\mathbb{Z}_p^*$ $\mathcal{O}(\|p\|^3)$
- **Aufwand für Ver- und Entschlüsselung**
  - Kein Aufwand für Umwandlung zwischen Text und Zahlen
  - Potenzierung von $8|w|/\|p\|$ Blöcken in $\mathbb{Z}_p^*$ $\mathcal{O}(|w|\cdot\|p\|^2)$
  - Schlüsseltext doppelt so lang wie Klartext (Nachrichtenexpansion)
  - Beschleunigung mit chinesischem Restsatz nicht möglich
- **Trotzdem (bei festem b) schneller als RSA**
  - Vorberechnung von $B := g^b$, A^b und $(B^a)^{-1}$ ist möglich
    Ver-/Entschlüsselung benötigt nur eine Multiplikation $\mathcal{O}(|w|\cdot\|p\|)$

Aufwand in anderen Gruppen als $\mathbb{Z}_p$ abhängig von Komplexität der Gruppenoperation ·

Sicherheit von ElGamal

- **Gleich schwer wie Diffie-Hellman Problem**
  - Kann ein Angreifer das DH-Problem lösen, so kann er aus (B, y) und $K = (p, g, A)$ auch $B^a = g^{a \cdot b}$ und damit $x = y \cdot (B^a)^{-1}$ bestimmen
  - Kann ein Angreifer das ElGamal System brechen, so kann er aus $A = g^a$, $B = g^b$ und festem $y = 1$ die Zahl $x = 1 \cdot (B^a)^{-1} = (g^{a \cdot b})^{-1}$ bestimmen. Invertierung von x löst das Diffie-Hellman Problem
- **Sicherheit des geheimen Schlüssels**
  - Bestimmung von a benötigt Lösung des DL Problems
    Äquivalenz zum Problem des diskreten Logarithmus nicht bewiesen
  - Berechnung diskreter Logarithmen ist i.w. exponentiell in $\|p\|$ $\mapsto$ §5.3
- **Semantische Sicherheit durch Randomisierung**
  - Zufällige Wahl von b macht Verschlüsselung nichtdeterministisch
  - Gleiche Klartexte werden verschieden verschlüsselt
    Resultierende Schlüsseltexte (B, y) sind zufällig und gleichverteilt

ElGamal ist sicherer oder effizienter als RSA