Kryptografische Grundlagen

Schutzziele

Ziel	Bedeutung
Vertraulichkeit	Nur Berechtigte können Inhalte lesen.
Integrität	Unbemerkte Änderungen werden erkannt.
Authentizität	Die Herkunft eines Schlüssels oder einer Nachricht lässt sich prüfen.
Verfügbarkeit	Daten und Schlüssel bleiben bei Bedarf nutzbar; Verschlüsselung allein gewährleistet dies nicht.

Symmetrische Verschlüsselung

Bei symmetrischer Verschlüsselung wird derselbe geheime Schlüssel zum Ver- und Entschlüsseln verwendet. Beispiele sind AES-256-GCM und ChaCha20-Poly1305. Beide verbinden Verschlüsselung mit einer Integritätsprüfung (AEAD). Die sichere Verwendung von Nonces und Schlüsseln ist entscheidend.

Asymmetrische Kryptografie

Ein Schlüsselpaar besteht aus einem privaten und einem öffentlichen Schlüssel. Je nach Algorithmus dient es der Signatur, der Verschlüsselung oder dem Schlüsselaustausch. Ed25519 ist ein Signaturverfahren, X25519 ein Verfahren zur Schlüsselvereinbarung; die beiden sind nicht beliebig austauschbar.

Hybride Verfahren

Große Datenmengen werden meist symmetrisch verschlüsselt. Ein asymmetrisches Verfahren schützt den Sitzungsschlüssel oder vereinbart ihn zwischen Kommunikationspartnern. TLS, SSH und OpenPGP nutzen solche Kombinationen in unterschiedlicher Weise.

Passphrase und Schlüsselableitung

Ein Passwort ist nicht zwangsläufig der kryptografische Schlüssel. Schlüsselableitungsfunktionen wie Argon2id, scrypt oder PBKDF2 erschweren Wörterbuchangriffe. Entscheidend sind starke Passphrasen und geeignete Parameter. Private Schlüssel sollten nach Möglichkeit mit einer Passphrase geschützt werden.

Schlüssellängen

Schlüssellängen sind zwischen Algorithmen nicht direkt vergleichbar. AES-256 verwendet einen 256-Bit-Schlüssel; RSA benötigt wesentlich längere Schlüssel für vergleichbare klassische Sicherheitsniveaus. Für neue SSH-Benutzerschlüssel ist Ed25519 häufig eine gute Standardwahl.

Zufall und Schlüsselgenerierung

Schlüssel müssen mit einem kryptografisch sicheren Zufallszahlengenerator erzeugt werden. Beispiel zur Anzeige von 32 zufälligen Bytes in Hexadezimaldarstellung:

```
openssl rand -hex 32
```

Achtung: Die Ausgabe ist geheim, wenn sie als echter Schlüssel verwendet wird. Sie kann im Terminalverlauf oder in Protokollen sichtbar werden.

Um das zu vermeiden, kann das Ergebnis direkt in eine Datei geschrieben werden:

```
openssl rand -hex 32 > /pfad/datei.txt
```

Wenn ein Programm einen binären Schlüssel mit exakt 32 Bytes benötigt (Speicherung ohne Hexadezimalkodierung)

```
openssl rand -out /pfad/schluessel.bin 32
```

Verlust und Kompromittierung

Zugangsschlüssel können meist ersetzt werden, sofern ein alternativer Administrationsweg existiert. Geht der einzige Entschlüsselungsschlüssel verloren, können Daten dauerhaft unzugänglich werden. Bei einer Kompromittierung müssen betroffene Zugriffsrechte entzogen und Schlüssel erneuert werden. Bereits abgeflossene Daten werden dadurch nicht nachträglich geheim.

Zurück zur Übersicht

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

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

Last update: **2026/11/15 20:10**