Das RSA Verfahren

Eine erste Einführung

https://iuk.one/1010-1011

Clemens H. **Cap**
ORCID: 0000-0003-3958-6136

Department of Computer Science
University of **Rostock**
Rostock, Germany
clemens.cap@uni-rostock.de

15. 12. 2020 Vers. 4

Inhaltsverzeichnis

Ziel

RSA...

Ist das bedeutendste und bekannteste asymmetrische Verschlüsselungsverfahren.

Ist nicht notwendigerweise das beste asymmetrische Verschlüsselungsverfahren.

Ist relativ leicht zu verstehen.

Benötigt für sichere Anwendung viel Zusatzwissen.

Dient hier als Prototyp für kryptographische Algorithmen.

1. Voraussetzungen

Ziel: Die mathematischen Voraussetzungen für ein Verständnis des RSA-Verfahrens erwerben.

1. Voraussetzungen

Modulo Rechnung

Definition

Zwei ganze Zahlen a und b heißen **kongruent modulo** einer ganzen Zahl m, wenn sie **bei Division** durch m **denselben ganzzahligen Rest** haben.

Wir schreiben: $a \equiv b \bmod m$

Formal: Für $a, b \in \mathbb{Z}, n \in \mathbb{N}$ sei definiert:
$a \equiv b \bmod n \ :\Leftrightarrow \ \exists \alpha, \beta, r \in \mathbb{Z} : a = \alpha \cdot n + r \wedge b = \beta \cdot n + r$

Beispiel: $9 \equiv 4 \bmod 5$ da $9 = 1 \cdot 5 + 4$ und $4 = 0 \cdot 5 + 4$.

Euklidischer Algorithmus: Eine Wiederholung an einem Beispiel

Der **Euklidische Algorithmus** bestimmt den größten gemeinsamen Teiler zweier Zahlen.

Wir lernen ihn hier kennen (bzw. wiederholen ihn) an einem Beispiel.

Aufgabe: Bestimme den größten gemeinsamen Teiler von 228 und 174.

1. Dividiere die größere Zahl (228) durch die kleinere (174).
2. Erhalte ganzzahligen Quotienten und Rest.
3. Erhaltener Quotient: Wird neue größere Zahl.
4. Erhaltener Rest: Wird neue kleinere Zahl.
5. Wiederhole – bis zur Termination bei Rest 0.

$$228 = 1 \cdot 174 + 54$$
$$174 = 3 \cdot 54 + 12$$
$$54 = 4 \cdot 12 + 6$$
$$12 = 2 \cdot 6 + 0$$

Stimmt der Algorithmus?

Zu zeigen ist hier:

1. **Termination:** Der Algorithmus **endet** irgend wann mit einem Ergebnis.
2. **Korrektheit:** Das Ergebnis **ist Teiler** von 228 und 174.
   Das Ergebnis ist unter allen Teilern von 228 und 174 der **größte**.

Euklidischer Algorithmus: Termination

$$228 = 1 \cdot 174 + 54$$
$$174 = 3 \cdot 54 + 12$$
$$54 = 4 \cdot 12 + 6$$
$$12 = 2 \cdot 6 + 0$$

1. Wenn wir mit zwei gleichen Zahlen beginnen, dann ist diese Zahl der ggT.
2. Wir beginnen also mit zwei Zahlen, von denen eine die größere ist.
3. Der Rest ist immer echt kleiner als der Divisor.
4. Der Forschritts-Mechanismus macht also den Divisor immer kleiner.
5. Früher oder später muß das Terminationskriterium erreicht werden.

Euklidischer Algorithmus: Teiler-Eigenschaft

$$
\begin{aligned}
228 &= 1 \cdot 174 + 54 \\
174 &= 3 \cdot 54 + 12 \\
54 &= 4 \cdot 12 + 6 \\
12 &= 2 \cdot 6 + 0
\end{aligned}
$$

Wir lesen von unten nach oben:

1. 6 teilt 12.
2. 6 teilt 6 und 12, also auch 54.
3. 6 teilt 12 und 54, also auch 174.
4. 6 teilt 54 und 174, also auch 228.
5. 6 ist (ein) gemeinsamer Teiler von 228 und 174.

Euklidischer Algorithmus: Größte-Eigenschaft

$$228 = 1 \cdot 174 + 54$$
$$174 = 3 \cdot 54 + 12$$
$$54 = 4 \cdot 12 + 6$$
$$12 = 2 \cdot 6 + 0$$

Wir lesen von oben nach unten:

1. Sei X Teiler von 228 und 174.
2. Dann ist X auch Teiler von 54.
3. X ist Teiler von 174 und 54, also auch von 12.
4. X ist Teiler von 54 und 12, also auch von 6.
5. Also ist X kleiner oder gleich 6.
6. Damit ist 6 der größte unter allen Teilern von 228 und 174, weil für jeden Teiler X von 228 und 174 gilt, daß $6 \geq X$ ist.

Bezoutsche Identität: Aussage und Algorithmus

Lemma von Bezout

Der **größte gemeinsame Teiler** $g = \mathrm{ggT}(a, b)$ zweier ganzer Zahlen a, b läßt sich **als ganzzahlige Linearkombination** $g = \alpha \cdot a + \beta \cdot b$ dieser zwei Zahlen darstellen.

Algorithmus:

1. Die Quotienten und Reste im Euklidischen Algorithmus unterstreichen.
2. Mit der untersten nicht-trivialen Darstellung des ggT beginnen, dh. dort, wo nicht beide Koeffizienten der Linearkombination 0 sind.
3. Die Quotienten und Reste in dieser Darstellung unterstreichen.
4. Jeweils die kleinere der unterstrichenen Zahlen ersetzen durch einen Ausdruck der sich aus der obersten möglichen Euklidschen Gleichung ergibt.

Bezoutsche Identität: Beispiel

$$228 = 1 \cdot 174 + 54$$
$$174 = 3 \cdot 54 + 12$$
$$54 = 4 \cdot 12 + 6$$
$$12 = 2 \cdot 6 + 0$$

$$6 = 1 \cdot 54 - 4 \cdot 12$$
$$6 = -4 \cdot 174 + 13 \cdot 54$$
$$6 = 13 \cdot 228 - 17 \cdot 174$$

es ist: $12 = 1 \cdot 174 - 3 \cdot 54$

es ist: $54 = 1 \cdot 228 - 1 \cdot 174$

Fertig: Wir erhalten den ggT 6 als Linearkombination von 228 und 174:

$$6 = 13 \cdot 228 - 17 \cdot 174$$

Modulo-Division

Satz von der Modulo-Division

Seien zwei ganze Zahlen x und n teilerfremd. Dann existiert eine ganze Zahl y so, daß $x \cdot y \equiv 1$ **mod** n. Die Zahl y nennt man die **Inverse** zu x **modulo** n.

Beweis:
Seien x und n teilerfremd. Also ist **ggT** $(x, n) = 1$.

Nach dem Lemma von Bezout existieren α, β so, daß $1 = \alpha \cdot x + \beta \cdot n$.

Diese Gleichung modulo n betrachtet ergibt $1 \equiv \alpha \cdot x + 0$ **mod** n

Beispiel:	Bestimme die Inverse von 5 modulo 9.
Nach Bezout:	$1 = (-1) \cdot 9 + 2 \cdot 5$
Wir erhalten:	$1 \equiv 2 \cdot 5$ **mod** 9.
Antwort:	Die Inverse von 5 modulo 9 ist 2.

Eulersche Phi-Funktion

Eulersche Phi-Funktion

Die Eulersche Phi Funktion $\phi(n)$ einer natürlichen Zahl ist die **Anzahl** aller positiven ganzen Zahlen kleiner oder gleich n, **die keinen gemeinsamen Teiler** mit n haben.

Per Konvention: $\phi(1) = 1$.

Beispiele: $\phi(5) = 4$ denn teilerfremd sind: 1, 2, 3, 4.

Ist p Primzahl, dann ist $\phi(p) = p - 1$. Streiche in $\{1, 2, \ldots, p\}$ die Zahl p selber.

$\phi(2 \cdot 3) = 2$. Streiche aus $\{1, 2, 3, 4, 5, 6\}$ die durch 2 teilbaren Zahlen und die durch 3 teilbaren Zahlen, die echt kleiner als 6 sind (das sind jeweils disjunkte Mengen); streiche schließlich die Zahl 6.

Nach demselben Prinzip erkennt man für zwei *verschiedene* Primzahlen p und q

$$\phi(p \cdot q) = p \cdot q - \underbrace{(q-1)}_{\text{durch } p \text{ teilbar}} - \underbrace{(p-1)}_{\text{durch } q \text{ teilbar}} - \underbrace{1}_{\text{Zahl } p \cdot q} = (p-1) \cdot (q-1).$$

Theorem von Euler

Theorem von Euler

Seien x und n zwei teilerfremde natürliche Zahlen, dann ist $x^{\phi(n)} \equiv 1 \textbf{ mod } n$.

Sei $A = \{a_1, a_2, \ldots, a_{\phi(n)}\}$ die Menge aller positiven Zahlen kleiner oder gleich n, die mit n keinen gemeinsamen Teiler haben. Davon gibt es $\phi(n)$ Stück.

Multiplikation mit x ist eine bijektive Abbildung,
da durch x dividiert werden kann (es gibt eine Inverse!).

Die Menge $x \cdot A = \{x \cdot a_1, x \cdot a_2, \ldots, x \cdot a_{\phi(n)}\}$ ist also, modulo n betrachtet,
genau dieselbe Menge wie A.

Das Produkt aller Elemente in A und aller Element in $x \cdot A$ ist also, modulo n betrachtet,
dieselbe Zahl: Also ist $x \cdot a_1 \cdot x \cdot a_2 \ldots x \cdot a_{\phi(n)} = a_1 \cdot a_2 \ldots a_{\phi(n)}$

Wir können die mit n teilerfremden Zahlen a_j alle abdividieren und erhalten:
$x^{\phi(n)} \equiv 1 \textbf{ mod } n$.

2. RSA-Algorithmus

Ziel: Das RSA-Verfahren in seiner Textbuch-Variante (die so in der Praxis nicht genutzt wird) kennenlernen.

Historie und Namensgebung

Abb. 1: Ron **R**ivest © Rechtsnachweis siehe Anhang.

Abb. 2: Adi **S**hamir © Rechtsnachweis siehe Anhang.

Abb. 3: Leonard **A**dleman © Rechtsnachweis siehe Anhang.

Das RSA Verfahren ist das erste bekannte asymmetrische Verschlüsselungsverfahren. 2002 wurde an die drei Erfinder, **R**ivest, **S**hamir und **A**dleman der Turing Preis verliehen, der oft als "Nobelpreis der Informatik" bezeichnet wird.

RSA-Verfahren

Vorgehensweise:

1. Wähle zwei große Primzahlen p und q.
2. Berechne $n = p \cdot q$ und $\phi(n) = (p-1) \cdot (q-1)$.
3. Wähle eine Zahl e, die teilfremd ist zu $\phi(n)$.
4. Bestimme das Inverse d zu e modulo $\phi(n)$: $e \cdot d = 1 \bmod \phi(n)$.
5. Nutze die RSA-Operation $R_z(x) := x^z \bmod n$.

RSA-Haupteigenschaft

Es gilt: $R_d(R_e(x)) = R_e(R_d(x)) \equiv x \bmod n$.

Beweis:

$$(x^e)^d = x^{e \cdot d} = x^{1+k \cdot \phi(n)} = x \cdot (x^{\phi(n)})^k = x \cdot 1^k = x.$$

Sicherheit des RSA-Verfahrens: (1) Faktorisierungs-Problem

Definition: Faktorisierungs-Problem

Gegeben sei eine Zahl n. Finde die Primzahl-Zerlegung von n.

Man **vermutet**, daß das Faktorisierungs-Problem für ein n, das Produkt zweier großer Primzahlen ist, sehr schwer zu lösen ist.

Man **weiß**, daß die Aufgabe, zur Zahl e die Inverse d modulo n zu bestimmen, gleich schwer ist wie das Faktorisierungs-Problem.

Jeder Person sind zugeordnet: Ein öffentlicher Schlüssel (n, e), bei dem jeder weiß, daß er dieser Person zugeordnet ist. Ein privater Schlüssel d, den nur die Person sleber kennt und den sie geheim hält.

Sicherheit des RSA-Verfahrens: (2) RSA-Problem

Definition: RSA-Problem

Gegeben seien y, n und e. Finde ein x so, daß $y = x^e \bmod n$.

Beachte:

- Das RSA-Problem ist das Ziehen der e-ten Wurzel **mod** n: $\sqrt[e]{x^e} = x$.
- $\sqrt[e]{\ }$ modulo n ist verschieden von der bekannten $\sqrt[e]{\ }$ im Reellen!
- Man **vermutet**, daß das RSA-Problem für ein n, welches das Produkt zweier großer Primzahlen ist, sehr schwer zu lösen ist.
- Wer ein d kennt mit $d \cdot e \equiv 1 \bmod \phi(n)$ kann das Problem sehr schnell lösen: Nach der RSA-Haupteigenschaft ist die Lösung $y^d \bmod n$.
- Wer die Faktorisierung $n = p \cdot q$ kennt, der kennt auch $\phi(n)$ und kann sich daher zu dem e das d berechnen.

Sicherheit des RSA-Verfahrens: Praktische Bewertung

Kerckhoffsches Prinzip: Jeder weiß, wie das RSA Verfahren funktioniert.

Sicherheit hängt an der Schwierigkeit der Lösung zweier mathematischer Probleme:

1. Faktorisierungs-Problem
2. RSA-Problem

Wer für eines der Probleme einen effizienten Algorithmus findet
wird das voraussichtlich publizieren und wird dann berühmt.

Die Sicherheit des RSA-Verfahrens ist daher *auch* eine soziale,
weil man (derzeit) (noch nicht) die genaue Komplexität der Faktorisierung kennt.

Bei vielen anderen Verfahren beruht die Sicherheit auf

- Bekannter Komplexität des Problems Typisches Beispiel: NP-vollständige Probleme.
- Statistischer Sicherheit Typisches Beispiel: One Time Pad.

Erzeugung der Schlüssel: Mathematisches Problem

Problem 1: Erzeuge zwei große Primzahlen p und q.
Wie prüfe ich auf Primzahl, wenn das Faktorisierungsproblem kompliziert ist?
Lösung: Probabilistische Primzahl-Tests. Diese können effizient gestaltet werden, sind aber mit einem Zufallsfehler behaftet. Pro Runde verringert sich Fehlerwahrscheinlichkeit um einen festen Faktor. Nach einer gewissen Anzahl Runden ist die Wahrscheinlichkeit für die Anwendung klein genug.

Problem 2: Finde eine Zahl e, die zu $\phi(n)$ teilerfremd ist.
Wie teste ich auf Teilerfremdheit, wenn Faktorisierung schwierig ist?
Lösung: Zufallszahl erzeugen und mit Euklidischem Algorithmus den ggT zu $\phi(n)$ berechnen als Test der Teilerfremdheit.

Der Euklidische Algorithmus ist schnell!

Problem 3: Bestimme die Inverse d zu e.
Lösung: Euklidischen Algorithmus verwenden

Erzeugen der Schlüssel: Organisatorische Probleme

Variante 1: **Vertrauensperson** erzeugen lassen.

- **Problem:** Vertrauensperson kennt den privaten Schlüssel.

Variante 2: **Selber** Erzeugen.

- **Problem 1:** Muß Know-How und ggf. Programm dafür haben.
- **Problem 2:** Wie prüfe ich das Programm auf Korrektheit?
  Stammt es vom Angreifer und sendet ihm den privaten Schlüssel?
- **Problem 3:** Wählt es Schlüssel nicht beliebig sondern in kleinerer Menge,
  die ein Angreifer kennt und mit brute force durchprobieren kann?

Variante 3: In **Hardware** erzeugen lassen.

- **Vorteil:** Privater Schlüssel bleibt in Hardware, nicht auslesbar oder verlierbar.
- **Problem:** Muß der Hardware vertrauen.
  Wählt sie den Schlüssel beliebig? Problem wie oben!

Auswahl je nach individuellem Angreifer-Modell.

Nutzung als Verschlüsselung

Alice möchte Bob die Nachricht x verschlüsselt zusenden.

Alice weiß: Bob nutzt das RSA-Verfahren mit öffentlichem Schlüssel (n, e).

Alice bestimmt $R_e(x) = x^e$ **mod** n und sendet diese Zahl an Bob.

Bob empfängt von Alice die Zahl y und holt seinen privaten Schlüssel d aus dem Safe.

Bob berechnet $R_d(y) = y^d$. Das ist nach der RSA-Haupteigenschaft aber gerade x.

Kerneigenschaften der Verschlüsselung:

- **Jeder** kann an Bob **verschlüsseln**.
- Bob **kann entschlüsseln**, also x bestimmen, dank seiner Kenntnis von d.
- **Nur** Bob kann x bestimmen, da das RSA-Problem schwer zu lösen ist.

Nutzung als Unterschrift

Kerneigenschaften der Unterschrift:

- **Nur Bob kann** als Bob unterschreiben.
- **Jeder kann** die Unterschrift von Bob prüfen.

Bob möchte die Nachricht x unterschreiben.

Bob holt seinen privaten Schlüssel d aus dem Safe.

Bob berechnet $y := R_d(x) = x^d \textbf{ mod } n$ und veröffentlicht den Wert.

Jeder kann prüfen, daß Bob die Nachricht x unterschrieben hat.

Dazu besorgt man sich den öffentlichen Schlüssel (n, e) von Bob und prüft $x = R_e(y) \textbf{ mod } n$.

3. Facetten und Grenzen

3.1. Homomorphie und RSA-Praxis

3.2. Blinde Unterschrift

Ziel: Wir lernen verschiedene weitere Facetten des RSA-Verfahrens und mit ihnen auch die Grenzen der Textbuch-Variante kennen.

Was ist eine große ganze Zahl?

25195908475657893494027183240048398571429282126204032027777137836043662020707595556264018525880784406918290641249515082189298559149176184502808489120072844992687392807287776735971418347270261896375014971824691165077613379859095700097330459748808428401797429100642458691817195118746121515172654632282216869987549182422433637259085141865462043576798423387184774447920739934236584823824281198163815010674810451660377306056201619676256133844143603833904414952634432190114657544454178424020924616515723350778707749817125772467962926386356373289912154831438167899885040445364023527381951378636564391212010397122822120720357

Achtung: Wer diese Zahl in Primfaktoren zerlegen kann, gewinnt 200.000$ bei den RSA Laboratories.

Beweiskraft einer Unterschrift

Bob findet eine große ganze Zahl s.

Bob besorgt sich den public key (e, n) von Alice.

Bob stellt fest, daß s^e **mod** n die Zahl $x = 736772327673696666932667966$ ergibt.

x ist die Nachricht `ICH LIEBE BOB`. Ascii 73 ist das I. Ascii 67 ist das C usw.

Bob weiß nun: Alice hat ihren private key verwendet, um die Nachricht x digital zu unterschreiben.
(oder Alice hat ihren privaten Schlüssel verloren oder weitergegeben)
(oder Alice hat diesen privaten Schlüssel von seinem erzeuger erhalten).

Auf andere Weise kann die Nachricht s nicht entstanden sein.

s kann **nicht zufällig** erzeugt worden sein.

Niemand anderes kann den selben privaten Schlüssel wie Alice erzeugt haben.

Die Existenz einer digitalen Unterschrift ist ein extrem starkes Indiz, daß der Inhaber des privaten Schlüssels den Text unterschrieben hat.

Sicherheit und Unmöglichkeit

Sicherheit ("*ist*") und Unmöglichkeit ("*kann nicht*") in der Kryptographie ist meistens **probabilistisch** gemeint im Sinn von "extrem (un)wahrscheinlich".

Fortgeschrittenere Texte formulieren das mathematisch präziser, formal gelegentlich aber deutlich aufwendiger.

In einer Einführungsveranstaltung wollen wir darauf verzichten.

Was bedeutet extrem unwahrscheinlich?

Das **AES-256 Verfahren** arbeitet mit 256 Bit langen Schlüsseln.
Alice hat einen AES-256 Schlüsel gewählt.

Wie lange braucht Bob, um diesen Schlüssel mit Durchprobieren zu knacken?

- Wir haben $2^{256} \sim 10^{77}$ Schlüssel.
- Ein heutiger Rechner kann rund 10 Giga $= 10^{10}$ Schlüssel pro Sekunde durchprobieren.
- Wir haben auf der Welt rund 1 Giga $= 10^{9}$ Rechner.
- Die Erde kann rund 10^{19} Schlüssel pro Sekunde durchprobieren.
- Wir brauchen $10^{77-19} = 10^{58}$ Sekunden für den ganzen Schlüsselraum.
- Ein Jahr hat rund $3 \cdot 10^{7} \sim 10^{8}$ Sekunden.
- Wir brauchen also 10^{50} Jahre für den ganzen Schlüsselraum.

Zum Vergleich: Das Weltall ist nach Schätzung von Astronomen 10^{10} Jahre alt.

Sicherheitsparameter

AES-256 ist sicher, weil

1. Der Schlüsselraum 2^{256} Schlüssel enthält.
2. Keine Abkürzung bekannt ist (es muß **wirklich jeder** Schlüssel durchprobiert werden).

Der **Sicherheitsparameter** ist $\log_2$(Anzahl durchzuprobierender Schlüssel).
AES-256 hat den Sicherheitsparameter 256; aktuell gilt 128 noch als sicher.

Bei RSA ist längst nicht jede 256 Bit lange Bitkette ein Schlüssel,
denn bestimmte zahlentheoretische Voraussetzungen müssen gegeben sein.
Bei RSA gibt man daher die Länge des n in Binärpositionen an.

1024 bit RSA	gilt als unsicher
2048 bit RSA	**gilt als sicher**
4096 bis RSA	gilt als militärisch sicher

Tab. 1: Sicherheitseinschätzung von RSA Schlüssellängen im Jahre 2020. Die größte bisher geknackte RSA-Test-Instanz ist RSA-768, es sind aber auch kleinere Instanzen wie RSA-260 noch "offen". Siehe https://en.wikipedia.org/wiki/RSA_Factoring_Challenge.

Ein weiteres Problem

Bob findet die Zahl a^d.

Also hat die Besitzerin von d die Nachricht a unterschrieben.

Bob findet die Zahl b^d.

Also hat die Besitzerin von d die Nachricht b unterschrieben.

Bob bildet das Produkt p der beiden Zahlen: $p = a^d \cdot b^d$.

Bob stellt fest, daß $p = (a \cdot b)^d$.

Überrascht fragt sich Bob, ob die Besitzerin von d
auch die Nachricht $a \cdot b$ unterschrieben hat.

Vermutlich nicht!

Fazit: Houston, we have a problem!

Homomorphie-Eigenschaft

Es gilt:

$$R_d(a \cdot b) = R_d(a) \cdot R_d(b)$$

Genauer: Die RSA-Operation ist ein multiplikativer **Homomorphismus** auf dem Ring $(\mathbb{Z}\ \textbf{mod}\ n, +, \cdot)$.

Frage 1: Wie verhindert man Angriffe, die sich daraus ergeben?
Konkret: Hole Unterschrift auf Dokument a und b ein, die beide harmlose Verträge sind. Bekomme dadurch Unterschrift auf $a \cdot b$, das eine große Schenkung darstellt.

Frage 2: Kann diese Eigenschaft anderweitig genutzt werden?

Schutz vor Homomorphie-Eigenschaft

Ansatz 1: Verändern.
Bevor man einen Text a unterschreibt, ändert man diesen ein wenig ab in a'.
Das zerstört einen möglichen, mittels a vorbereiteten Angriff,
da es die vorbereitete Eigenschaft $a \cdot b = c$ zerstört.

Ansatz 2: Randomisieren.
Bevor man einen Text a unterschreibt, diesen links und rechts mit Zufallszahlen erweitern.
Das ist eine Erweiterung von Ansatz 1.

Ansatz 3: Hash Funktion
Man unterschreibt ohnehin nie ein Dokument sondern eine Hash-Funktion des Dokuments.
Denn: Dokumente sind meistens länger als der Modul n.
Die Hash-Funktion wirkt ähnlich wie die Ansätze 1 und 2.

Textbook-RSA

Gegen RSA gibt es eine Vielzahl sehr effizienter Angriffe.

Gegen die Implementierungen:

1. Timing Attacke
2. Power Attack
3. Differential Power Attack
4. Accustic Emanation Attacke
5. Belcore Attacke
6. Randomization Attack

Gegen das Konzept:

1. Wiener Attacke
2. Coppersmith Attacke
3. ROCA Attacke
4. Chosen Plaintext Attac
5. Chosen Ciphertext Attac
6. Adaptive Chosen Ciphertext (Bleichenbacher) Attack
7. Non strong prime number attack

Daher wird RSA ohnehin nie in der Textbuch-Variante eingesetzt.

Seitenkanal-Angriffe

Seitenkanal-Angriffe

Seitenkanal-Angriffe nutzen Informationen, welche aus der konkreten Implementierung eines Verfahrens zugänglich werden und zur Kompromittierung von Schutzzielen genutzt werden können.

- Zwischenbuffern der Stromversorgung: Hält meßbaren Verbrauch konstant.
- Eingeschränkter Zugang zu Verschlüsselungs-Hardware: Verhindert Messung physikalischer Parameter.
- Eingeschränkter Zugang zu Verschlüsselungs-Hardware: Reduziert Testmessungen durch Angreifer.

Power Attacke als Beispiel für Seitenkanal-Angriff

Beobachtung: Anzahl der 1-en auf dem Bus beeinflußt den Stromverbrauch der CPU.

Power Attacke auf RSA-Hardware:

- Messe den Stromverbrauch des Systems während der Verarbeitung.
- Erfasse Ein- und Ausgaben des Systems.
- Bestimme daraus den privaten Schlüssel.

Spezifische Verteidigung:

- Zwischenbuffern der Stromversorgung: Hält meßbaren Verbrauch konstant.

Praktische Nutzung von RSA

Leitsatz

In der Kryptographie sollen stets nur professionelle Implementierungen von ausgewiesenen Experten verwendet werden.

Konkrete Aspekte bei RSA

- Seitenkanal-Sicherheit — Durch geeignete Implementierung.
- Randomisierung nutzen.
- Padding nutzen.
- Signatur: Hash signieren statt Dokument
- Verschlüsselung: Hybride Verschlüsselung
  RSA für den Austausch eines symmetrischen Schlüssel.

Blinde Unterschrift

Alice hat ein Dokument x und braucht darauf eine digitale Unterschrift s von Bob.

Bob soll das Dokument x nicht mit Alice in Verbindung bringen können.

Anwendungsbeispiel: Elektronisches Geld.

Parteien: Alice, Bank und Händler.

Blinde Signatur nach David Chaum

Situation: Bob hat public key (n, e) und private key d.

Aufgabe: Alice will blinde Unterschrift von Bob auf Dokument x.

Vorgehensweise:

1. Alice wählt Blinding Factor B teilerfremd zu n.
2. Alice sendet $y = x \cdot R_e(B)$ **mod** n an Bob.
3. Bob unterschreibt, sendet also $z = R_d(y)$ an Alice.
4. Alice berechnet multiplikative Inverse G zu B.
5. Alice berechnet $z \cdot G$.

Korrektheit der Blinden Signatur

Signatur: Alice erhält damit eine Signatur von Bob auf x.

$$
\begin{aligned}
& z \cdot G && \text{Definition von } z \\
& \equiv y^d \cdot G && \text{Definition von } y \\
& \equiv (x \cdot R_e(B))^d \cdot G = && \text{Definition von } R_e \\
& \equiv (x \cdot B^e)^d \cdot G = && \text{Eigenschaft des Potenzierens} \\
& \equiv x^d \cdot B^{e \cdot d} \cdot G = && \text{RSA-Haupteigenschaft} \\
& \equiv x^d \cdot B \cdot G = && \text{Definition von G} \\
& x^d \ \textbf{mod}\ n
\end{aligned}
$$

Blindheit: Bob kann sich y merken, aber daraus nicht auf x schließen.
Denn: Multiplikation mit $R_e(B)$ erlaubt Alice eine *beliebige* Transformation von x innerhalb der Menge aller zu n teilerfremden Werte.

Digitales Geld

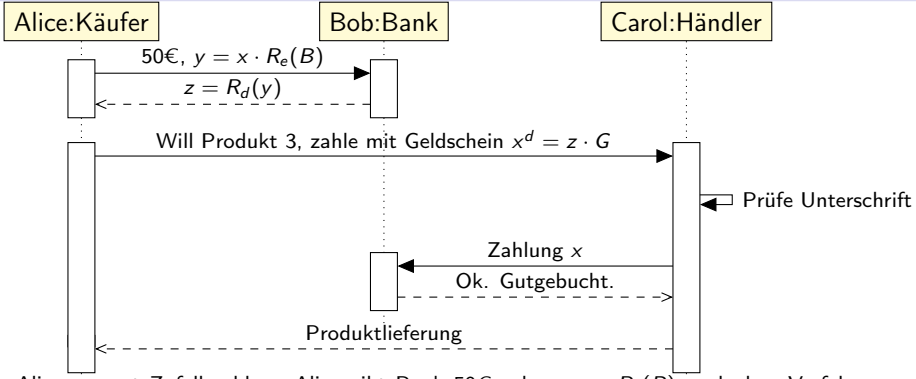

Abb. 4: Alice erzeugt Zufallszahl x. Alice gibt Bank 50€und $y = x \cdot R_e(B)$ nach dem Verfahren von Chaum. Alice erhält von Bank Unterschrift z auf y. Alice berechnet $z \cdot G = x^d$. Alice sendet x^d an die Carol. Carol prüft Unterschrift der Bank. Wenn ok reicht Carol x^d bei Bank zur Gutschrift ein. Bank kann prüfen, ob x bzw. x^d schon einmal zur Gutschrift eingereicht wurde und verhindert so Doppelnutzungvon x durch Alice. Bank kann nicht erkennen, daß die Zufallszahl von Alice stammt. Hier nicht erklärte Protokoll-Elemente müssen noch sicherstellen, daß Carol die Münze nicht für sich selber nutzt und dann gegenüber Alice behauptet, sie hätte sie (Alice) hätte sie bereits für sich selber genutzt.

Anhang

Übersicht

Verzeichnis aller Abbildungen

Verzeichnis aller Tabellen

Rechtsnachweise

Abb. 1 Quelle: https://upload.wikimedia.org/wikipedia/commons/7/79/Ronald_L_Rivest_photo.jpg, Ronald L. Rivest, Nutzung nach CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0.

Abb. 2 Quelle: https://upload.wikimedia.org/wikipedia/commons/9/9a/Adi_Shamir_Royal_Society.jpg, The Royal Society, Nutzung nach CC BY-SA 3.0 https://creativecommons.org/licenses/by-sa/3.0.

Abb. 3 Quelle: https://upload.wikimedia.org/wikipedia/commons/a/af/Len-mankin-pic.jpg, len adlmen, Nutzung nach CC BY-SA 3.0 https://creativecommons.org/licenses/by-sa/3.0.

Rechtliche Hinweise

Die hier angebotenen Inhalte unterliegen deutschem Urheberrecht. Inhalte Dritter werden unter Nennung der Rechtsgrundlage ihrer Nutzung und der geltenden Lizenzbestimmungen hier angeführt. Auf das Literaturverzeichnis wird verwiesen. Das Zitatrecht in dem für wissenschaftliche Werke üblichen Ausmaß wird beansprucht. Wenn Sie eine Urheberrechtsverletzung erkennen, so bitten wir um Hinweis an den auf der Titelseite genannten Autor und werden entsprechende Inhalte sofort entfernen oder fehlende Rechtsnennungen nachholen. Bei Produkt- und Firmennamen können Markenrechte Dritter bestehen. Verweise und Verlinkungen wurden zum Zeitpunkt des Setzens der Verweise überprüft; sie dienen der Information des Lesers. Der Autor macht sich die Inhalte, auch in der Form, wie sie zum Zeitpunkt des Setzens des Verweises vorlagen, nicht zu eigen und kann diese nicht laufend auf Veränderungen überprüfen.

Die Bereitstellung der Inhalte erfolgt hier zur persönlichen Information des Lesers. Eine Haftung für mittelbare oder unmittelbare Schäden wird im maximal rechtlich zulässigen Ausmaß ausgeschlossen, mit Ausnahme von Vorsatz und grober Fahrlässigkeit. Eine Garantie für den Fortbestand dieses Informationsangebots wird nicht gegeben.

Die Anfertigung einer persönlichen Sicherungskopie für die private, nicht gewerbliche und nicht öffentliche Nutzung ist zulässig, sofern sie nicht von einer offensichtlich rechtswidrig hergestellten oder zugänglich gemachten Vorlage stammt.

Zitierweise dieses Dokuments

Wenn Sie Inhalte aus diesem Werk nutzen oder darauf verweisen wollen,
zitieren Sie es bitte wie folgt:

Clemens H. Cap: Das RSA Verfahren. Electronic document. https://iuk.one/1010-1011 15. 12. 2020.

Bibtex Information: https://iuk.one/1010-1011.bib

```
@misc{doc:1010-1011,
   author       = {Clemens H. Cap},
   title        = {Das RSA Verfahren},
   year         = {2020},
   month        = {12},
   howpublished = {Electronic document},
   url          = {https://iuk.one/1010-1011}
}
```

Typographic Information:
Typeset on December 15, 2020
This is pdfTeX, Version 3.14159265-2.6-1.40.21 (TeX Live 2020) kpathsea version 6.3.2
This is pgf in version 3.1.5b
This is preamble-slides.tex myFormat©C.H.Cap

Index

Verzeichnis aller Folien

Legende:
Fortsetzungsseite
Seite ohne Überschrift
Bildseite