Kryptografische Grundlagen

Einführung

Kryptografie bezeichnet Verfahren, mit denen Informationen durch mathematische Methoden geschützt werden.

Moderne kryptografische Verfahren kommen insbesondere bei der Verschlüsselung von Dateien und Datenträgern, der sicheren Datenübertragung, der Authentifizierung und bei digitalen Signaturen zum Einsatz.

Dabei muss zwischen dem **kryptografischen Algorithmus**, dem **Schlüssel** und einer gegebenenfalls verwendeten **Passphrase** unterschieden werden.

Ein Algorithmus beschreibt das mathematische Verfahren. Der Schlüssel bestimmt, wie das Verfahren bei einer konkreten Anwendung arbeitet.

Die Sicherheit eines modernen kryptografischen Systems soll nicht von der Geheimhaltung des Algorithmus abhängen, sondern insbesondere vom Schutz seiner geheimen Schlüssel.

1. Symmetrische Kryptografie

Bei der symmetrischen Verschlüsselung wird derselbe geheime Schlüssel zum Verschlüsseln und Entschlüsseln verwendet.

Der Schlüssel muss allen berechtigten Kommunikationspartnern bekannt sein und geheim gehalten werden.

Funktionsprinzip

```
Klartext + geheimer Schlüssel
             |
             v
       Verschlüsselung
             |
             v
        Geheimtext
             |
             v
       Entschlüsselung
             |
             v
         Klartext
```

Für die Entschlüsselung wird derselbe geheime Schlüssel benötigt.

Typische Algorithmen

Algorithmus	Eigenschaften	Typische Anwendung
AES-128	Symmetrische Blockchiffre mit 128-Bit-Schlüssel	Datenträger, Netzwerkverbindungen
AES-256	Symmetrische Blockchiffre mit 256-Bit-Schlüssel	Dateien, Backups, Datenträger
ChaCha20	Symmetrische Stromchiffre	Netzwerkprotokolle, mobile Geräte
XChaCha20	Variante mit erweitertem Nonce	Moderne Datei- und Datenverschlüsselung

AES und ChaCha20 werden häufig in authentifizierten Verschlüsselungsverfahren eingesetzt, beispielsweise AES-GCM oder ChaCha20-Poly1305.

Diese Verfahren schützen nicht nur die Vertraulichkeit, sondern ermöglichen auch die Erkennung unbemerkter Veränderungen an verschlüsselten Daten.

Vorteile

- Hohe Verarbeitungsgeschwindigkeit
- Geeignet für große Datenmengen
- Vergleichsweise geringer Rechenaufwand
- Bewährte Verfahren für Dateien, Datenträger und Netzwerkverbindungen

Nachteile

- Der geheime Schlüssel muss sicher aufbewahrt werden.
- Bei mehreren Kommunikationspartnern ist eine sichere Schlüsselverteilung erforderlich.
- Wer den geheimen Schlüssel besitzt, kann grundsätzlich die damit geschützten Daten entschlüsseln.

Typische Anwendungen

- VeraCrypt
- LUKS / dm-crypt
- BorgBackup
- GnuPG mit symmetrischer Verschlüsselung
- Verschlüsselte TLS-, SSH- und WireGuard-Verbindungen

2. Asymmetrische Kryptografie

Bei der asymmetrischen Kryptografie werden zwei mathematisch zusammengehörende Schlüssel verwendet.

- **Private Key:** Privater Schlüssel, der geheim gehalten werden muss.
- **Public Key:** Öffentlicher Schlüssel, der weitergegeben werden darf.

Die beiden Schlüssel erfüllen je nach Algorithmus unterschiedliche Aufgaben.

Asymmetrische Verschlüsselung

Bei einem geeigneten asymmetrischen Verschlüsselungsverfahren können Daten mit dem öffentlichen Schlüssel eines Empfängers verschlüsselt werden.

Nur mit dem zugehörigen privaten Schlüssel lassen sie sich entschlüsseln.

```
Öffentlicher Schlüssel
          |
          v
     Verschlüsselung
          |
          v
       Geheimtext
          |
          v
     Entschlüsselung
          ^
          |
 Privater Schlüssel
```

In der Praxis werden größere Datenmengen normalerweise nicht unmittelbar asymmetrisch verschlüsselt.

Stattdessen wird ein symmetrischer Schlüssel für die Datenverschlüsselung verwendet und dieser über ein asymmetrisches Verfahren geschützt oder vereinbart.

Dieses Vorgehen wird als **hybride Verschlüsselung** bezeichnet.

Digitale Signaturen

Bei digitalen Signaturen ist die Verwendung der Schlüssel anders:

- Der private Signaturschlüssel erzeugt die Signatur.
- Der öffentliche Signaturschlüssel überprüft die Signatur.

Eine gültige Signatur bestätigt, dass die signierten Daten seit der Signatur nicht verändert wurden und die Signatur zum verwendeten öffentlichen Schlüssel passt.

Ob dieser Schlüssel tatsächlich zu einer bestimmten Person oder Organisation gehört, muss gesondert überprüft werden.

Typische asymmetrische Algorithmen

Algorithmus	Funktion	Typische Anwendung
RSA	Verschlüsselung oder digitale Signaturen, abhängig vom Verfahren	OpenPGP, Zertifikate
Ed25519	Digitale Signaturen	OpenSSH, OpenPGP
X25519	Vereinbarung gemeinsamer Geheimnisse	WireGuard, TLS, OpenPGP

Algorithmus	Funktion	Typische Anwendung
ECDSA	Digitale Signaturen	TLS-Zertifikate, SSH

Wichtig: Nicht jeder asymmetrische Algorithmus eignet sich sowohl zum Verschlüsseln als auch zum Signieren.

Ed25519 ist beispielsweise ein Signaturverfahren, während X25519 zur Schlüsselvereinbarung verwendet wird.

Vorteile

- Öffentliche Schlüssel dürfen offen verteilt werden.
- Geeignet für digitale Signaturen und Authentifizierung.
- Ermöglicht sichere Schlüsselvereinbarung über unsichere Kommunikationswege.
- Private Schlüssel müssen nicht an Kommunikationspartner weitergegeben werden.

Nachteile

- Höherer Rechenaufwand als bei symmetrischen Verfahren.
- Die Zuordnung öffentlicher Schlüssel zu Identitäten muss überprüft werden.
- Der Verlust eines privaten Entschlüsselungsschlüssels kann zum dauerhaften Verlust des Datenzugriffs führen.

3. Hybride Verschlüsselung

Viele moderne Anwendungen kombinieren symmetrische und asymmetrische Verfahren.

Dabei übernimmt die asymmetrische Kryptografie die sichere Schlüsselvereinbarung oder den Schutz eines Sitzungsschlüssels.

Die eigentlichen Nutzdaten werden anschließend symmetrisch verschlüsselt.

Typische Beispiele sind:

- TLS / HTTPS
- OpenSSH
- OpenPGP
- WireGuard

Die konkrete Schlüsselvereinbarung unterscheidet sich je nach Protokoll.

Bei TLS und SSH werden üblicherweise kurzlebige Sitzungsschlüssel vereinbart.

OpenPGP verschlüsselt dagegen typischerweise einen zufällig erzeugten Sitzungsschlüssel für die jeweiligen Empfänger.

4. Schlüssel, Passwörter und Passphrasen

Ein kryptografischer Schlüssel ist ein Wert, der für eine kryptografische Operation verwendet wird.

Er wird bei modernen Verfahren normalerweise durch einen kryptografisch sicheren Zufallszahlengenerator erzeugt.

Ein Passwort oder eine Passphrase ist dagegen eine vom Benutzer eingegebene Zeichenfolge.

Schlüsselableitung

Bei passwortbasierter Verschlüsselung wird aus einer Passphrase mithilfe einer Schlüsselableitungsfunktion ein kryptografischer Schlüssel oder ein Schlüssel zum Schutz weiterer Schlüssel erzeugt.

Geeignete Schlüsselableitungsverfahren erschweren das systematische Ausprobieren von Passwörtern.

Beispiele:

- Argon2id
- scrypt
- PBKDF2

Die Sicherheit hängt von der Stärke der Passphrase, dem verwendeten Verfahren und dessen Parametern ab.

Schutz privater Schlüssel

Private Schlüssel können zusätzlich durch eine Passphrase geschützt werden.

Bei OpenSSH wird beispielsweise die private Schlüsseldatei verschlüsselt gespeichert, wenn bei der Erstellung eine Passphrase vergeben wird.

Die Passphrase schützt dann die gespeicherte Schlüsseldatei.

Sie ist nicht identisch mit dem eigentlichen privaten Schlüssel.

5. Algorithmen und Schlüssellängen

Die Schlüssellänge beeinflusst die Sicherheit eines kryptografischen Verfahrens.

Allerdings sind Schlüssellängen unterschiedlicher Algorithmen nicht direkt miteinander vergleichbar.

Beispielsweise bietet ein 256-Bit-Schlüssel bei AES nicht dieselbe Art von Sicherheitsbewertung wie ein 256-Bit-Schlüssel bei einem elliptischen Kurvenverfahren.

Verfahren	Übliche Schlüssellänge	Bemerkung
AES-128	128 Bit	Für viele Anwendungen ausreichend
AES-256	256 Bit	Hohe Sicherheitsreserve
ChaCha20	256 Bit	Moderne symmetrische Stromchiffre
RSA	3072 oder 4096 Bit	Deutlich längere Schlüssel erforderlich
Ed25519	256-Bit-Kurvensystem	Signaturverfahren mit ungefähr 128 Bit klassischer Sicherheitsstärke
X25519	256-Bit-Kurvensystem	Schlüsselvereinbarung mit ungefähr 128 Bit klassischer Sicherheitsstärke

Die Wahl eines sicheren Algorithmus ist mindestens ebenso wichtig wie die Schlüssellänge.

Veraltete Verfahren wie DES, RC4 oder MD5 für sicherheitskritische kryptografische Zwecke sollten nicht neu eingesetzt werden.

6. Zufallszahlen und Schlüsselgenerierung

Kryptografische Schlüssel müssen ausreichend zufällig erzeugt werden.

Vorhersagbare Schlüssel können die Sicherheit eines ansonsten geeigneten Algorithmus vollständig aufheben.

Unter Linux steht hierfür ein kryptografisch sicherer Zufallszahlengenerator des Betriebssystems zur Verfügung.

Beispiel: Zufälligen Schlüssel erzeugen

Mit OpenSSL lässt sich ein zufälliger Wert von 32 Byte erzeugen:

```
openssl rand -hex 32
```

Die Ausgabe enthält 64 hexadezimale Zeichen und entspricht 256 Bit Zufallsdaten.

Der Wert kann beispielsweise als Ausgangsmaterial für einen symmetrischen Schlüssel verwendet werden, sofern das jeweilige Verfahren dieses Format unterstützt.

Achtung: Der erzeugte Wert wird im Terminal angezeigt und sollte nicht ungeschützt protokolliert oder veröffentlicht werden, wenn er tatsächlich als geheimer Schlüssel verwendet wird.

Beispiel: SSH-Schlüsselpaar erzeugen

Ein Ed25519-Schlüsselpaar kann unter Linux folgendermaßen erzeugt werden:

```
ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519_beispiel -C "beispiel"
```

Bei der Erstellung sollte für interaktiv genutzte Schlüssel eine geeignete Passphrase vergeben

werden.

Es entstehen zwei Dateien:

- `id_ed25519_beispiel` – privater Schlüssel
- `id_ed25519_beispiel.pub` – öffentlicher Schlüssel

Der private Schlüssel darf nicht veröffentlicht werden.

Weitere Informationen: OpenSSH und SSH-Schlüssel

7. Schlüssel und Zertifikate

Ein öffentlicher Schlüssel enthält für sich genommen noch keinen vertrauenswürdigen Nachweis über die Identität seines Besitzers.

Ein digitales Zertifikat kann einen öffentlichen Schlüssel mit einer Identität verbinden.

Bei TLS-Zertifikaten erfolgt diese Zuordnung üblicherweise durch eine Zertifizierungsstelle (Certificate Authority, CA).

Ein Zertifikat enthält beispielsweise:

- Informationen über den öffentlichen Schlüssel
- Den Namen der zugehörigen Domain
- Den Aussteller
- Den Gültigkeitszeitraum
- Eine digitale Signatur des Ausstellers

Ein TLS-Zertifikat für `wiki.bluegnu.de` ermöglicht einem Browser, die Verbindung zu dieser Domain kryptografisch zu überprüfen, sofern die Zertifikatskette und die weiteren Prüfbedingungen gültig sind.

Weitere Informationen: TLS und Let's Encrypt

8. Sicherheit und Schlüsselverlust

Die Sicherheit kryptografischer Verfahren hängt wesentlich vom Umgang mit den Schlüsseln ab.

Dabei sind drei Fälle zu unterscheiden.

Verlust eines Zugangsschlüssels

Ein verlorener SSH- oder WireGuard-Schlüssel kann normalerweise durch ein neues Schlüsselpaar ersetzt werden.

Voraussetzung ist, dass noch eine andere Möglichkeit zur administrativen Verwaltung des Systems besteht.

Verlust eines Entschlüsselungsschlüssels

Geht der einzige Schlüssel verloren, mit dem verschlüsselte Daten entschlüsselt werden können, sind diese Daten unter Umständen dauerhaft unzugänglich.

Deshalb müssen unersetzliche Entschlüsselungsschlüssel und Wiederherstellungsinformationen zuverlässig gesichert werden.

Kompromittierung eines Schlüssels

Ein Schlüssel gilt als kompromittiert, wenn ein unberechtigter Dritter Zugriff darauf erlangt haben könnte.

Je nach Verfahren müssen dann:

- Der Schlüssel ausgetauscht werden.
- Zugehörige Zugriffsberechtigungen entfernt werden.
- Zertifikate widerrufen oder ersetzt werden.
- Betroffene Daten gegebenenfalls mit neuen Schlüsseln verschlüsselt werden.
- Bereits erfolgte Zugriffe überprüft werden.

Ein bloßer Schlüsselwechsel macht möglicherweise bereits offengelegte Daten nicht wieder geheim.

9. Weiterführende Dokumentation

- Hashwerte, Fingerprints und digitale Signaturen
- GnuPG und OpenPGP
- VeraCrypt
- LUKS2 / dm-crypt
- OpenSSH
- BorgBackup
- WireGuard
- TLS und Let's Encrypt
- Schlüsselverwaltung

Zurück zur Übersicht: Kryptografische Schlüssel und Zertifikate

From:
https://wiki.bluegnu.de/ - **wiki**

Permanent link:
https://wiki.bluegnu.de/doku.php/open:it:xold_grundlagen

Last update: **2026/11/15 17:02**