

Kryptografie I

IT-Security

Andreas Unterweger

Studiengang Web Business & Technology
FH Kufstein

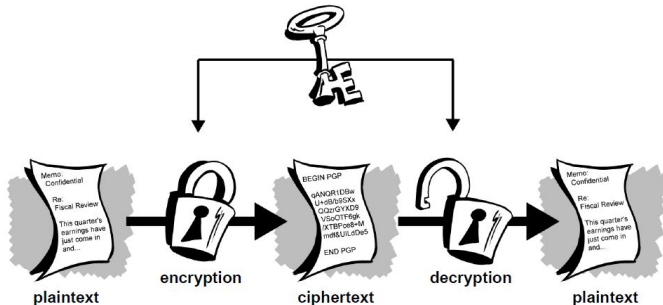
Sommersemester 2019

- Kryptografie
 - Aus dem Griechischen („verborgen schreiben“)
 - Nachrichten/Daten in sichere(re) Form bringen
- Ziele
 - Daten für Außenstehende unlesbar machen
 - Kein Datenzugriff für unautorisierte Personen
 - CIA-Schutzziele
- Weitere Schutzziele (mit Spezialverfahren, siehe nächste Vorlesung)
 - Authentifizierung (engl. *authentication*): Echtheit des Absenders (der Absender ist der, für den er sich ausgibt)
 - Nichtabstreitbarkeit (engl. *non-repudiation*): Nachricht wurde gesendet (nachträgliches Abstreiten, dass gesendet wurde, ist nicht möglich)

- **Verschlüsselung** (klassische Kryptografie)
 - Symmetrisch (alle CIA-Schutzziele)
 - Asymmetrisch (zusätzlich Authentifizierungsschutzziel)
- **Hashes**
 - Nur Integritätsschutzziel
- Signaturen (nächste Vorlesung)
- Zertifikate (nächste Vorlesung)

Grundbegriffe der Verschlüsselung

- Klartext (engl. *plaintext* p): (Entschlüsselte) Nachricht
- Geheimtext (engl. *ciphertext* c): Verschlüsselte Nachricht
- Verschlüsselung (engl. *encryption* E): Klartext in Geheimtext bringen
- Entschlüsselung (engl. *decryption* D): Geheimtext in Klartext bringen
- Schlüssel (engl. *key* k): Parameter für Verschlüsselung



Quelle: Pacia, C.: Beginners' Guide To Off-the-Record Messaging.

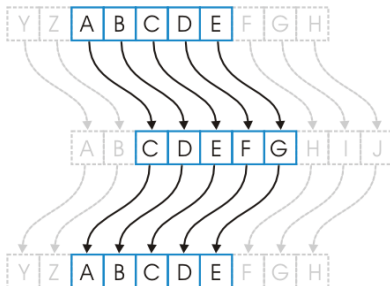
<http://www.bitcoinnotbombs.com/beginners-guide-to-off-the-record-messaging/> (Zugriff am 18.2.2017), 2014.

Arten von Verschlüsselungsalgorithmen

- Symmetrisch
 - Sender und Empfänger verwenden den gleichen Schlüssel
 - $c = E(k; p)$ und $p = D(k; c)$
 - Schlüsselaustausch zwischen Sender und Empfänger notwendig
 - Asymmetrisch
 - Sender und Empfänger verwenden unterschiedliche Schlüssel
 - $c = E(k_1; p)$ und $p = D(k_2; c)$
 - Schlüsselaustausch unproblematisch (ein Schlüssel öffentlich)
 - Allgemeine Notation für Ver- bzw. Entschlüsselungsalgorithmen f :
 $f(x; y)$, wobei x Schlüssel und y Klar- bzw. Geheimtext
 - Bei allen Algorithmen ist **Schlüssellänge** entscheidend für Sicherheit
- Angreifer muss alle möglichen Schlüssel ausprobieren

Beispiele für symmetrische Kryptografie I

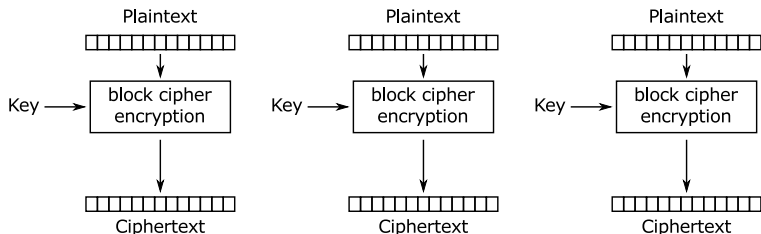
- Caesar-Chiffre: Alphabet um n (z.B. 3) Zeichen verschieben
- n ist Schlüssel (muss Sender und Empfänger bekannt sein)
- Empfänger verschiebt um $+n$, Empfänger um $-n$
- Unsicher, da Schlüsselraum sehr klein



Quelle: Smith, J.: A functional implementation of the Caesar cipher in Swift.
<https://github.com/ijoshsmith/swift-caesar-cipher> (Zugriff am 19.2.2017), 2015.

Beispiele für symmetrische Kryptografie II

- Advanced Encryption Standard (AES)
 - Variationen mit unterschiedlichen Schlüssellängen (128, 192, 256 Bit)
 - Klartext wird blockweise in Geheimtext übersetzt
 - Aktuell sicher (keine relevanten Angriffe bekannt)
 - Standardbetriebsmodus *Electronic Code Book* (ECB) **nicht** sicher



Electronic Codebook (ECB) mode encryption

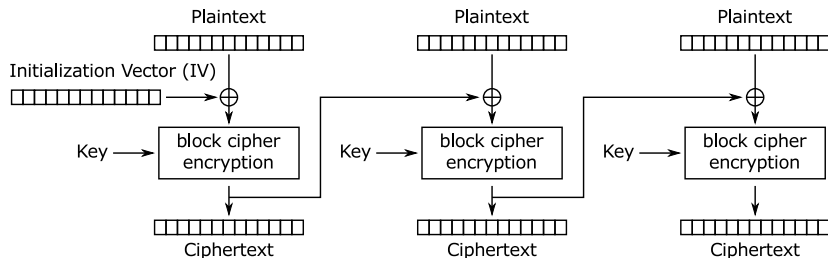
Quelle: Timberwolf, W.: Encryption using the Electronic Code Block (ECB) mode.

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:ECB_encryption.svg (Zugriff am 19.2.2017), 2013.

Beispiele für symmetrische Kryptografie III

- AES im Modus Cipher Block Chaining (CBC)

- Geheimtext eines Blocks hängt vom Geheimtext vorheriger Blöcke ab
- Keine Muster in Blöcken mehr erkennbar
- Initialisierungsvektor (IV) für ersten Block (lang und praktisch zufällig)
- Erlaubt Wiederverwendung des gleichen Schlüssels für Nachrichten



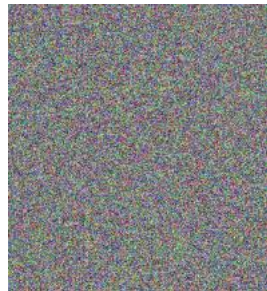
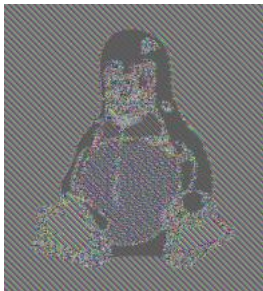
Cipher Block Chaining (CBC) mode encryption

Quelle: Timberwolf, W.: Encryption using the Cipher Block Chaining (CBC) mode.

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:CBC_encryption.svg (Zugriff am 19.2.2017), 2013.

Beispiele für symmetrische Kryptografie IV

- Klartext vs. Geheimtext AES (ECB) vs. Geheimtext AES (CBC)



Quellen: Ewing, L.: Tux. https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:Tux.jpg (Zugriff am 19.2.2017), 1996; Lunkwill: Encrypted using ECB mode.

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:Tux_ecb.jpg (Zugriff am 19.2.2017), 2006; Lunkwill: Modes other than ECB result in pseudo-randomness.

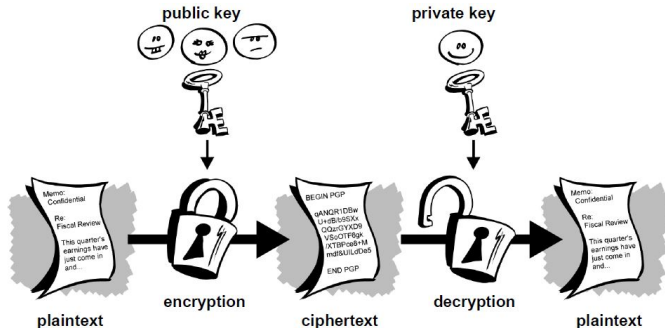
https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:Tux_secure.jpg (Zugriff am 19.2.2017), 2006.

- Negativbeispiele WEP (WLAN): Kurze, wenig zufällige IV → WPA

- Schlüssel (und evtl. IV) wird von Sender und Empfänger verwendet
- Sicherheit hängt **nur** von Schlüssel/IV ab (Kerckhoffs Prinzip)
- Schlüssel **muss** geheim gehalten werden
- **Problem: Wie Schlüssel/IV austauschen?**
- Diffie-Hellman-Schlüsselaustausch (DH-Schlüsselaustausch)
 - Gemeinsame Berechnung, die schwer umkehrbar ist, mit teilweise öffentlichen Werten (ohne Details)
 - Beide Parteien generieren mathematisch gleichen Schlüssel
- Diffie-Hellman Ephemeral (DHE)
 - DH-Schlüsselaustausch für jede neue Nachricht → neue Schlüssel
 - *Perfect Forward Secrecy*: Ein Schlüssel entschlüsselt nur eine Nachricht
 - Andere Nachrichten sind nicht kompromittiert

Kryptografie mit öffentlichen Schlüsseln

- Öffentlicher Schlüssel (engl. *public key*): Darf öffentlich bekannt gegeben werden und wird zur Verschlüsselung verwendet
- Privater Schlüssel (engl. *private key*): Muss vom Empfänger geheim gehalten werden und wird zur Entschlüsselung verwendet



Quelle: Pacia, C.: *Beginners' Guide To Off-the-Record Messaging*.

<http://www.bitcoinnotbombs.com/beginners-guide-to-off-the-record-messaging/> (Zugriff am 18.2.2017), 2014.

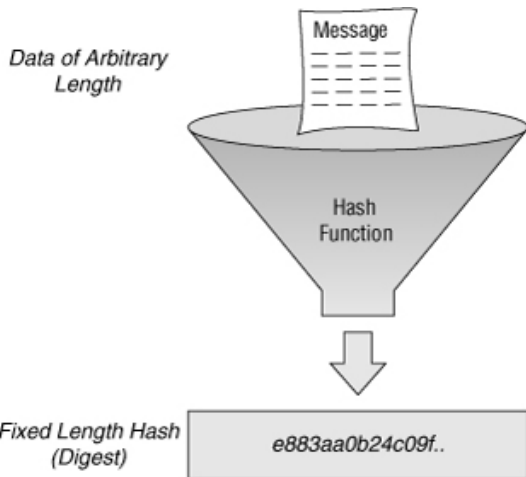
- Rivest-Shamir-Adleman (RSA)
 - Schlüssellängen typischerweise zwischen 1024 und 3072 Bit
 - Relativ aufwändig zu berechnen (große Werte)
 - Mathematische Grundlage: Primfaktorenzerlegung
 - Aktuell sicher (keine relevanten Angriffe bekannt)
- Elliptische Kurven (engl. *elliptic curve cryptography*, ECC)
 - Schlüssellängen typischerweise zwischen 128 und 384 Bit
 - Schneller zu berechnen als RSA (kleinere Werte)
 - Mathematische Grundlage: Diskreter Logarithmus (elliptische Kurven)
 - Aktuell sicher (keine relevanten Angriffe bekannt)
- Quellen für aktuell sichere Schlüssellängen:
 - BlueKrypt: Cryptographic Key Length Recommendation. <https://www.keylength.com/en/compare/> (Zugriff am 19.2.2017), 2017.
 - Lenstra, A. K. und Verheul, E. R.: Selecting Cryptographic Key Sizes. *Journal Of Cryptology*, vol. 14, pp. 255-293, 2001. Auch verfügbar unter <https://infoscience.epfl.ch/record/164526/files/NPDF-22.pdf>.

- Datei-(system-)Verschlüsselung
 - Einzelne Dateien und/oder gesamtes Dateisystem verschlüsseln
 - Verwendet meist symmetrische Verschlüsselung
 - Teilweise transparent für den Benutzer (evtl. Passworteingabe)
 - Beispiele: EFS (Microsoft), VeraCrypt (IDRIX)
- Festplatten-(partitions-)Verschlüsselung
 - Gesamte Festplatte(-npartition) ist verschlüsselt
 - Verwendet meist symmetrische Verschlüsselung
 - Erfordert Passworteingabe beim Booten bzw. Mounten
 - Beispiele: BitLocker (Microsoft), VeraCrypt (IDRIX)
- Meist Wiederherstellungsmöglichkeit: Symmetrischen Schlüssel asymmetrisch verschlüsseln und speichern; bei Bedarf entschlüsseln

- Hardwareseitige Verschlüsselung
 - Gerät (z.B. USB-Stick) ver- bzw. entschlüsselt bei Zugriff
 - Vorteil: Schneller, da Verschlüsselung auf eigenem Prozessor
 - Verwendet symmetrische Verschlüsselung
 - Erfordert Passwordeingabe o.ä. beim Verbinden
 - Beispiel: Kingston IronKey
- *Trusted Platform Module*
 - Auf Mainboard fest verlöteter Chip
 - Bietet Funktionen zur Schlüsselerstellung u.ä.
 - Ermöglicht z.B. Festplattenverschlüsselung
 - Vorteil: Per Software praktisch nicht auszuhebeln
 - Für Benutzer transparent (außer bei Mainboardwechsel)
 - Aktuell auf praktisch allen Mainboards verbaut

- Definition
 - Erzeugt „Fingerabdruck“ der (Eingabe-)Daten
 - „Fingerabdruck“ ist immer gleich lang
 - Unumkehrbar (Einwegfunktion)
- Ziele
 - Kurze Repräsentation der Daten (Prüfsumme)
 - Erlaubt Vergleich von Daten ohne Originale
 - Ermöglicht Integritätsprüfung (CIA-Schutzziel)
- Ideale Eigenschaften (praktisch nicht erreichbar)
 - Ungleiche Eingabedaten führen immer zu ungleichen Hashes
 - Erzeugung von alternativen Eingabedaten mit gleichem Hash unmöglich
- Praxis: Schwierig, Alternativdaten mit gleichem Hash zu erzeugen

Begriff Hash II

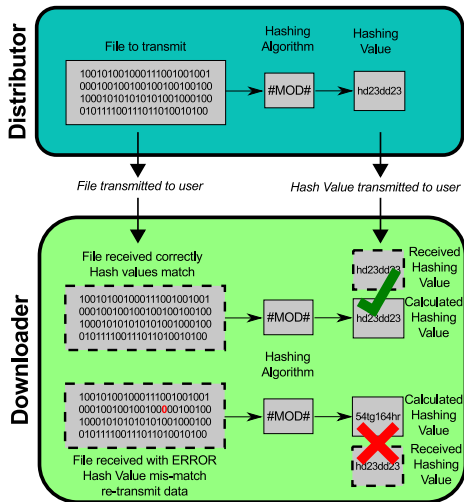


Quelle: Long, H. P.: Chapter 04: Fundamentals of Cryptography (Part01).

<http://ciscodocuments.blogspot.co.at/2011/05/chapter-04-fundamentals-of-cryptography.html> (Zugriff am 18.2.2017), 2011.

- Message Digest 5 (MD5)
 - Ausgabelänge: 128 Bit
 - Nicht mehr sicher (Angriffe bekannt)
- Secure Hashing Algorithm 2 (SHA-2)
 - Sechs Variationen mit Ausgabelängen zwischen 224 und 512 Bit
 - Ausgabelänge z.B. 256 Bit bei SHA-256
 - Aktuell sicher (keine relevanten Angriffe bekannt)
- Quelle für sichere Ausgabelängen: BlueKrypt: Cryptographic Key Length Recommendation.
<https://www.keylength.com/en/compare/> (Zugriff am 19.2.2017), 2017.

Anwendung Integritätsprüfung

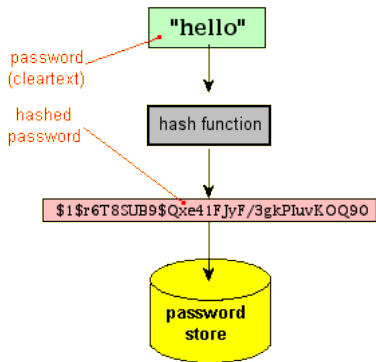


Quelle: Pluke: Diagram showing use of MD5 hashing in file transmission.

<https://en.wikipedia.org/wiki/MD5#/media/File:CPT-Hashing-File-Transmission.svg> (Zugriff am 18.2.2017), 2012.

Anwendung Passwortspeicherung I

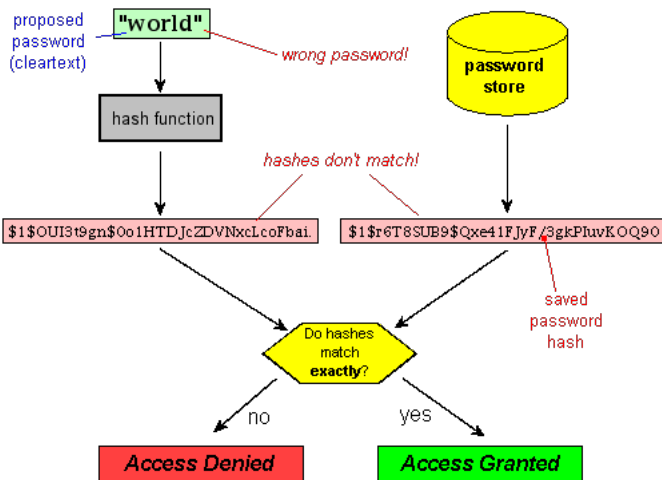
- Passwörter nicht im Klartext speichern
 - Stattdessen Hash von Passwort speichern
- Bei Login Hashvergleich statt Klartextpasswortvergleich



Quelle: Friedl, S.: An Illustrated Guide to Cryptographic Hashes.

<http://www.unixwiz.net/techtips/iguide-crypto-hashes.html> (Zugriff am 18.2.2017), 2005.

Anwendung Passwortspeicherung II



Quelle: Friedl, S.: An Illustrated Guide to Cryptographic Hashes.

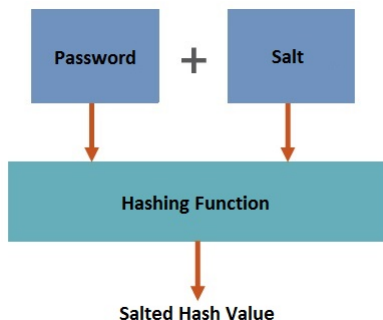
<http://www.unixwiz.net/techtips/iguide-crypto-hashes.html> (Zugriff am 18.2.2017), 2005.

Angriffe auf Passworthashes (Auswahl)

- „Rohe Gewalt“ (engl. *brute force*)
 - Alle Möglichkeiten ausprobieren, bis Übereinstimmung gefunden
 - Geschwindigkeit abhängig von Passwortlänge und Zeichensatz
- Wörterbuchangriff (engl. *dictionary attack*)
 - Überlegung: Häufig verwendete Passwörter (vgl. letzte Vorlesung)
 - Liste von Wörtern durchprobieren
 - Deutlich schneller als „rohe Gewalt“
- Regenbogentabellen (engl. *rainbow tables*)
 - Im Voraus berechnete Liste von Hashes mit dazugehörigen Passwörtern
 - Deutlich schneller als Wörterbuchangriff, aber nicht immer erfolgreich
 - Nachteil: Sehr hoher Speicherbedarf (typischerweise Giga- bis Terabyte)
- Spezialsoftware für Passwortangriffe, z.B. *John the Ripper*

Erschwerung von Passwortangriffen I

- Problem: Gleiches Passwort (mehrerer Benutzer) → gleiche Hashes
- Knacken eines Hashes knackt alle gleichen Passwörter
- Lösung: *Salting*: Benutzerspezifische Datenergänzung vor Hashing







Quelle: Venkadeshwaran, T. S.: Best way to secure password using Cryptographic algorithms in C# .NET.

<http://immortalcoder.blogspot.co.at/2015/11/best-way-to-secure-password-using-cryptographic-algorithms-in-csharp-dotnet.html> (Zugriff am 18.2.2017), 2015.

Erschwerung von Passwortangriffen II

- Gleiches Passwort (mehrerer Benutzer) → verschiedene Hashes
- Angriffe viel aufwändiger, Regenbogentabelle wertlos
- Da Salt gespeichert werden muss → Regenbogentabelle pro Benutzer

				
Password	bob	bob	bob	bob
Salt	-	-	et52ed	ye5sf8
Hash	f4c31aa	f4c31aa	lvn49sa	z32i6t0

Quelle: Zeeshan, A. A.: Hashing Passwords using ASP.NET's Crypto Class.

<https://www.codeproject.com/Articles/844722/Hashing-Passwords-using-ASP-NETs-Crypto-Class> (Zugriff am 18.2.2017), 2014.

Fragen?