Proof of Concept für den Angriff eines IT-Systems durch Implementierung kleptographischer Schwachstellen in kryptographischen Bibliotheken

Studienarbeit

für die Prüfung zum

Bachelor of Science

des Studienganges Informatik / Informationstechnik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

Yannic Hemmer

Abgabedatum 16. Mai 2022

Bearbeitungszeitraum	24 Wochen
Matrikelnummer	6853472
Kurs	TINF19B2
Ausbildungsfirma	SySS GmbH Tübingen
Betreuer der Ausbildungsfirma	-
Gutachter der Studienakademie	Rolf Felder

Erklärung

Ich versichere hiermit, dass ich meine Studienarbeitmit dem Thema: »Proof of Concept für den Angriff eines IT-Systems durch Implementierung kleptographischer Schwachstellen in kryptographischen Bibliotheken« selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt. ____________________

____

Ort Datum Unterschrift

Sofern vom Dualen Partner ein Sperrvermerk gewünscht wird, ist folgende Formulierung zu verwenden:

Sperrvermerk

Der Inhalt dieser Arbeit darf weder als Ganzes noch in Auszügen Personen außerhalb des Prüfungsprozesses und des Evaluationsverfahrens zugänglich gemacht werden, sofern keine anderslautende Genehmigung vom Dualen Partner vorliegt.

Zusammenfassung

Dieses LaTeX-Dokument kann als Vorlage für einen Praxis- oder Projektbericht, eine Studien- oder Bachelorarbeit dienen.

Zusammengestellt von Prof. Dr. Jürgen Vollmer <juergen.vollmer@dhbw-karlsruhe.de> `https://www.karlsruhe.dhbw.de`. Die jeweils aktuellste Version dieses LaTeX-Paketes ist immer auf der *FAQ-Seite* des Studiengangs Informatik zu finden: `https://www.karlsruhe.dhbw.de/inf/studienverlauf-organisatorisches.html` → *Formulare und Vorlagen*.

Stand `$Date: 2020/03/13 15:07:45 $`

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Liste der Algorithmen

Formelverzeichnis

Abkürzungsverzeichnis

Kapitel 1

Grundlagen

In diesem Kapitel werden die theoretischen Grundlagen der Kryptologie, Mathematik, Komplexitätstheorie und der IT-Sicherheit erläutert, die in dieser Arbeit eine Rolle spielen. Aus diesen sollen die grundlegenden Funktionen eines kleptographischen Angriffes und dessen Folgen abgeleitet werden.

1.1 Kryptologie

Die Kryptologie ist die wissenschaftliche Disziplin für den Schutz von Daten. Unter ihr stehen die zwei Felder der Kryptographie und der Kryptoanalyse.

1.1.1 Kryptographie

Die Kryptologie befasst sich mit der Entwicklung von Verfahren und Techniken für den sicheren Austausch von Daten. Dabei stehen zwei Eigenschaften im Fokus:

Eigenschaften

Geheimhaltung Durch Geheimhaltung sollen, bei der Übertragung von Daten zwischen Teilnehmern, Unbeteiligte keine Erkenntnisse über den Inhalt erlangen. Dies kann durch physikalische oder organisatorische Maßnahmen erreicht werden, wobei Unbeteiligten der Zugang zu den übertragenen Daten verwehrt wird. Diese Maßnahmen sind sinnvoll bei der Übergabe der Daten in einer nicht digitalen Welt. Bei der Kommunikation in digitalen Netzen, wie u.a. dem Internet, sind diese Maßnahmen nur schwer zu implementieren. Dies gilt nicht für kryptographische Maßnahmen. Dabei ist es nicht mehr das Ziel, Unbeteiligten den Zugang zu den übertragenen Daten zu erschweren, sondern den Inhalt der Daten während der Übertragung zu verschlüsseln. Dadurch soll es Unbeteiligten nahezu unmöglich sein, aus den mitgehörten oder abgefangenen Daten, Rückschlüsse auf deren Inhalt zu erlangen.[1]

Authentifikation Durch Authentifikation soll es den Teilnehmern einer Kommunikation möglich sein, die anderen Teilnehmer und empfangene Nachrichten zweifelsfrei identifizieren und zuweisen zu können. Hierbei spielen Signaturverfahren eine wichtige Rolle, da kein Geheimnis

[1] Beutelspacher, Schwenk und Wolfenstetter 2015, S. 1.

benötigt wird um einen Teilnehmer zu authentifizieren. Dabei können sich Teilnehmer durch das Wissen oder den Besitz eines Geheimnisses (Passwort, Zertifikat, Schlüssel) authentifizieren.[2]

Nur wenn beide Eigenschaften gegeben sind ist eine Übertragung von Daten als sicher anzusehen. Falls die Geheimhaltung fehlt, kann der Inhalt durch Sniffing mitgelesen werden. Falls die Authentifikation der Teilnehmer fehlt, können sich Unbeteiligte als ëchte"Teilnehmer ausgeben und somit die Daten an ihrem Endpunkt entschlüsseln.

Zusätzliche Eigenschaften

Perfect Forward Security Perfect Forward Security

Kryptographische Verfahren

Kryptographische Verfahren sind Algorithmen, welche die Genheimhaltung von Daten und die Authentifikation von Teilnehmers und Nachrichten sicherstellt. Dadurch kann man sie in Verschlüsselungsverfahren und Authentifikationsverfahren unterscheiden. Dabei können Verfahren, wie z.B. Rivest-Shamir-Adleman (RSA) beiden Aufgaben übernehmen.

Asymmetrische Verschlüsselung Bei asymmetrischen Verschlüsselungsverfahren wird statt dem gleichen Schlüssel für das Ver- und Entschlüsseln, zwei verschiedene Schlüssel verwendet. Dabei hat jeder Teilnehmer ein öffentlichen Schlüssel e und einen privaten und geheimen Schlüssel d. Hierbei ist es vorgesehen, dass möglichst alle potenziellen Teilnehmer den Schlüssel e kennen. Wenn eine Nachricht mit einem der beiden Schlüssel chiffriert wurde, kann nur mittels dem anderen Schlüssel dechiffriert werden. Somit können Nachrichten an einen Teilnehmer verschlüsselt versandt werden, indem diese mit dem öffentlichen Schlüssel e des Teilnehmers chiffriert wird. Nun kann nur der Teilnehmer mit dem zugehörigen privaten Schlüssel d, die Nachricht verschlüsseln. Zusätzlich kann auch die Authentifikation von Teilnehmer und die Authentizität von Nachrichten mit hoher Sicherheit festgestellt werden. Somit kann der Author einer Nachricht, einen Fingerabdruck dieser Nachricht mit seinem privaten Schlüssel d signieren, an die Nachricht anhängen und dann beide Teile verschlüsseln. Diese Signatur kann verifiziert werden, indem der Empfänger die Nachricht entschlüsselt und dann die Signatur verifiziert, indem er den öffentlichen Schlüssel des Authors e auf diesen anwendet. Danach vergleicht er den empfangenen Fingerabdruck mit einem eigens erstellten Fingerabdruck. Somit kann die Geheimhaltung, Authentizität und Integrität der Nachricht bestimmt werden.

Die zwei Schlüssel e und d eines Teilnehmers, werden auch als Schlüsselpaar bezeichnet. Ein solches verfahren, wird asymmetrisch genannt, da für das Ent- und Verschlüsseln zwei unterschiedliche Informationen vorliegen müssen. Diese Informationen sind auch nicht auseinander ableitbar, wie es z.B. bei multiplikativen Chiffren der Fall wäre. Die Funktionalität des Verfahrens, beruht auf der Annahme, dass alle Teilnehmer Zugang zu den öffentlichen Schlüssel jedes anderen Teilnehmers haben bzw. haben können. Durch diese Charakteristika werden solche Verfahren auch als Public-Key-Kryptographie bezeichnet.

Dabei wird stets die Annahme getroffen, dass der private Schlüssel eines Teilnehmers ausschließlich diesem vorliegt. Anderenfalls ist die Geheimhaltung und die Authentifikation beim Informationsaustausch von und mit diesem Teilnehmer nicht mehr gewährleistet. Somit wäre die Sicherheit kompromittiert.

Die Schlüssel eines Schlüsselpaar bilden somit Umkehrfunktionen zueinander.

[2] Beutelspacher, Schwenk und Wolfenstetter 2015, S. 2.

Hybride Verfahren Asymmetrische Verschlüsselungsverfahren haben häufig den Nachteil, dass die deutlich rechenaufwändiges sind, wie wir später bei RSA sehen werden. Es liegen zwar effiziente Verfahren vor, um z.B. die modularen Potenz aus zwei 300-stelligen Zahlen und einem Modulo zu bilden 1.2.4. Dennoch sind diese Verfahren mit mehr Aufwand verbunden, als z.B. symmetrische Blockchiffren wie Advanced Encryption Standard (AES).

Deshalb werden asymmetrische Verfahren für die Initialisierung der Kommunikation verwendet. In dieser Initialisierungsphase soll der Teilnehmer authentifiziert werden und ein gemeinsamer, geheimer, symmetrischer Schlüssel vereinbart werden.

In der darauffolgenden Kommunikationsphase werden die Nachrichten durch symmetrische Chiffren mittels des vereinbarten Schlüssels, effizient verschlüsselt und auf der Gegenseite entschlüsselt. Asymmetrisch Chiffren werden hier benutzt um Fingerabdrücke von Nachrichten zu signieren und zu verifizieren, wie oben 1.1.1 gezeigt. Zusätzlich werden durch asymmetrische Verfahren regelmäßig neue symmetrische Schlüssel vereinbart.

Solche hybriden Verfahren sind z.B. beim Browsen im Internet zu finden: $TLS\_ECDHE\_RSA\_WITH\_A$

1.1.2 Kryptoanalyse

Moderne kryptographische Verfahren, werden nach ihrer Sicherheit und ihrer Effizienz beurteilt. Dabei kann die Sicherheit eines Verfahrens auf mathematische Probleme gestützt werden, welche aktuell und in der nahen Zukunft nicht trivial lösbar sind.

Alle Angriffe auf kryptographische Verfahren, gelten dem Erlangen des Geheimtextes einer chiffrierten Nachricht oder dem Berechnen, des verwendeten Geheimnisses.

Bei Angriffen auf das Geheimnis werden hier lediglich computergestützte Angriffe betrachtet, also nur Angriffe, die durch den Einsatz von Rechnerressourcen und Algorithmen versuchen, den geheimen Schlüssel zu bestimmen. Es wird im Allgemeinen davon ausgegangen, das Nutzergeheimnisse, wie Passwörter, Zertifikate und private Schlüssel nicht öffentlich zugänglich sind. Brute-Force Angriffe sind beispielhaft für die Angriffe. Hierbei wird systematisch der Zahlenraum (bzw. Zeichenraum) aller möglicher Schlüsselkombinationen durchprobiert. Weiterentwicklungen dieses Angriffs versuchen auf Grundlage von statistischen Erkenntnissen den Zahlenraum des Geheimnisses einzugrenzen, wie z.B. Wörterbuchangriffe. Hierbei ist die Länge und die Zufälligkeit des Geheimnisses der entscheidende Faktor für einen effektiven Schutz vor Angriffen.

Um die Sicherheit von kryptographischen Verfahren beurteilen zu können, werden erfolgreiche Angriffe auf diese Verfahren, nach den hierfür notwendigen Voraussetzungen, unterteilt.[3]

Known Cipher Attack Hierbei benötigt der Angreifer beliebige Menge an verschlüsselten Text, um aus diesem den Schlüssel und somit den Geheimtext ableiten zu können.

Known Plaintext Attack Bei diesen Angriffen, kennt der Angreifer eine echte Teilmenge der verschlüsselten Textes und den dazugehörigen Geheimtext. Diese Angriffe sind erfolgreich, wenn sich aus einer echten Teilmenge des Klartextes und dem dazugehörigen Geheimtext, der verwendete Schlüssel berechnen lässt.

Chosen Plaintext Attack Bei Chosen Plaintext Attack kann der Angreifer das Chiffrat zu einem von ihm gewählten Klartext berechnen. Dies ist ein klassischer Fall bei Public-Key-Kryptographie 1.1.1, da hier der Algorithmus (Prinzip von Kerckhoff) und die Schlüssel öffentlich

[3] Beutelspacher 2015, S. 20.

sind. Somit kann sich eine Angreifer zu jedem beliebigen Klartext das entsprechende Chiffrat berechnen. Angriffe haben diese Eigenschaft, wenn sich dadurch das Geheimnis (bei Public-Key-Kryptographie der private Schlüssel) berechnen lässt.

Zusätzlich ist noch eine weitere Kategorie verwendbar:

Chosen Chipher Attack Hierbei kann der Angreifer jeglichen Geheimtext entschlüsseln. Zusätzlich liegen ihm eine beliebige Menge an abgefangenen Geheimtexten zur Verfügung. Dabei ist die natürlich die Vertraulichkeit bereits versandter Nachrichten kompromittiert. Jedoch ist hierbei das Ziel des Angriffs, das verwendete Geheimnis zu berechnen.

Nur wenn für kryptographische Verfahren keiner der aufgeführten Stufen an Voraussetzungen ausreicht, um das verwendete Geheimnis zu berechnen sind diese als sicher zu betrachten. In dieser Arbeit wird mit kryptographischen Verfahren gearbeitet, die als sicher betrachtet werden können.

1.1.3 Prinzipien

In der Kryptologie gelten verschiedene Prinzipien. Diese sind zwar in der theoretischen Betrachtung nicht notwendig, haben aber in der realen Welt eine große Bedeutung.

Prefect Forward Security

Perfect Forward Secrecy (PFS) ist ein Prinzip für kryptographische Verfahren, dass durch zukünftige Veröffentlichung des Geheimnisses, die Vertraulichkeit von in der Vergangenheit versandten Nachrichten nicht gefährdet ist. Dies wird garantiert dadurch, dass langlebige Geheimnisse (bzgl. der Speicherung und Nutzung) zusammen mit temporären Geheimnissen zur Verschlüsselung genutzt werden. Somit kann ein Angreifer, der alte Geheimtexte gesammelt hat und im Besitz des langlebigen Geheimnisses ist, die gespeicherten Geheimtexte nicht entschlüssel. Dies kann auch z.B. durch rotierende Geheimnisse, wie Sitzungsschlüssel erreicht werden.

Prinzip von Kerckhoff

Das Prinzip von Kerckhoff besagt, dass die Sicherheit eines kryptographischen Verfahrens nicht auf der Geheimhaltung des Algorithmus beruhen darf. Dabei soll die Sicherheit alleinig auf dem verwendeten Geheimnisses und seiner Geheimhaltung beruhen. Natürlich sind Verfahren denkbar, die gegen Kerckhoff's Prinzip verstoßen denkbar, aber auf Grundlage der geschichtlicher Erkenntnisse zu vermeiden.[4]

1.2 Mathematik

1.2.1 Primzahlen

Primzahlen werden in mehreren asymmetrischen Verfahren genutzt. Im Rahen dieser Arbeit wird die Menge aller Primzahlen als $\mathbb{P}$. definiert. Wenn eine Zahl prim ist, ist diese Element von $\mathbb{P}$.

Mathematische Probleme stellen die Grundlage für moderne Kryptographie.

[4] Beutelspacher 2015, S. 19.

1.2.2 Diskreter Logarithmus

Bei der Bestimmung des Logarithmus wird der Exponent (hier: x) gesucht, welcher mit einer bekannten Zahl als Basis z, eine weitere bekannte Zahl y ergibt.

$$z^x = y \tag{1.1}$$

Der diskrete Logarithmus bezieht hier auf die Berechnung des Logarithmus in ein Gruppe. Diese Gruppe bildet sich aus der Rechnung mit Restklassen (modulo). Dadurch entsteht folgendes Problem, bei der die Variable x gesucht ist und alle anderen Variablen bekannt sind.

$$z^x \pmod{n} \equiv y \tag{1.2}$$

Hierbei ist in der Notation zu beachten, dass sich durch das Rechnen auf mit einer Gruppe, Äquivalenzklassen ($\equiv$) bilden. Diese entsprechen den Restklassen des Rechnen mit Modulo. n ist die Mächtigkeit der Aquivalenzklassen.

Die Bestimmung von x in 1.2 wird als Problem des diskreten Logarithmus bezeichnet. Mit der Komplexität wird sich in den Grundlagen der Komplexitätstheorie beschäftigt.

Dabei ist die Umkehrfunktion, des diskreten Logarithmus $f(x)$ 1.2, mathematisch einfach zu berechnen. Diese Umkehrfunktion entspricht der diskreten Exponentialfunktion:

$$f^{-1}(x) = z^x \pmod{n} \equiv y \tag{1.3}$$

Hierbei sind z, x, n gegeben und y gesucht.

1.2.3 Faktorisierung

Bei der Faktorisierung wird versucht eine Zahl in Faktoren zu zerlegen. Dabei handelt es sich, im Kontext der Kryptographie, meist um die Faktorisierung des Produkts zweier großer Primzahlen. Dadurch bildet sich folgende Formel, wobei p und q Primzahlen sind (also Element der Menge der Primzahlen $\mathbb{P}$) und n das resultierende Produkt:

$$n = p * q \mid p, q \in \mathbb{P} \tag{1.4}$$

Da n das Produkt zweier Primzahlen ist, sind seine einzigen Teiler: n selbst, 1 und die seine Primfaktoren p und q. Deshalb handelt es sich hierbei auch um eine Primfaktorzerlegung von n.

Dabei ist die Primfaktorzerlegung von n ein rechenaufwändiges Problem, falls p udn q große Zahlen sind. Im Gegensatz dazu ist die Berechnung von bzw. die Validierung mit n sehr einfach, da hierfür nur die Multiplikation von p und q notwendig ist. Somit liegt die gleiche Situation, wie beim Problem des diskreten Logarithmus 1.2.2 vor: Ein rechenaufwändiges Problem, dessen Umkehrfunktion sehr einfach ist[5].

1.2.4 Effiziente Berechnung der diskreten Exponentialfunktion

In der Kryptographie werden große Zahlen genutzt, um die Sicherheit der verwendeten Algorithmen zu gewährleisten. Hierfür wird als Beispiel angenommen, dass als Basis z eine 256-bit Lange Zahl hoch einem 300-bit langem Exponenten x genommen werden soll. Hierbei ist n 1024-bit lang.

[5] Beutelspacher, Schwenk und Wolfenstetter 2015, S. 179.

Wenn mann nun z in Byte berechnet wäre dies eine 32 Byte lange Zahl.

x entspricht einer ungefähr 90. stelligen Zahl.

$$z^{10^{90}} \pmod n \equiv y \tag{1.5}$$

Eine numerische Berechnung von $z^{10^{90}}$ ist aufgrund von begrenzten Ressourcen nicht möglich.

Jedoch kann man sich die diskrete Eigenschaft dieser Problems sich zu nutzte machen. Hierfür können Verfahren, wie Square-and-Multiply zusammen mit der Restklassenberechnung genutzt werden. Dadurch lassen sich auch großzahlige Exponenten berechnen. Hierfür soll ein einfaches Beispiel gegeben werden:

$$37^{52} \pmod{128} \equiv y \tag{1.6}$$

Bei Betrachtung der Äquivalenzgleichung fällt auf, dass 37^{52} eine große Zahl ergibt. Jedoch wird diese Zahl noch $x \pmod 1 28$ gerechnet. Dadurch liegt das Ergebnis in einem Zahlenraum von:

$$y \in \mathbb{N} \mid 0 \leq y < 128 \tag{1.7}$$

Auf Grundlage der Potenzgesetze wird 37^{52} nun zerlegt.

$$\begin{aligned} 52 = 32 + 16 + 4 = 2^5 + 2^4 + 2^2 \\ 37^{52} \pmod{128} \equiv 37^{2^5} * 37^{2^4} * 37^{2^2} \\ \equiv 37^{2^5} \pmod{128} * 37^{2^4} \pmod{128} * 37^{2^2} \pmod{128} \end{aligned} \tag{1.8}$$

Die einzelnen Bestandteile werden dann iterativ berechnet und durch Multiplikation zusammengefasst (siehe 1.8). Dies wird als Square-and-Multiply-Verfahren bezeichnet.

$$\begin{aligned} 37^{2^2} \pmod{128} &\equiv (37^{2^1} \pmod{128})^2 \\ 37^{2^3} \pmod{128} &\equiv (37^{2^2} \pmod{128})^2 \\ 37^{2^4} \pmod{128} &\equiv (37^{2^3} \pmod{128})^2 \\ 37^{2^5} \pmod{128} &\equiv (37^{2^4} \pmod{128})^2 \end{aligned} \tag{1.9}$$

Allgemein

Gegeben mit gesucht y:

$$z^x \pmod n \equiv y \tag{1.10}$$

Zerlegung von x eine Summe von Zweierpotenzen:

$$x = 2^0 + 2^1 + 2^2 + ... \tag{1.11}$$

Dabei bilden die binären Logarithmen der einzelnen Zweierpotenzen die Menge $\mathbb{K}$.

Berechnung der einzelnen Faktoren durch iteratives Square-and-Multiply-Verfahren. Dies wird bis $f(max(\mathbb{K}))$ berechnet. $max(\mathbb{K})$ steht hier für das Element von $\mathbb{K}$, mit dem größten Wert.

$$f(i+1) = f(i)^2 \pmod n \mid f(1) = z^1 \pmod n \tag{1.12}$$

Zuletzt wird das Produkt, aller Ergebnisse von $f(x)$ für die Elemente der Menge $\mathbb{K}$, gebildet. Dabei gilt:

$$\prod_{k \in \mathbb{K}} f(k) \equiv z^x \pmod n \equiv y \tag{1.13}$$

1.3 Komplexitätstheorie

Die Komplexitätstheorie befasst sich mit der Komplexität von Problemen, welche durch Algorithmen gelöst werden. Dabei wird der Speicherbedarf und der Zeitaufwand eines Algorithmus. Schrankenfunktionen werden gebildet durch die Betrachtung des Speicherbedarf und des Zeitaufwands im Bezug auf die Länge der Eingabeparameter. Da hier eine reine kryptographische Betrachtung der Schrankenfunktionen stattfinden soll, wird hier nur in zwei verallgemeinerte Schrankenfunktionen[6] unterschieden:

- Polynomiale Komplexität
- Nichtpolynomiale Komplexität

Polynomiale Komplexität umfasst hier alle Probleme, die algorithmisch mit polynomialem Aufwand (Zeit/Speicher) gelöst werden können. D.h. bei steigender Eingabelänge n steigt der Aufwand im schlimmsten Fall mit $\mathcal{O}(n^c \mid \text{c is constant})$. Diese Probleme gehören damit zur Komplexitätsklasse **P**. Diese umfasst alle Probleme, welche algorithmisch mit maximal polynomialem Aufwand gelöst werden können. Diese Probleme können meistens von modernen Computern gelöst werden.

Nichtpolynomiale Komplexität hingegen umfasst alle Probleme, die mehr als polynomialen Aufwand im Worst-Case brauchen. Dies können Probleme sein, die algorithmisch nur mit exponentiellen $\mathcal{O}(d^n \mid d > 1)$ oder faktoriellen $\mathcal{O}(n!)$ Aufwand[7] gelöst werden können. Diese Probleme werden der Komplexitätsklasse **NP** zugewiesen. Dies sind Probleme können nicht von deterministischen Computern in mit polynomialen Aufwand gelöst werden.

Bezug zur Kryptographie Dadurch sind sie für die Kryptographie besonders interessant, da man die Sicherheit eines Systems auf ein NP-vollständiges Problem stützen kann. Somit ist die theoretische Sicherheit des Systems nicht brechbar. Jedoch sollte darauf geachtet werden, dass die vorgesehenen Teilnehmer an einem Datenaustausch nicht auch das NP-vollständige problem lösen müssen. Ihr Aufwand soll so gering wie möglich gehalten werden, wobei der Aufwand für einen Angreifer exponentiell oder faktoriell zur Sicherheit der Systems (z.B. die Länge des Schlüssels) ist.

Beispiele für solche Probleme sind der diskrete Logarithmus 1.2.2 und die Faktorisierung 1.2.3 eines Produkt von Primzahlen[8]. Weitere Beispiele wäre die Berechnung des Isomorphismus zweier Graphen, das Berechnen von Modularen Quadratwurzeln oder die Multiplikation auf elliptischen Kurven.

[6] Beutelspacher, Schwenk und Wolfenstetter 2015, S. 178.
[7] Wikipedia 2021.
[8] Beutelspacher, Schwenk und Wolfenstetter 2015, S. 179.

Kapitel 2

RSA

RSA ist ein kryptographisches Verfahren, welches zu den Public-Key-Verfahren gehört. Der Verfahren wurde von R. Rivest, A. Shamir und L. Adleman entwickelt und trägt deshalb ein Anagram der Erfinder als Namen.

2.1 Ablauf

Ausgangszenario Teilnehmer A will über ein öffentliches Netz sicher mit anderen Teilnehmer kommunizieren.

Schlüsselgeneration Damit andere Teilnehmer geheime Nachrichten schicken können muss A sich ein Schlüsselpaar generieren. Dafür wählt er zwei zufällige und große Primzahlen: p und q.

Das Produkt von p und q bildet n, welche den Modulo / den Zahlenraum für weitere mathematische Operationen festlegt:

$$n = p * q \tag{2.1}$$

Daraufhin berechnet der Teilnehmer die Eulersche ϕ-Funktion von p und q:

$$\phi(n) = (p-1) * (q-1) \tag{2.2}$$

)

Der öffentliche Schlüssel e ist dann eine zu $\phi(n)$ teilerfremde Zahl. Man kann dies auch vereinfachen und eine Fermat'sche Primzahl für e verwenden:

$$2^{2^n} + 1 \mid n \in \{0, 1, 2, 3, 4\} \tag{2.3}$$

Der größte gemeinsame Teiler von e und $\phi(n)$ bildet d, den privaten Schlüssel. Dabei gilt:

$$c * \phi(n) + e * d \equiv 1 \tag{2.4}$$

Danach besitzt der Teilnehmer A ein öffentlichen Schlüssel e und einen privaten Schlüssel d. Er kann nun den öffentlichen Schlüssel e zusammen mit n veröffentlichen. e zusammen mit n und d sind ein Schlüsselpaar welches in einen öffentlichen Teil und ein privaten Teil geteilt wird. Dabei herrscht eine eindeutige Zuordnung zwischen den Teilen.

2.1.1 Verschlüsselung

Wenn Teilnehmer B eine verschlüsselte Nachricht m an Teilnehmer A senden will, braucht er hierfür den öffentlichen Schlüssel von A. Dabei B verschlüsselt wie folgt, wobei c der resultierende Geheimtext ist.

$$m^{e_A} \pmod{n} = c \tag{2.5}$$

Wenn Teilnehmer A den Geheimtext c von B erhält, entschlüsselt er diesen mit seinem privaten Schlüssel. Dadurch berechnet er die von Teilnehmer B verschlüsselte Nachricht m.

$$c^{d_A} \pmod{n} = m \tag{2.6}$$

Da d privat und eindeutig für den öffentlichen Teil des Schlüsselpaars ist, können nur Teilnehmer, die über d verfügen, Nachrichten entschlüsseln, die mit dem zugehörigen öffentlichen Teil verschlüsselt wurden.

2.1.2 Signatur

Das RSA-Verfahren ermöglicht auch das Signieren von Nachrichten. Dies wird z.B. im Internet genutzt um Teilnehmer zu authentifizieren und Nachrichtenintegrität zu sichern. RSA wird jedoch meist nur auf kleinere Nachrichten wie Fingerabdrücke angewandt.

Signatur s einer Nachricht m von Teilnehmer A:

$$m^{d_A} \pmod{n} = s \tag{2.7}$$

Zum Überprüfen der Echtheit braucht der Teilnehmer B den öffentlichen Schlüssel von A:

$$s^{e_A} \pmod{n} = m \tag{2.8}$$

Eine Signatur kann eindeutig Teilnehmer A zugeordnet werden, da nur ein Teilnehmer im Besitz von d_A eine Signatur einer Nachricht erstellen kann, die mit dem öffentlichen Teil des Schlüsselpaars von A überprüft werden kann.

2.2 Sicherheit

Die Sicherheit des RSA-Verfahrens basiert auf zwei mathematischen Problemen, welche unter Aufwand endlicher Ressourcen, nicht gelöst werden können. Hierbei wird sich sowohl auf RSA-gestützte Verschlüsselungs- und Signaturverfahren bezogen. Diese zwei Probleme sind:

- Faktorisierung einer bekannten Zahl, welche das Produkt zweier großer Primzahlen ist. Im Kontext von RSA ist diese Zahl mit n repräsentiert.
- Bestimmung des diskreten Logarithmus. Bei RSA wäre dies die Bestimmung von

$$d \mid m^d \equiv c \pmod{n}. \tag{2.9}$$

Für die Sicherheit der Public-Key-Verschlüsselung von RSA, spielt Unberechenbarkeit der Faktorisierung die Hauptrolle. Falls mit RSA signiert werden soll, ist zusätzlich die Unberechenbarkeit des diskreten Logarithmus wichtig. Ansonsten könnte der private Schlüssel bestimmt werden.

2.2.1 Angriffe auf RSA

Kapitel 3

Kleptographie

3.1 Definition

3.2 Geschichte

3.3 Angriffskategorie

Bisher wurden Angriffe auf kryptographisches Systeme in eine der vier Kategorien 1.1.2 unterteilt (Known Cipher, Known Plaintext, Chosen Cipher, Chosen Plaintext). Ein kleptographischer Angriff fällt jedoch in keiner dieser Kategorien. Für kleptographische Angriffe müsste eine weitere, fünfte Kategorie geschaffen werden: Known Key Attacks.

3.4 Aufbau kleptographischer Angriffe

3.4.1 Voraussetzungen

Die Voraussetzungen für den hier beschriebenen kleptographischen Angriff ist die einmalige Manipulation eines kryptographischen Systems zur Erzeugung von RSA-Schlüsselpaaren. Dies kann während der Herstellung oder des Betriebs des Systems gelingen. Dabei muss nur die Schlüsselerstellung manipuliert werden. Die Funktionen zum Signieren, Verifizieren, Ver- und Entschlüsseln des Systems sind davon sich betroffen.

3.4.2 SETUP

3.5 SETUP für RSA

3.5.1 Voraussetzungen

Für ein Secretly Embedded Trapdoor with Universal Protection (SETUP)-Angriff auf eine Implementation von RSA hat der Angreifer ein eigenes Schlüsselpaar: N_A Modulus des Angreifers, E_A Öffentlicher Schlüssel des Angreifers, D_A Privater Schlüssel des Angreifers. Das Schlüsselpaar wird wie für RSA üblich generiert.

3.5.2 Generierung und Verschlüsselung

Schritt 1 Es wird eine zufällige Primzahl P generiert. P wird dann mit dem öffentlichen Schlüssel des Angreifers verschlüsselt:

$$vP = P^{E_A} \mod N_A \tag{3.1}$$

Schritt 2 N' wird gebildet indem vP und eine Zufallszahl gleicher Länge t in binärer Form konkateniert werden:

$$N' = vP||t \tag{3.2}$$

N' ist dabei nicht der Modulus des generierten RSA-Schlüsselpaars sondern nur eine temporäre Form.

Schritt 3 Berechnung der zweiten Primzahl Q:

$$P \cdot Q + R = N' \tag{3.3}$$

Schritt 4 Bestimmung des Modulus N, wie für RSA üblich durch:

$$N = P \cdot Q \tag{3.4}$$

Schritt 5 Wählen des öffentlichen Schlüssels E und Berechnen des privaten Schlüssels D mittels der modularen multiplikativen Inversen bzgl. $\phi(N)$:

$$D = modular\_multiplicative\_inverse(E, \phi(N)) \tag{3.5}$$

Schritt 6 Mit Schritt 5 wurde ein vollkommen funktionales RSA-Schlüsselpaar erstellt. Mittels diesem können nun Informationen verschlüsselt/signiert, Chiffren entschlüsselt und Signaturen verifiziert werden, wie in 2.1.1 und 2.1.2 gezeigt wurde.

3.5.3 Angriff

Schritt 1 Der Angreifer erlangt den öffentlichen Schlüssel des Ziels und besitzt somit N und E. Dies ist möglich, da diese Informationen öffentlich sind.

Schritt 2 Der Angreifer teilt N in binärer Form in der Hälfte womit er vP erhält. Die mathematische Begründung hierfür in 3.5.4.

Schritt 3 P wird durch die Entschlüsslung von vP mittels des privaten Schlüssels des Angreifers berechnet:

$$P = (vP)^{D_A} \mod N_A \tag{3.6}$$

Damit ist dieser Schritt die inverse Operation zu 3.5.2. Zusätzlich muss auch $vP + 1$ entschlüsselt werden. Die mathematische Begründung hierfür in 3.5.4.

$$P' = (vP + 1)^{D_A} \mod N_A \tag{3.7}$$

Schritt 4 Hiermit ist der Angreifer im Besitz des ersten Primfaktors P oder P'. Somit ist die Primfaktorzerlegung von N trivial:

$$Q = N/P \tag{3.8}$$

Die Primfaktorzerlegung muss, gleich wie bei 3.5.3, für den alternativen Primfaktor P' berechnet werden:

$$Q' = N/P' \tag{3.9}$$

Schritt 5 Der Angreifer ist hiermit im Besitz der Primfaktoren P, Q und kann den privaten Schlüssel D bestimmen (3.5). Gleiches muss für die alternativen Primfaktoren berechnet werden.

Schritt 6 Der Angreifer besitzt den privaten und öffentlichen Schlüssel. Somit können Chiffren entschlüsselt und Signaturen gefälscht werden. Dabei muss der Angreifer, wenn noch nicht geschehen, den privaten Schlüssel D und den alternativen privaten Schlüssel D' einmalig testen, um den richtigen zu bestimmen.

3.5.4 Hintergründe zum RSA-SETUP

Informationsgewinnung von vP aus N

Längendefinitionen:

- N, N' hat n Bit
- vp, R hat $n/2$ Bit

vP kann aus N bestimmt werden, durch die Beziehung von N und N'. Dabei gilt $N + R = N'$, wobei R dem Angreifer nicht bekannt ist. Jedoch ist definiert, dass R nur halb so viele Bits wie N hat. Dadurch findet die Addition von $N + R$ nur auf den niederwertigeren Bits von N statt. Somit sind die höherwertigen Bits von N (Bits $> (n/2)$) nicht von der Addition mit R beeinflusst und somit gleich zu den höherwertigen Bits von N'. Dies ist nur dann nicht der Fall, wenn es bei der Addition zu einem Überlauf für. In diesem Fall sind alle Bit $> (n/2)$ von N, die höherwertigen Bits von $N' + 1$. Zudem ist definiert, dass die höherwertigen von N' gleich vP sind (siehe 3.2). Da der Angreifer nur N aber nicht auch R kennt, muss die Möglichkeit eines Überlauf durch R berücksichtigt werden. Aufgrund dessen kann der Angreifer vP aus den höherwertigen Bits von N lesen. Muss aber auch $vP + 1$ als Option berücksichtigen, da er nicht weiß, ob ein Überlauf stattgefunden hat.

3.5.5 Verfahren zur Bestimmung der korrekten Primfaktoren

Die Schritte 3 bis 6 des Angriffs befassen sich mit dem Finden des korrekten Primfaktor aus den zwei resultierenden Möglichkeiten von vP vP und $vP + 1$. Daraus werden die Werte und ihre Alternativen für P, N, Q und D berechnet. Um schlussendlich zu entscheiden, ob die Werte die aus vP oder $vP + 1$, kann eine Signatur mit D und D' mit einer Signatur des Angriffsziels verglichen werden. Dadurch kann eine eindeutige Entscheidung getroffen werden.

Diese Entscheidung kann jedoch unter Umständen früher berechnet werden. Dieser Fall kann bei folgenden Berechnungsschritten auftreten:

Berechnung von P

P wird berechnet indem vP mittels dem privaten Schlüssel D_A des Angreifers entschlüsselt wird. Dabei sollte, wie für eine RSA-Ver-/Entschlüsselung üblich, eine vollkommen zufällige Zahl resultieren. Jedoch ist es eine Bedingung, dass P prim ist. Die Wahrscheinlichkeit, dass eine Zufallszahl, mit steigender Anzahl an Stellen, prim ist sehr gering. Mathematische Erläuterung

Falls das entschlüsselte P nicht prim ist, muss es die Alternative P' sein. Gleiches gilt wiederum auch für P'.

Berechnung von Q

Bei der Berechnung von Q gilt die gleiche Eigenschaft, wie bei P, dass Q prim seien muss.

Jedoch kann bei der Berechnung von Q eine eindeutige Entscheidung getroffen werden. Der Grund dafür ist, dass Q eine ganze Zahl seien muss, also $Q \in (Z)$. Ohne weitere geltende Bedingungen wäre die Überprüfung von $Q = N/P$ und $Q' = N/P'$ auf

$$\text{Correct prime factors} = \begin{cases} (P, Q) & Q \in Z \wedge Q' \notin (Z) \\ (P', Q') & Q' \in Z \wedge Q \notin (Z) \\ (P, Q) & sonst \end{cases} \tag{3.10}$$

Ohne geltende Bedingungen von RSA könnte $Q, Q' \in (Z)$ gelten. Da N ein vielfaches beider Zahlen P und P' seien kann. Jedoch ist N in RSA das Produkt von zwei Primzahlen. Dadurch gibt es zwei Zahlen x für die gilt $(N/x) \in (Z)$. Dabei handelt es sich um die korrekten Primfaktoren P und Q. Dadurch gilt immer nur einer der beiden Fälle in 3.10. Im Fall 1 ist $(N/P) \in \mathbb{Z}$, während N kein vielfaches von P' ist. Umgekehrtes gilt für Fall 2. Der dritte Fall wird dennoch benötigt. Er tritt ein, wenn $Q' = P$ oder $Q = P'$ gilt. Dieser Fall tritt dann ein, wenn $(P^E + 1)^D \bmod N = Q$ gilt. Unter der Annahme, dass die Entschlüsselung einer Hash-Funktion gleich, tritt dieser Fall mit einer Wahrscheinlichkeit von $p(1/N)$ auf. Dieser sehr unwahrscheinliche Fall wird durch Fall 3 von 3.10 abgedeckt. In diesem Fall wird eine der beiden Möglichkeiten ausgewählt, da durch die Kommutativität der Multiplikation es egal ist, ob $[P, Q] \vee [Q, P]$ die richtigen Primfaktoren von N sind.

Durch 3.10 kann sehr einfach und effizient die richtigen Primfaktoren ausgewählt werden. Diese Überprüfung hat keinen Einfluss auf die Laufzeit des Algorithmus.

Somit ist diese Überprüfung nicht nur eindeutiger als eine Überprüfung von P auf prim, sondern auch effizienter.

Fehler bei der modularen multiplikativen Inverse

Bei der Berechnung von D wird die modulare multiplikative Inverse von E und $\phi(N)$ bestimmt. Dabei kann es zu einem Fehler kommen, da für den Tupel von E und $\phi(N)$ möglicherweise keine modulare multiplikative Inverse existiert.

Kapitel 4

Angriffskonzept

4.1 Ziel

Folgender Abschnitt der Arbeit, befasst sich mit der Auswahl des Ziels für einen kleptographischen Angriff. Hierbei soll es sich um eine Softwarebibliothek handeln. Zudem soll diese Open-Source und hinreichend verbreitet sein. Die zugrundeliegende Programmiersprache soll abstrakt genug sein, dass die kryptographischen Operationen in Software abgebildet werden. Programmiersprachen, welche Hardware, wie Trusted Platform Module (TPM) nutzten, sind nicht für einen Angriff auf Softwarebibliotheken weniger geeignet.

Kapitel 5

Implementation

Es folgt die konkrete Implementation einer kleptographsichen Schwachstelle in die ausgewählte Softwarebibliothek.

Kapitel 6

Risikoanalyse

In dieser Risikoanalyse (engl. Threat Assessment) wird die Gefahr analysiert und evaluiert, welche kleptographische Angriffe auf kryptographische Softwarebibliotheken bilden.

6.1 Angriffsvektoren

6.2 Gegenmaßnahmen

6.3 Risiko

Kapitel 7

Fazit

Literatur

BEUTELSPACHER, Albrecht [2015]. *Kryptologie.* springer [siehe S. 10, 11].

BEUTELSPACHER, Albrecht, Jörg SCHWENK und Klaus-Dieter WOLFENSTETTER [2015]. *Moderne Verfahren der Kryptographie.* springer [siehe S. 8, 9, 12, 14].

WIKIPEDIA [2021]. *Komplexitätstheorie* [siehe S. 14].