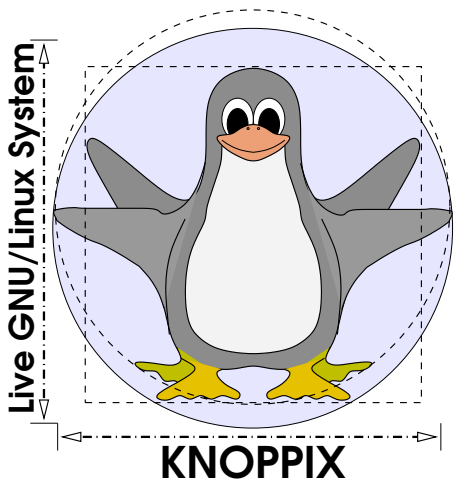
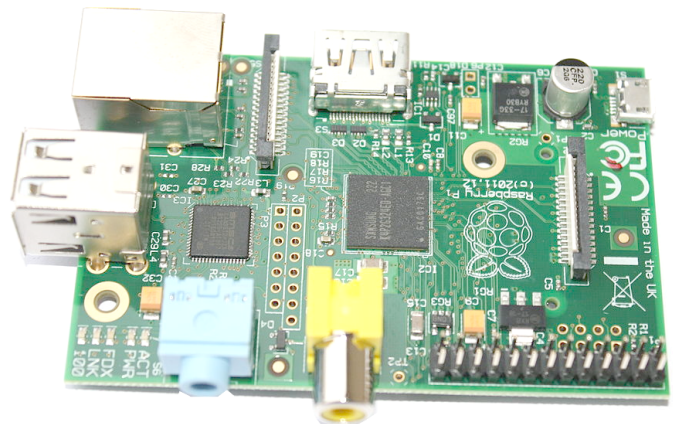
IT-Sicherheit

Prof. Dipl.-Ing. Klaus Knopper





Vorlesung an der Hochschule Kaiserslautern im Wintersemester 2020/2

Organisatorisches

☞ Modul „Recht der Informationstechnologie und IT-Sicherheit“, Teil „IT Sicherheit“, Vorlesung mit Übungen jeweils Freitags 10:30 Uhr in A216.1

Fr. 11.10.2019	Organisatorisches, Einführung
Fr. ...	...
3.2.2020	Klausur

☞ **http://knopper.net/bw/itsec**

Kursziel

- Grundlagen der zu schützenden Güter und Schutzmechanismen kennen lernen,
- Schwachstellen und Angriffspunkte sowie Ansätze zur Verteidigung kennen lernen (praktischer Teil),
- IT-Sicherheit als integraler Bestandteil eines Gesamtkonzeptes verstehen,
- „Best Practice“ Beispiele, Normen und Anleitungen zur IT-Sicherheit kennen.

Zur Benutzung von Folien/Skript

- Der Foliensatz wird nach Bedarf während des Semesters erstellt (Work in Progress). Daher bitte Vorsicht beim Ausdrucken.
- Verweise auf sinnvolle ☞ Sekundärliteratur sind entsprechend gekennzeichnet und i.d.R. direkt anklickbar.
- Prüfungsrelevant (Klausur) sind grundsätzlich alle in der Vorlesung und in den Übungen behandelten Themen.

IT-Sec. und der Business Value of IT (1)

$$\text{BVIT} = \frac{\text{Business Performance}}{\text{IT Investment}}$$

IT Investment: Eine Investition in die **Fähigkeit, ein Geschäft zu führen**

Der **Geschäftserfolg** (Business success, business value, linke Seite der Gleichung) hängt wesentlich davon ab, den Zählerwert in der Wertegleichung (Business Performance) zu erhöhen, nicht (nur) den Wert des Nenners zu reduzieren.

IT-Sec. und der Business Value of IT (2)

Was bedeutet dies für die „IT Security“, die ja als Investition im Nenner steht und sich somit grundsätzlich erst mal „negativ“ auf das Ergebnis auswirkt?

☞ Die **Business Performance** hängt vom **Funktionieren der IT** ab. Ein Ausfall durch unzureichendes IT Investment in die Infrastruktur kann also den Wert des Zählers in verheerender Weise zerstören.

Ziele

- ☞ Vertraulichkeit
- ☞ Integrität
- ☞ Verfügbarkeit

Vertraulichkeit

☞ Vertraulichkeit ist die Eigenschaft einer Nachricht, nur für einen beschränkten Empfängerkreis vorgesehen zu sein. Weitergabe und Veröffentlichung sind nicht erwünscht. Vertraulichkeit wird durch Rechtsnormen geschützt, sie kann auch durch technische Mittel gefördert oder erzwungen werden.

Maßnahmen: ☞ Verschlüsselung, ☞ Digitale Rechteverwaltung (z.B. DRM)

Verfügbarkeit

Die Verfügbarkeit eines technischen Systems ist die **Wahrscheinlichkeit** oder das Maß, dass das System bestimmte Anforderungen zu einem bestimmten Zeitpunkt bzw. innerhalb eines vereinbarten Zeitrahmens erfüllt.

Als Kennzahl:

$$\text{Verfügbarkeit} = \frac{\text{Gesamtzeit} - \text{Ausfallzeit}}{\text{Gesamtzeit}}$$

Auch: Uptime = Gesamtzeit − Ausfallzeit oder „Mean Time between Failure„ (☞ MTBF)

Maßnahmen: Redundanz, Backup, Archivierung, Hot-Standby, ...

Problem: Haltbarkeit aktueller physischer Datenträger!

Integrität

s.a. Wikipedia

Alte Definition: „Verhinderung unautorisierter Modifikation von Information“

Bundesamt für Sicherheit in der Informationstechnik (BSI): „Korrektheit (Unversehrtheit) von Daten und der korrekten Funktionsweise von Systemen“ (weiter gefasst)

Kriterien: **Korrekter Inhalt**, **Unmodifizierter Zustand** bzw. die **Möglichkeit, Modifikationen zu erkennen und zuordnen zu können**

Maßnahmen: ☞ **Prüfsummen**, ☞ **Digitale Signatur**

Verschlüsselung und Signatur

Ziel Verschlüsselung: Nur **berechtigten Personen** den Zugriff auf die Daten ermöglichen (Vertraulichkeit)
Ziel Signatur: Beweis, dass ein Dokument oder eine Datei **unverändert, wie von den Autoren veröffentlicht** ist, und nicht manipuliert wurde. (Authentizität)

Mittel: Anwendung (möglichst sicherer) mathematisch-algorithmischer Verfahren.

- Symmetrische Verfahren (11)
- Asymmetrische Verfahren (16)

Symmetrische Verschlüsselung

Der Caesar-Algorithmus ist eines der einfachsten (und unsicheren) Verfahren, um Verschlüsselung zu zeigen.

```
if(eingabe >= 'A' && eingabe <= 'M')
  ausgabe = eingabe + 13;
else if(eingabe >= 'N' && eingabe <= 'Z')
  ausgabe = eingabe - 13;
else
  ausgabe = eingabe;
```

XOR

Ein einfacher, auf Computern besonder effizient einsetzbarer Verschlüsselungs-Algorithmus, ist das bitweise **Exklusiv Oder**, dessen Anwendung ebenfalls umkehrbar ist:

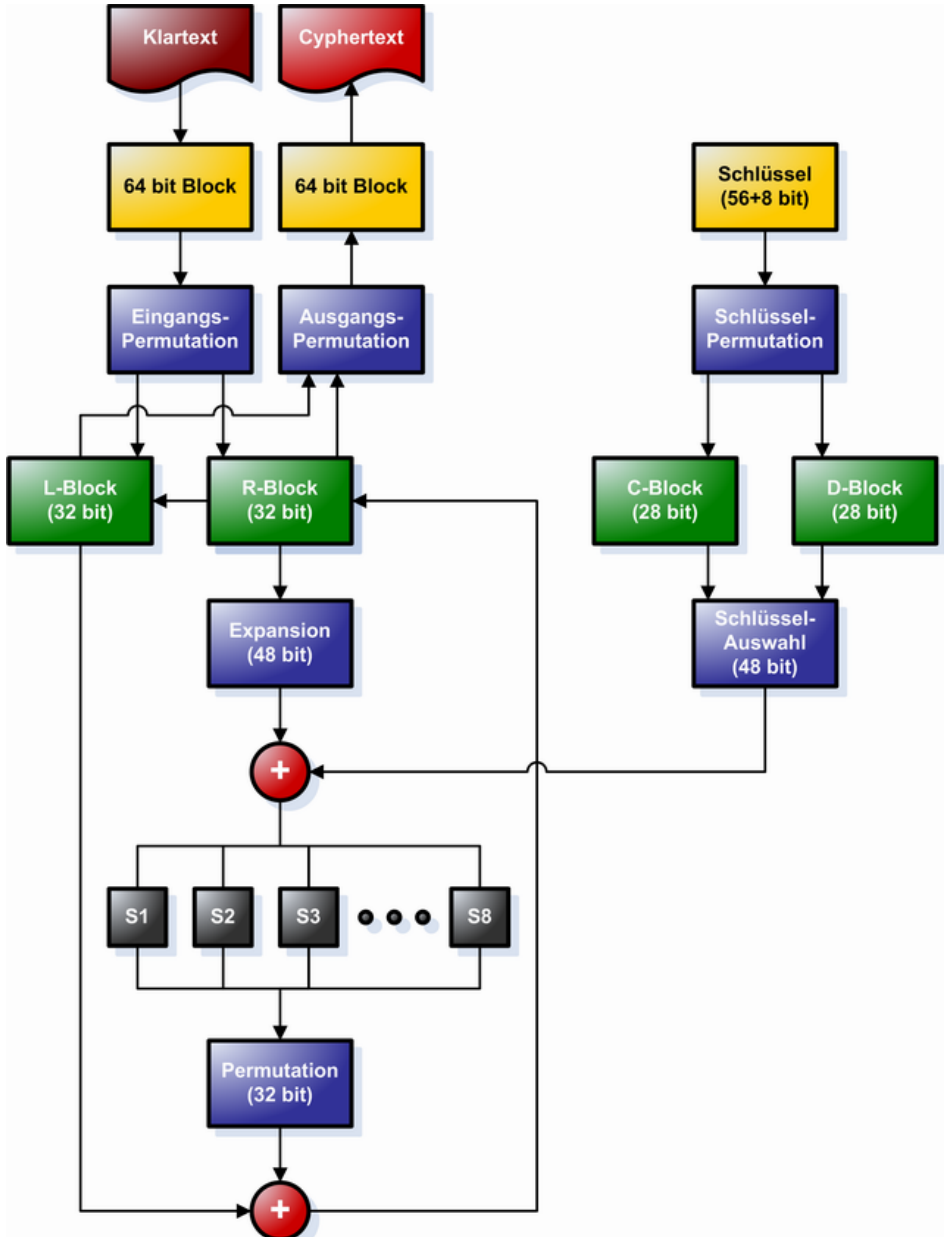
a	**b**	**a** $\oplus$ **b**
0	0	0
1	0	1
0	1	1
1	1	0

Block Cipher (1)

Da die bitweise Anwendung immer in Blöcken mit „sinnvoller Größe“ durchgeführt wird, (ggf. Größe eines Prozessor-Registers), spricht man von einer Blockchiffre (Block Cipher).
Beispiel (Pseudocode) XOR:

```
u_32   schluessel=0xabcd;
u_32[] data = "Hallo, Welt ...";
for(int i=0; i < sizeof(data) / sizeof(u_32); i++)
 data[i] =  data[i] ^ schluessel;
```

DES


Klartext
Cyphertext
64 bit Block
64 bit Block
Schlüssel (56+8 bit)
Eingangs-Permutation
Ausgangs-Permutation
Schlüssel-Permutation
L-Block (32 bit)
R-Block (32 bit)
C-Block (28 bit)
D-Block (28 bit)
Expansion (48 bit)
Schlüssel-Auswahl (48 bit)
+
S1
S2
S3
S8
Permutation (32 bit)
+

AES

Der Advanced Encryption Standard wurde als DES-Nachfolger in einer lizenzkostenfreien Implementierung von Joan Daemen und Vincent Rijmen erfunden (auch „Rijndael-Algorithmus“ genannt). Er ist heute Standard in den meisten Crypto-Produkten und wird auch bei der Datenübertragung im Internet (https, WPA etc.) eingesetzt.

Problem bei der symm. Verschlüsselung

Während die Verschlüsselung an sich bei modernen Verfahren wie AES als sicher gelten kann, ist die sichere Aufbewahrung und Verteilung der Schlüssel ein großes Problem.

Bei bekannt werden des symmetrischen Schlüssels können alle damit verschlüsselten Dokumente entschlüsselt werden.

Zwei Personen, die verschlüsselt kommunizieren wollen, stehen somit zunächst vor dem Problem, den Schlüssel auf sichere Weise zu erhalten.

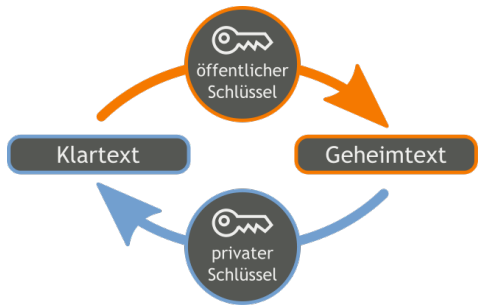
Zwei komplementäre Schlüssel

Bei der asymmetrischen Verschlüsselung werden zwei sich gegenseitig ergänzende (komplementäre) Schlüssel generiert. Daten, die mit einem der Schlüssel verschlüsselt werden, können NICHT mit dem gleichen Schlüssel wieder entschlüsselt werden, sondern nur mit dem anderen.

Einer der Schlüssel wird geschützt aufbewahrt (z.B. in einer symmetrisch verschlüsselten Datei) und ist nur dem Besitzer bekannt. ☞ **Secret Key** (mitunter auch als „Private Key" bezeichnet).

Der anderen Schlüssel wird an die Kommunikationspartner verteilt, und muss nicht besonders geschützt werden. ☞ **Public Key** (Öffentlicher Schlüssel).

Ablauf der sicheren Kommunikation

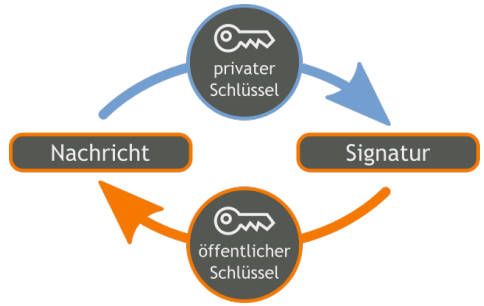


1. Eine Nachricht, die sicher übertragen werden soll, wird mit dem Öffentlichen Schlüssel des beabsichtigten Empfängers (der dem Sender bekannt ist) verschlüsselt.

2. Die verschlüsselte Nachricht wird über einen, möglicherweise „unsicheren" Kanal (jeder kann sie kopieren) an den Empfänger übertragen. Mit Hilfe des Öffentlichen Schlüssels kann sie jedoch nicht wieder lesbar gemacht werden.

3. Der Empfänger, welcher den passenden Privaten Schlüssel besitzt, kann die Nachricht lesen.

Ablauf der Verifikation von Daten



1. Der Urheber eines digitalen Dokumentes bildet eine Prüfsumme (Hash) über das Dokument.
2. Er verschlüsselt diese Prüfsumme, diesmal aber mit seinem Geheimen Schlüssel, der nur ihm selbst zugänglich ist.
3. Das Dokument wird zusammen mit der verschlüsselten Prüfsumme an die beabsichtigten Empfänger verschickt (unverschlüsselt oder nicht, spielt dabei keine Rolle).
4. Die Empfänger können mit Hilfe des ihnen bekannten Öffentlichen Schlüssels des Dokumenten-Urhebers die Prüfsumme dechiffrieren, und überprüfen, ob sie noch zum Dokument passt. Ist das nicht der Fall, wurde das Dokument manipuliert. Ansonsten ist es authentisch.

Erzeugung und Benutzung

... eines Asymmetrischen Schlüsselpaars: Siehe Skript.

Die Anwendung der Modulo-Operation (Teilungsrest) innerhalb des Algorithmus sorgt dafür, dass bestimmte Schritte nicht umkehrbar sind! Mit z.B. dem Teilungsrest „9" bei „X Modulo 10" kann, offensichtlich, nicht die Zahl X berechnet werden. Hierdurch ist es unmöglich, zu einem Öffentlichen Schlüssel den Privaten Schlüssel exakt zu berechnen.

Es gibt, aus dem gleichen Grund, mathematisch gesehen, unendlich viele Private Schlüssel, die genauso komplementär zu einem Öffentlichen Schlüssel sind wie der ursprünglich generierte, allerdings sind sie bei der erforderlichen hohen Schlüssellänge praktisch unmöglich durch „Ausprobieren" zu finden.

Angriffsvektor bei Asymm. Verschlüsselung

Der Öffentliche Schlüssel kann zwar problemlos über ein unsicheres Netzwerk übertragfen werden, da man mit ihm die verschlüsselten Daten nicht wieder entschlüsseln kann, jedoch kann ein Angreifer versuchen, mit falscher Identität einen Öffentlichen Schlüssel für jemand anders zu propagieren, zu dem er selbst jedoch den passenden geheimen Schlüssel besitzt.

Lösungsansatz: Schlüssel von einer „vertrauenswürdigen Instanz" signieren lassen, deren Öffentlicher Schlüssel unwiderlegbar authentisch ist. (Ist das so? ☞ Liste von vertrauenswürdigen Signierern in Firefox)

SSL-Zertifikat

...ist nichts anderes als eine Datei mit

1. Öffentlicher Schlüssel,
2. weitere Informationen wie Name, Mailadresse, Webseite, Postadresse, Geodaten, ...
3. digital signiert von einer „vertrauenswürdigen Instanz" (oder, im einfachsten Fall, vom Besitzer selbst mit Hilfe seines NICHT im ARCHIV ENTHALTETEN Privaten Schlüssels).

Eine Veränderung von Daten innerhalb des Zertifikats führt dazu, dass sich ihre Prüfsumme ändert und damit die digitale Signatur hinfällig wird.

Das Zertifikat wird, wie der Offentliche Schlüssel, an alle Kontakte verteilt.

Zu jedem Zertifikat muss der Eigentümer den Privaten Schlüssel separat gut gesichert aufbewahren.

Softwaretechnik

- openssl (Secure Socket Layer)
  - Webserver (https)
  - Mail-Server (smtps, pop3s, imaps)
  - Mail-Clients (S/MIME)
  - Datei- und Archiv-Manager mit Verschlüsselung
  - Access Points (WPA, WPA Enterprise)
  - div. Server: SAMBA, LDAP, MYSQL, SSH
- PGP (Pretty Good Privacy)
  - Mail-Programme (PGP-MIME)
  - Dateien verschlüsseln

Best Practice

Der Industriestandard ist ☞ SSL (Secure Socket Layer) bzw. TLS (Transport Layer Security) mit ☞ OpenSSL als Open Source Standard Implementierung.

Die OpenSSL Libraries, die in fast allen Krypto-Produkten verwendet werden, werden ständig aktualisiert und fehlerbereinigt. Einige Versionen haben allerdings designbedingte Fehler und werden daher nicht mehr eingesetzt (d.h. z.B. im Webserver abgeschaltet, Verbindungen mit angreifbaren Verschlüsselungsalgorithmen oder Fehlern im Schlüsselaustausch werden abgewiesen).

Aufgrund der Komplexität und der vielfältigen Angriffsszenarien im Kryptobereich gilt es als fahrlässig, eine eigene, proprietäre Verschlüsselung in kritischen Bereichen zu implementieren und einzusetzen!

Intermezzo: Aus aktuellem Anlass...

Angriff auf WPA2 (WLAN): Was ist dran? Risiken? Was muss man tun?

Tabelle der angenommenen und tatsächlichen Angriffsszenarien!

S.a. separates ☞ Handout zu „WPA2 Krack Attack“.

Übungen zu Verschlüsselung und Signatur

Ziele:

1. Fähigkeit, eine persönliche Public-/Secret Key Infrastruktur einzurichten.
2. Fertigkeit erlernen, auf Datei-Ebene Verschlüsselung und Signatur „Low Level" durchzuführen.
3. ☞ Übertragung auf Software-Produkte wie Mailprogramm, Office, Browser, Zugangs-Software.

Mittel:

1. PGP (Pretty Good Privacy) bzw. GnuPG (Open Source Version),
2. OpenSSL (Secure Socket Layer Frontend-Programme)

Übungen: GnuPG

Website / Download: **`https://www.gnupg.org/`**

Aufgaben: Im Vorlesungs-Skript!

Übungen: OpenSSL

Website: **`https://www.openssl.org/`**
Binary Downloads:
`https://wiki.openssl.org/index.php/Binaries`

Aufgaben: Im Vorlesungs-Skript!

Fazit PGP und OpenSSL

Zwei Verfahren für die gleiche Funktionalität?

Obwohl PGP historisch gesehen die „ältere" Software ist, hat sich SSL v.a. bei kommerziellen Produkten durchgesetzt. Dies mag auch mit der Tatsache zu tun haben, dass „Secure Socket Layer" sich speziell für Web-Anwendungen und auch Streaming von Inhalten durchgesetzt hat, und das x509-Format für die Zertifikate auch nützliche und erweiterbare Meta-Informationen enthält, die in PGP Public Keys nicht vorgesehen waren. Mit ☞ S/MIME existiert ein auf SSL basierender Standard für Mail-Attachments, während ☞ PGP/MIME nicht standardisiert ist (obwohl bei vielen Mailern als Plugin implementiert).
Die im ☞ PKCS#12 Format gespeicherten SSL Public/Private-Keypaare lassen sich in den meisten Mailprogrammen und Browsern unter „Zertifikat-Verwaltung" importieren (mit Passworteingabe), ebenso wie die Zertifikate (ohne Passwort).

Speziell für die Verschlüsselung und Signatur von E-Mail sind PGP und S/MIME noch beinahe paritätisch vertreten, und viele Programme unterstützen, ggf. über Plugins, beide Methoden.

„Festplatten“-Verschlüsselung

... soll v.a. sensitive Daten bei Diebstahl des Gerätes bzw. Datenspeichers (Smartphone mit privaten/Firmen-Daten, Notebook, auch PCs) vor Ausspähen schützen (Thema Vertraulichkeit).

- Datei-, Partitions- oder Datenträger-weise Verschlüsselung,
- i.d.R. symmetrische Blockcipher,
- der Schlüssel kann ein Passwort sein (Länge vs. Komfort, Sicherheit vs. Merkbarkeit), oder ein Binärschlüssel mit Zufallsanteil bei der Generierung, der selbst wieder durch ein Passwort geschützt in einem separaten Datenbereich gespeichert wird,
- verschiedene Software verfügbar, teils plattformneutral, teils OS-spezifisch.

Beispiel „Android“-Verschlüsselung (1)

- „Full Disk Encryption“ gehört seit Android 5 zur (optional in den Einstellungen unter „Sicherheit“ aktivierbaren) Standard-Ausstattung,
- bedeutet: „die **`/data`**-Partition wird verschlüsselt“,
- symmetrische, AES-basierte Blockverschlüsselung mit 128Bit,
- der „Device Encryption Key“ (DEK) wird in einem separaten Bereich auf dem Gerät mit einer PIN oder Passwort verschlüsselt gespeichert („KeyMaster“),
- bei „Hardware Encryption“ übernimmt ein Teil des Chipsatzes („TrustZone“, von der der KeyMaster ein Teil ist) die Ver- und Entschlüsselungsalgorithmen quasi als „Blackbox“, von außen über eine API von signierten Applikationen gesteuert,
- Angriffspunkt: Vom Hersteller signierte Apps können die „TrustZone“ zum Auslesen des DEK, Ver- und Entschlüsseln

Beispiel „Android"-Verschlüsselung (2)

- Mit dem Android Debugger (ADB) können die Daten vom dem Gerät über USB-Kabel oder Netzwerk gesichert werden, so lange die verschlüsselte(n) Partition(en) entsperrt sind. Das Backup sollte natürlich auch wieder durch eine geeignete Maßnahme (z.B. **`pgp`** oder AES) geschützt werden.
- Im ausgeschalteten Zustand oder bei Sperre des Gerätes muss der DEK erst wieder durch Entschlüsseln aktiviert werden.
- Ein „Umverschlüsseln" oder eine komplete Entschlüsselung mit Zurückspeichern wird softwareseitig bislang nicht unterstützt. Es ist aber relativ leicht, die PIN oder das Passwort, mit der/dem der DEK selbst verschlüsselt wird, auszutauschen (unter Kenntnis der alten PIN).

Beispiel „Festplatten“-Verschlüsselung (1)

Übersicht:
`https://de.wikipedia.org/wiki/Festplattenverschlüsselung`

Produkte (OS-spezifisch):

- Windows: ☞ EFS, ☞ BitLocker,
- Mac: ☞ FileVault
- Linux: dm-crypt (auch die bei Android verwendete Variante, das **`/data`**-Block Device (Flash-Partition) transparent zu ver-/entschlüsseln)

Beispiel „Festplatten“-Verschlüsselung (2)

Produkte (Multi-/Cross-plattform):

- Container-Formate mit Verschlüsselungs-Unterstützung, z.B. ☞ 7Zip,
- ☞ CrossCrypt: Zugriff auf verschlüsselte Linux Loop-AES-Partitionen unter Windows,
- VeraCrypt (Nachfolger von ☞ TrueCrypt
- Weitere, teils proprietäre Produkte:

  https://de.wikipedia.org/wiki/Festplattenverschl%C3%BCsselung#Software

Beispiel „Festplatten"-Verschlüsselung (3)

In Ländern, in denen Kryptographie verboten ist und um - falschen oder berechtigten - Verdächtigungen zu entkommen, ist bei Verschlüsselungsmechanismen der Begriff der „plausible deniability„ (Glaubhafte Abstreitbarkeit ein Sicherheits- und Qualitätskriterium. Dies bedeutet in Bezug auf die Festplattenverschlüsselung, dass ohne Kenntnis des Schlüssels nicht bewiesen werden kann, ob und welche Daten auf einer verschlüsselten Partition vorhanden sind, und nicht einmal, ob überhaupt verschlüsselt wurde. Der Besitzer kann also „glaubhaft bestreiten", im Besitz sensitiver Daten zu sein, was bei entsprechend geeigneten Mechanismen auch nicht widerlegt werden kann.

Angriffspunkte „Festplatten“-Verschlüsselung

Angriff	**Verteidigung**
Ausspähen des symmetrischen Schlüssels	Nur „im Kopf“ speichern oder Schlüssel sicher verwahren (z.B. verschlüsselter Container ☞ keepassX)
Erraten durch „Brute Force“ (Ausprobieren)	Hohe Schlüssellängen, keine einfachen Passwörter mit „Wörterbücher“-Anteilen, keine „gelben Zettel“ am Monitor aufkleben.
Abgreifen der Daten vor Verschlüsselung bzw. nach Entschlüsselung	Vermeiden von längerfristigen unverschlüsselten „Zwischenspeichern“, möglichst direkt verschlüsselt speichern und erst beim Lesen entschlüsseln

Datei- vs. Geräteverschlüsselung

Bei dateiweiser Verschlüsselung kann für jeden Ordner oder jede Datei ein separater Schlüssel verwendet werden. Problem: Wie und wo werden diese Schlüssel gespeichert? Angriffspunkte: „Klartext-Attacke", durch Dateinamen und andere Metadaten wie Größe und Position im Dateisystembaum kann auf den Inhalt einer Datei geschlossen werden, was „Brute Force"-Attacken erleichtert. Zum Zugriff auf eine Datei wird bei einigen Algorithmen oft eine unverschlüsselte Version der Datei im Speicher oder temporären Bereichen der Festplatte gehalten, um den Aufwand für die transparente Dechiffrierung gering zu halten, dort wären die Daten auslesbar.

Bei Geräteweiser (=Datenträger bzw. Partition oder Container-Image) Verschlüsselung wird der gesamte verschlüsselte Datenbereich blockweise verschlüsselt, und muss „als Ganzes" aufgeschlossen werden. Es wird immer der gerade gelesene Block dechiffriert, und nur so viel unverschlüsselt im Speicher gehalten, wie von der Anwendung benötigt wird. Die Komplexität der Algorithmen ist geringer als bei der dateiweisen Verschlüsselung. Allerdings sind bei Verlust des Schlüssels weder die Dateisystem-Struktur, noch die Dateien selbst wiederherstellbar. Nicht

Löschen von Daten?

Problem:

- System calls wie **`unlink(char *filename)`**... entfernen nur Einträge aus dem Inhaltsverzeichnis des Dateisystems. Gleiches gilt für „formatieren".
- Datensegmente einer Datei bleiben i.d.R. erhalten (ggf. über den Datenträger „verstreut",
- Auch ältere Versionen der Dateien bleiben erhalten,
- „Überschreiben" wird vom Betriebssystem meist so gehandhabt, dass eine NEUE Datei angelegt wird, und die alte lediglich als „gelöscht" markiert wird (d.h. der Platz ist wieder nutzbar, sobalt notwendig).

Lösungsansatz?

Sicheres Löschen von Daten

- Dateisysteme, die ein „in-place“ überschreiben der physikalischen Sektoren unterstützen,
- Datei erst „normal“ löschen, dann den „freien Platz“ des Datenträgers komplett überschreiben (durch eine riesige Datei),
- Datenträger von vornherein verschlüsseln, und den symmetrischen Schlüssel verwerfen, wenn der Datenträger außer Betrieb genommen wird (Achtung: Brute Force offline-Entschlüsselung ggf. eine Angriffsmöglichkeit)

Forensik (1)

= gelöschte oder versteckte Daten (teilweise) wiederherstellen.

Bei versehentlicher Datenlöschung, Korruption des Dateisystems durch Schadsoftware oder Hardwarefehlern, die ein „normales“ Lesen der Dateien verhindern, kommt spezialisierte Software zum Einsatz. Auch Forensiker, die Daten von hastig formatierten Datenträgern im Auftrag der Strafverfolgung wiederherstellen oder Ursachenforschung betreiben, setzen diese Tools unter Linux oder Windows gerne ein:

- ☞ **`testdisk`**: Wiederherstellung von Partitionstabellen und „gelöschter“ Dateien,
- ☞ **`photorec`** (vom gleichen Entwickler): Teilweise oder komplette Wiederherstellung von Dateien (Data Carving) auch dann, wenn die Dateisystemstruktur nicht mehr zu retten ist, durch Suche nach Dateisignaturen. Hierdurch sind i.d.R. nur die Inhalte der Dateien unter neuem Namen zu retten, nicht die ursprünglichen Dateinamen die im

Forensik (2)

- ☞ **ddrescue** (Linux only): 1:1 Kopie eines Datenträgers in eine Image-Datei anfertigen, dabei Ersetzen von physischen Lesefehlern (zerstörte Sektoren) durch Nullen, um die Position der Daten zu erhalten für anschließende Datenrettung aus der Kopie.

Sicherheit im Netzwerk

Angriff auf den Datentransport und Verteidigungsmechanismen.

- Wiederholung Grundlagen ISO/OSI-Modell und TCP/IP (V4 und V6), Intra- und Internet, mit Fokus auf Schwachstellen,
- Netzwerk-Architektur und Netzwerktopologie unter Sicherheits-Aspekten,
- beispielhafte Maßnahmen wie Firewall (iptables), VLAN, 802.1x, ipsec, ...

ISO/OSI

☞ Das 7-Schichten Modell beim Datentransport über Netzwerke
NB: Auf den ☞ OSI-Layer 8 wird speziell bei Sicherheitsthemen auch oft eingegengen, man könnte dies als Zusammenfassung der „nicht-technischen Problemstellungen“ im Modell interpretieren. ;-)

TCP/IP

☞ TCP/IP

➪ Entspricht nur teilweise dem ISO/OSI-Modell,

TCP/IP Adressnotation

☞ s.a. Wikipedia-Artikel über das IP-Adressschema

Adressen (IPV4): Vier 8 Bit Dezimal-Zahlen, durch Punkte getrennt, z.B. **`10.20.30.40`**.
$\Longrightarrow$ 4 Milliarden ($4 \cdot 10^9$) IP-Adressen

Adressen (IPV6): Acht 16 Bit Hexadezimal-Zahlen, durch Doppelpunkte getrennt, z.B. **`2001:0db8:85a3:08d3:1319:8a2e:0370:7344`**
$\Longrightarrow$ 340 Sextillionen ($340 \cdot 10^{36}$) IP-Adressen

Obwohl die IPV4-Adressen inzwischen komplett aufgebraucht sind (Providern und Organisationen zugewiesen), und alle modernen Betriebssysteme auch das IPV6-Protokoll beherrschen, sind sie immer noch die am häufigsten (wieder-)verwendeten.

Netzwerkmaske

Durch die Netzwerk (Bit-)maske wird angegeben, welcher Teil einer IP-Adresse für alle Computer im gleichen Netzwerk verwendet wird, die sich direkt erreichen können. Die anderen Bits werden zur Adressiereung der einzelnen Rechner im Netzwerk verwendet.
Beispiel: Netzwerkadresse **`10.20.30.0`** mit Netzmaske **`255.255.255.0`** erlaubt 255 Computer im gleichen Netz,
Netzwerkadresse **`10.20.0.0`** mit Netzmaske **`255.255.0.0`** erlaubt $255 \cdot 255 = 65025$ Computer im gleichen Netz.

Wer vergibt die IP-Adressen?

1. Technisch: ein DHCP-Server (Router, Gateway, s. (48))
2. Organisatorisch/Technisch: Der Netzwerk/Internet-Provier
3. Organisatorisch: ☞ IANA (Internet Assigned Numbers Authority)

Lokal dürfen auch einige „private“ IPV4-Adressen vergeben werden, die nicht direkt im Internet weitergeleitet werden, z.B. **`10.*.*.*`**, **`192.168.*.*`**.

Router (1)

Um Daten über das eigene Netzwerk hinaus zu verschicken, ist ein Computer mit mindestens zwei Netzwerkkarten erforderlich, je eine pro Netzwerk, der die Datenpakete anderer Computer zwischen diesen Netzen weiterleitet ☞ Router.

Bei einem Router, der einen Transfer der Datenpakete in ein anderes Protokoll oder in größere Netzwerkstrukturen durchführt, spricht man von einem ☞ Gateway.

Das ☞ Default Gateway wird auf Computern, die netzwerkübergreifend andere Computer (z.B. via Internet) erreichen sollen, für die (IPV4-) Zieladresse `0.0.0.0` eine Adresse im lokalen Netzwerk definiert, die für alle Weiterleitungen zuständig ist. Dies kann z.B. ein DSL-Router, Accesspoint, oder bei Tethering auch ein Smartphone sein, das seine Internetverbindung „teilt".

Seine zentrale Funktion und die Weiterleitung vieler Datenpakete macht das Default Gateway zu einem beliebten Angriffspunkt.

Router (2)

Unter Unix (Linux, MacOS, ...) kann die aktuelle Routing-Tabelle mit dem Kommando **`route`** ausgegeben werden:

```
$ route -n
Kernel-IP-Routentabelle
Ziel         Router         Genmask         Flags Metric Ref   Use Iface
0.0.0.0      172.16.0.1     0.0.0.0         UG    600    0       0 wlan0
10.3.0.0     10.104.0.1     255.255.255.0   UG    0      0       0 tun0
10.4.0.0     10.104.0.1     255.255.255.0   UG    0      0       0 tun0
10.5.0.0     10.104.0.1     255.255.255.0   UG    0      0       0 tun0
172.16.0.0   0.0.0.0        255.255.0.0     U     600    0       0 wlan0
```

Unter Windows erfüllt das Kommando **`route print`** den gleichen Zweck.

Traceroute

Mit dem Programm **`traceroute`** (Unix) bzw. **`tracert`** (Windows) kann der Weg, den Datenpakete vom eigenen Rechner zum Ziel nehmen, zurückverfolgt werden:

```
$ traceroute www.google.de
traceroute to www.google.de (172.217.20.227), 30 hops max, 60 byte packets
 1  gateway (172.16.0.1)  10.395 ms  10.376 ms  10.338 ms
 2  10.255.255.1 (10.255.255.1)  110.696 ms  110.637 ms  110.653 ms
 3  192.168.64.2 (192.168.64.2)  110.674 ms  110.674 ms  110.668 ms
 4  bb1-euraix.relaix.net (46.183.103.1)  110.656 ms  111.237 ms  114.156 ms
 5  87.128.239.241 (87.128.239.241)  110.581 ms  148.040 ms  171.765 ms
 7  80.150.170.70 (80.150.170.70)  296.608 ms  135.352 ms  135.306 ms
 8  * * *
...
15  108.170.227.199 (108.170.227.199)  140.599 ms 108.170.227.189
    (108.170.227.189) 136.297 ms 108.170.227.199 (108.170.227.199)
      136.247 ms
16  muc11s11-in-f3.1e100.net (172.217.20.227)  135.732 ms  131.655 ms
    130.769 ms
```

Jede der 15 „Zwischenstationen“ in diesem Beispiel hat prinzipiell die Möglichkeit, den Datenstrom vor dem Weitertransport zu lesen oder zu manipulieren!

Nameserver (Domain Name System, DNS)

Der oder die Nameserver, die den Netzwerkteilnehmern bekannt sein müssen (z.B. in der Konfigurationsdatei **`/etc/resolv.conf`** unter Unix), können Computernamen und Internet-Domains wie **`knopper.net`** oder **`www.google.de`** die entsprechenden IP-Adressen zuordnen, ohne die keine Kontaktaufnahme zu diesen Rechnern möglich ist.

Einige der global nutzbaren Nameserver sind aufgrund ihrer einfach zu merkenden IP-Adressen regelrecht „berühmt“: **`8.8.8.8`** und **`9.9.9.9`**.

Aus Performance-Gründen wird auf den Internet-Gateways, auch auf dem heimischen Accesspoint, oft ein „Caching Nameserver“ betrieben, der die Auflösung häufig verwendeter Namen von Websites etc. stark beschleunigt.

Nameserver (Domain Name System, DNS)

Angriffsvektor: Wer den Nameserver kontrolliert oder manipuliea-tr, kann z.B. Webseiten-Anfragen auf eigene Adressen „umlei-ten“ , ohne dass dies in der Adressleiste des Browsers sichtbar ist!

Abhilfe: Protokolle wie **`https`** (Statt **`http`**) unterstützen durch SSL-Zertifikate nicht nur Verschlüsselung, sondern auch Signatur der Antworten auf Anfragen. Hierdurch kann der Browser mit Hilfe der eingebauten Signierer-Zertifikate erkennen, ob die Webseite authentisch ist, oder auf eine falsche Adresse umgeleitet wurde, die den richtigen privaten Schlüssel nicht kennt.

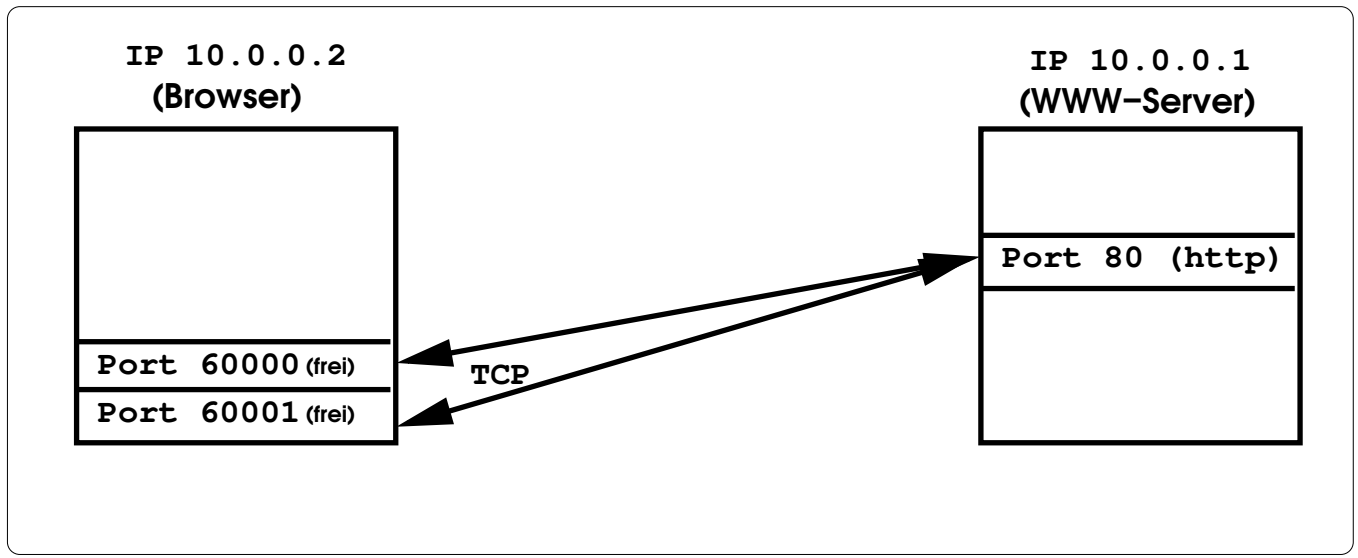
5 Kenngrößen beim Datentransport

Jede Verbindung im Internet hat 5 **eindeutige** Parameter: **`IP-Adresse Quelle`**, **`Port Quelle`**, **`IP-Adresse Ziel`**, **`Port Ziel`** und das Transportprotokoll, entweder **`TCP`** (Transmission Control Protocol, mit Erkennung des Aufbaus und Abbruchs/Ende) oder **`UDP`** (User Datagram Protocol, ein „verbindungsloses" Protokoll, das weder die Reihenfolge, noch überhaupt den EMpfang von Datenpaketen garantiert, aber dafür wenig Overhead besitzt und für die Übertragung von Live-Streams, bei denen es nicht auf jeden Frame ankommt, optimiert ist).

Eindeutigkeit: Es kann zum gleichen Zeitpunkt nie mehrere Datenpakete mit den exakt gleichen 5 Kenngrößen im Internet geben, dies führt zum Verwerfen des „doppelten" Pakets (und ggf. Neuversand zu einem späteren Zeitpunkt)!

5 Kenngrößen bein Datentransport

Beispiel:



Ports, Dienste und Clients

Ein Server-Dienst (hat immer die gleiche Portnummer, z.B. **80** für ☞ **http/www**, **443** für die verschlüsselte Variante ☞ **https**) kann mehrere Clients bedienen (die folglich unterschiedliche Quelle-Portnummer und/oder Quelle-Adressen besitzen), und ein Client kann mehrere Verbindungen aufbauen, die jeweils unterschiedliche Quell-Ports, aber den gleichen Zielport besitzen.

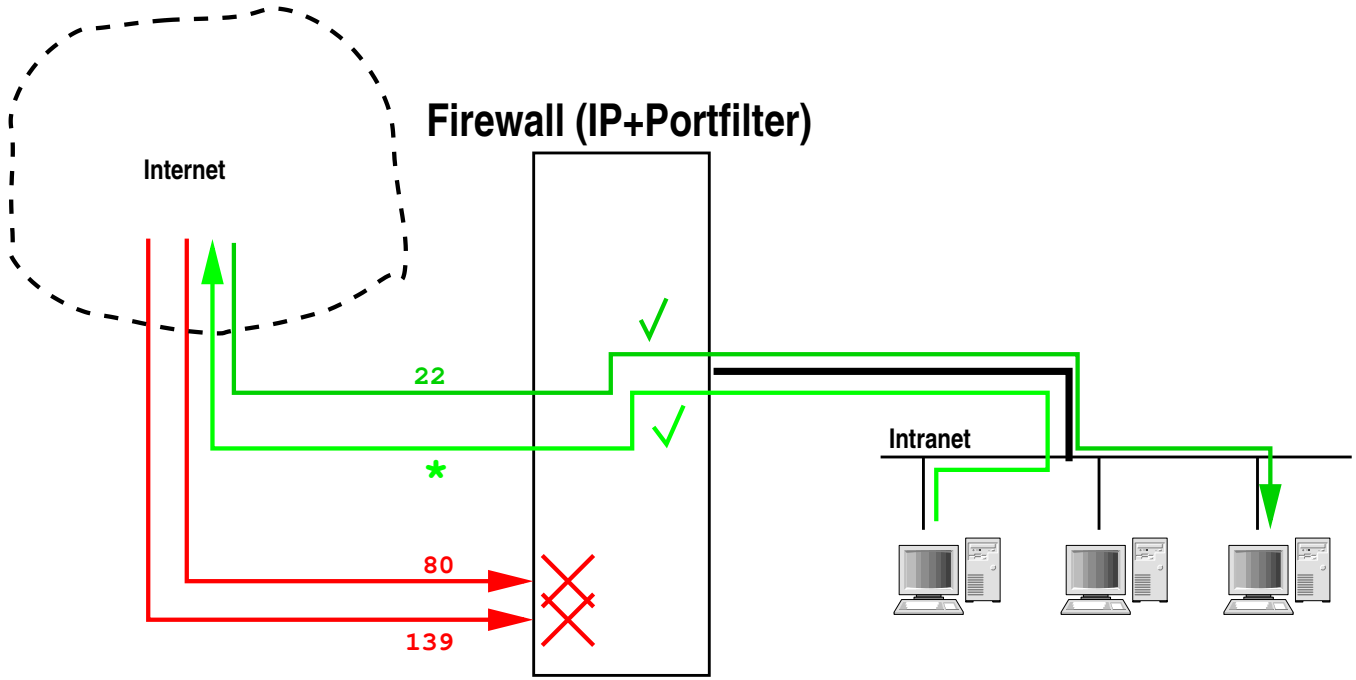
Netzwerke / Netzdienste erkennen (Scanning)

Beispiel Linux / Mac:

- Lokal/Server: **`netstat -tulpen`**
- Remote: **`nmap -P0 -O -sT [-sU] ip-adresse(n)_opfer/netzmaske`**

Firewall

Der (oder die) Firewall filtert nach IP-Adresse, Hardware-Adresse (MAC) oder Port, und kann so z.B. Verbindungen zu bestimmten Computern oder Ports im Zielnetzwerk unterbinden, während andere Adressen oder Ports erlaubt sind.



Forwarding

Ein **Router** soll IP-Pakete weiterleiten, also nicht selber in Empfang nehmen und beantworten. Das Weiterleiten wird als FORWARDING bezeichnet, und unter UNix mit einer entsprechenden Option auf dem virtuellen Netzwerk-Gerät oder global freigeschaltet.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
# Für alle:
iptables -P FORWARD ACCEPT
# Für eine bestimmte Netzwerkkarte:
iptables -I FORWARD -o wlan0 -j ACCEPT
```

Masquerading („Maskieren“)

s.a. ☞ Wikipedia „Port Address Translation“

Eine besonders elegante Form der Network Address Translation ist das „Masquerading“ (Port Address Translation), das bei fast allen Internet-Accesspoints (WLAN oder LAN) angewandt wird.

Hierbei ersetzt der Router die Quell-IP-Adresse aller ins Internet gehenden IP-Pakete durch seine eigene. Bei Rückantworten (folglich an sich selbst) ersetzt er die Ziel-Adresse durch die des Rechners im Intranet, der die Verbindung ursprünglich aufgebaut hat.

So bleibt die Netzwerkstruktur des Intranet vor der Außenwelt verborgen, und dennoch ist bidirektionaler Datenverkehr möglich, allerdings immer nur, wenn der Verbindungsaufbau vom privaten ins öffentliche Netz erfolgte.

(Full) Network Address Translation

s.a. ☞ Wikipedia „Netzwerkübersetzung“

Firewalls können auch eine Umsetzung von Quell- und Zieladressen vornehmen, indem die entsprechenden Segmente eines IP-Paketes „umgeschrieben“ werden. Der Zielrechner „sieht“ dann eine andere Adresse als die des Rechners, der ein Paket tatsächlich verschickt hat. Um eine Antwort weiterzuleiten, muss das gleiche Verfahren dann für die vermeintliche Zieladresse wiederholt werden.

VPN (neu 1)

Ein **Virtual Private Network** soll IP-Pakete zwischen meist örtlich getrennten Netzen weiterleiten, diese Netze also quasi (wie ein normaler Router) **verbinden**, allerdings erfolgt der Datentransport in einer potenziell feindlichen/unsicheren Umgebung (Internet).

Die „unsichere" Strecke wird durch Verschlüsselung in beiden Richtungen überbrückt. Um dies für die Benutzer transparent zu halten, wird in beiden Netzen i.D.R. ein **VPN-Router** integriert, der die Ver- und Entschlüsselung für die anderen Rechner in den beteiligten Netzen transparent (diese merken nichts davon) durchführt.

VPN (neu 2)

Ein häufiger Use Case ist auch, Mitarbeitern Zugriff auf das komplette interne Netzwerk der Firma zu geben, indem sie sich am VPN-Router authentifizieren (Password und/oder SSL-Zertifikate). Das Gegenstück des VPN-Routers auf der Client-Seite ist dann eine Software (bzw. „VPN-Netzwerktreiber"), der direkt auf dem Client-PC läuft, und IP-Pakete per Masquerading in das Intranet „umleitet", wenn sie entsprechende Zieladressen haben.

Umgehen von Firewall-Restriktionen

Manchmal sind Firewalls „unsinnig“ konfiguriert, und erlauben nur unverschlüsselte Kommunikation (z.B. http statt https für WWW, imap statt imaps für E-Mail), so dass Passwörter und private Daten ausspioniert werden könnten.

Abhilfe kann hier das sog. „Firewall Piercing“ schaffen, das mit einer Protokollumsetzung nur vermeintlich den freigeschalteten, unsicheren Dienst auf Port 80 nutzt, in Wirklichkeit aber verschlüsselte Datenpakete in die „Web-Abfragen“ und Antworten packt, die dann auf dem simulierten Web-Server auf dem Zielsystem entpackt und ins Internet geroutet werden.

Beispiel: ☞ httptunnel (Server **`hts`**, Client **`htc`**)

Netzwerk-Pakete „sniffen“

Als ☞ Sniffing oder ☞ Snooping wird das Verfahren bezeichnet, Netzwerk-Pakete abzufangen, zu speichern und zu analysieren, die NICHT für die eigene IP-Adresse als Empfänger beabsichtigt sind.
Passive Sniffer (kaum erkennbar für den Rest des Netzwerkes):

- **`iptraf`**, **`etherape`** Traffic-Analyzer (Portbasiert)
- **`tcpdump`**: Primitiver, aber effizienter IP-Logger

Hierfür wird der „Promiscuous Modus“ der jeweiligen Netzwerkkarte aktiviert, der auch Pakete empfangen kann, die nicht für die eigene IP gedacht sind.

Angriffe auf die Netzwerk-Übertragung (1)

Direkter Einbruch in die Infrastruktur:

- Kompromittierung: Übernahme von Router/Accesspoint durch Missbrauch herstellerseitig installierter Services bzw. Hintertüren (☞ „Backdoor“ (z.B. Fern-Administration über unzureichend geschützte Service-Ports bei Netzwerk-Komponenten),
- Aufstellen eigener Accesspoints mit bekannter SSID aber ohne Authentifizierung/Verschlüsselung, bzw. Mobilfunk-Basen (☞ IMSI-Catcher),

Aktive Sniffer

Häufig eingesetzte „Man in the Middle“ Attacken (auch **aktive** Sniffer, die Routing oder Pakete verändern):

- Umleiten von Paketen im LAN an Hardware-Adressen eigener Rechner per ☞ Arp-Spoofing mit ☞ **`ettercap`**, auch „feindliche Übernahme“ von Verbindungen möglich,
- Mitschneiden des Netzwerk-Datenverkehrs, auch verschlüsselt, zur späteren „Offline“-Auswertung ☞ **`wireshark`**.

Erkennung von Netzwerk-Manipulationen?

- Doppelte Hardware- oder IP-Adressen, häufige Verbindungsabbrüche (**`tcpdump`**-Analyse,
- Protokollanalyse: Wird eine unverschlüsselte Verbindung erzwungen? (Mobilfunk: **`SnoopSnitch`**)

Gefahr durch IoT-Geräte?

- Das „Internet of Things“ erlaubt das das Monitoring, Auslesen und Verarbeiten von Sensordaten, Ein- und Ausschalten sowie Regelung von Parametern komfortabel über das heimische WLAN oder das Internet per App.
- Da sich das IoT-Gerät (z.B. „Smart“-Steckdose mit WLAN und kompatibler App auf dem Smartphone) hinter der heimischen Firewall bzw. dem Maquerading Router (Accesspoint) befindet, sollte es (eigentlich) nicht aus dem Internet scan- oder kontaktierbar.
- Wie kann dann eigentlich die App auf dem Smartphones von unterwegs aus den Strom ein- und ausschalten oder Messwerte abrufen???*)

*) Und warum soll man sich schon bei der Installation der App mit einem Passwort registrieren?

Gefahr durch VPNs und BYOD?

- Jedes Netzwerk ist nur so sicher wie die „schwächste“ Komponente,
- Beim Zusammenschließen von Netzwerken per VPN (auch wenn die Verbindung an sich „sicher“ ist) wird die Anzahl von potenziellen Angriffszielen und Angreifern erhöht.
- Wird erlaubt, eigene Geräte ins Netzwerk zu bringen („Bring Your Own Device“), müssen alle anderen Netzteilnehmer entsprechend gesichert werden, falls es sich bei dem neuen Gerät vielleicht um einen „Angreifer“ handelt. Für dieses Szenario gibt es die „Client Isolation“ als Abhilfe in den meisten Routern/Accesspoints, hierdurch wird Datenaustausch direkt zwischen den Clients unterbunden. Das ist allerdings oft verwirrend oder kontraproduktiv.

Cross Site Scripting Angriffe (XSS)

s.a. ☞ Wikipedia

- (Scheinbare oder echte) Manipulation einer an sich harmlosen Webseite, für den Anwender im Browser nicht erkennbar (Code injection),
- mit oder ohne Ändern von Daten auf dem Server zu diesem Zweck,
- Eigentlich kein direkter Angriff auf den Server, aber Ausnutzen von dessen Vertrauenswürdigkeit, möglicherweise überschreiben Serverseitiger Daten.
- „Unsichere Webformulare“ bzw. Content-Management.Systeme.

Live-Beispiel

... einer an sich harmlosen und einfachen PHP Web-Applikation aus der Softwaretechnik-Vorlesung.

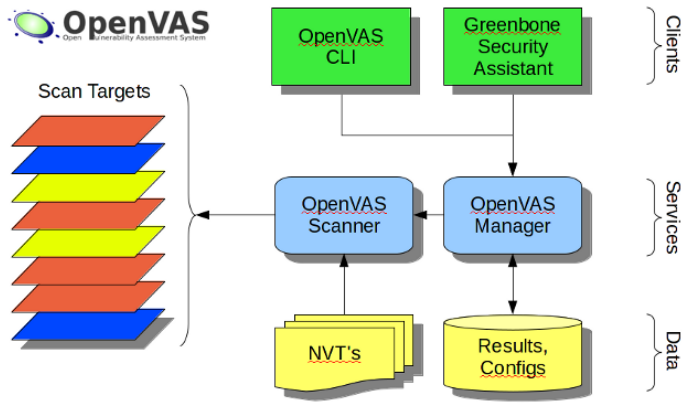
Empfehlungen des BSI

(Bundesamt für Sicherheit in der Informationstechnik)

- Empfehlungen zur „Cyber-Sicherheit“
- Netzwerk-Tool für Sicherheits-Beauftragte: ☞ Nessus Security Scanner (ehemals Open Source, Lizenzänderung durch die Urheber erlaubt nicht mehr die freie Verbreitung ☞ Open Source Fork OpenVAS (nächste Seite)

OpenVAS

Vom ☞ BSI empfohlen: Open Vulnerability Assessment System (OpenVAS)



Weitere Tools

- ☞ BSI: mapWOC
- ☞ OWasp XSS Testing

SPAM / Mailfilter (auch Malware)

Ziel: Entfernen bzw. Quarantäne unerwünschter bzw. gefährlicher Inhalte

Probleme (1):

1. Erkennung von Schadsoftware und unerwünschten Inhalten anhand von Signaturen, die ständig in einer Datenbank aktualisiert werden müssen. „Neue" Schadsoftware kann nur anhand eines angenommenen „Verhaltens" identifizert werden, was eine schwierige Aufgabe ist.

2. Verschlüsselte Inhalte oder solche, die in Archiven mit Passwort (das dem Empfänger über andere Kanäle mitgeteilt wird) untergebracht ist, kann nicht untersucht werden.

3. Überlastung des Filters durch Massenmails und große Anhänge.

SPAM / Mailfilter (auch Malware)

Probleme (1):

1. „False Positives“: Auch ungefährliche / wichtige Inhalte werden fälschlicherweise gefiltert, wenn eine zu einfach gestaltete Sigatur passt und erreichen den Empfänger nicht.

2. Arbeitsaufwand: In Quarantäne geschickte Inhalte müssen manuell untersucht werden, wenn sie nicht generell verworfen werden sollen.

3. Sollen die Absender-Adressen (die sich leicht fälschen lassen) darüber informiert werden, dass ihre E-Mail nicht ankam? Gefahr durch ☞ Backscatter

Lösungsansatz aus der Praxis (1)

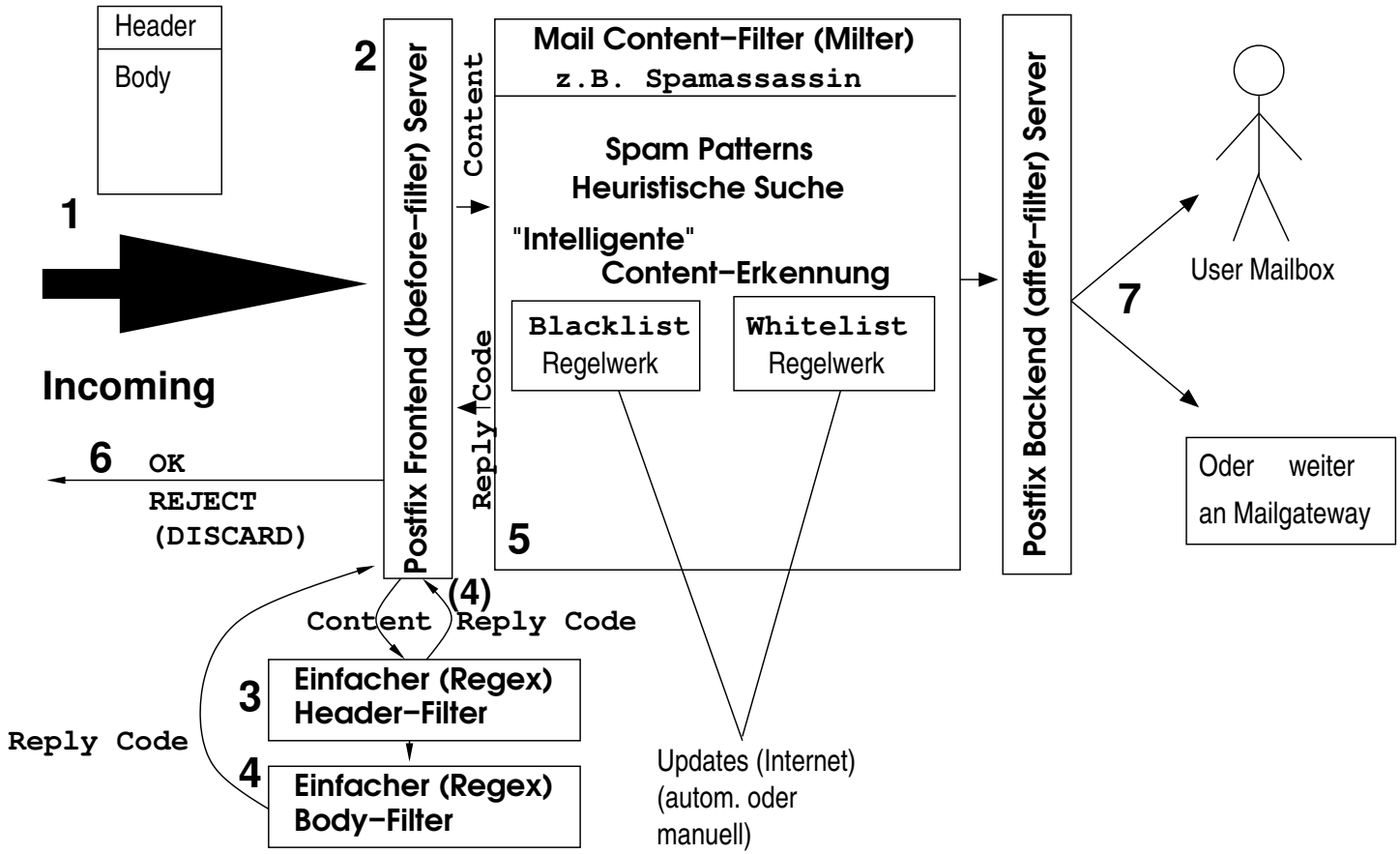
„Sandwich-Konfiguration“ des Nachrichtenfilters:

1. Ein „Frontend“-Server nimmt die Nachricht zunächst an, sendet aber noch keinen Bestätigungs-Code für den „erfolgreichen Empfang“ sondern hält die TCP-Verbindung aufrecht.
2. Ein leistungsfähiger „Backend“-Server (Rechenleistung, Parallelverarbeitung, Speicher, temporärer Speicherplatz, ...) untersucht die Nachricht auf unerwünschte Inhalte oder Malware und meldet den Status der Untersuchung („OK“ oder gefundene Bedrohungen mit Fehlercode) an den Frontend-Server.

   (a) Bei „guten“ Nachrichten stellt der Frontend- (oder Backend-)Server die E-Mail zu, der Frontend-Server meldet einen Erfolgs-Code über die noch bestehende Verbindung zum Server und kann diese trennen (oder auf weitere Nachrichten über die gleiche Verbindung warten).

   (b) Bei „gefährlichen“ oder „unerwünschten“ Nachrichten meldet der Frontend-Server einen Fehlercode über die noch bestehende Verbindung zum Sender und trennt die Verbindung. Die Nachricht wird verworfen oder ein Quarantäne-Verzeichnis verschoben.

Lösungsansatz aus der Praxis (2)



Lösungsansatz aus der Praxis (3)

Beispiel: ☞ Postfix-Mailserver in Sandwich-Konfiguration
Hier: Der Spam- und Malware-Filter läuft zwischen erstem und zweitem Mailserver als sog. „milter“ (Mailfilter), nimmt Mails nur vom Frontend-Server an, der allerdings zuvor schon SMTP-Protokollbasiert und/oder mit Hilfe einfacher „Muster-Filter“ eine erste Zurückweisung unerwünschter Mails vornehmen kann. Der Backend-Server stellt schließlich die als „OK“ durchgegangenen Mails zu oder leitet sie an ein Mailgateway weiter. ☞ Für Technik-Fans: Konfigurationsbeispiel im Ordner **`postfix-config`**

Lösungsansatz aus der Praxis (4)

„Sandwich-Konfiguration“ des Nachrichtenfilters:

Vorteile:

- Der Sender erhält einen Fehlercode, der auch (bei SMTP) eine Klartext-Meldung erhalten kann, und erfährt daher, dass die Nachricht nicht zugestellt wurde.
- Backscatter wird vermieden, da keine Fehler-Nachricht neu generiert und an den ggf. gefälschten Absender aktiv zugesandt wird.

Lösungsansatz aus der Praxis (5)

„Sandwich-Konfiguration“ des Nachrichtenfilters:
Nachteile:

- Der Backend-Server muss leistungsfähig genug sein, dass auch in Zeiten hohen Nachrichtenaufkommens innerhalb einer dienstspezifischen Timeout-Spanne kein Verbindungsabbruch während der Untersuchung der Nachrichteninhalte auftritt.
- Zwei Server notwendig, wodurch die Möglichkeit eines Software-Fehlers erhöht wird, durch den der Nachrichtenversand möglicherweise komplett zum Erliegen kommt.

Intermezzo: Aus (immer noch) aktuellem Anlass...

Sicherheitslücke aus Hardware-Ebene **Meltdown** und **Spectre**, Analyse, Auswirkungen und Folgen: Was ist dran? Risiken? Was muss/kann man tun?

S.a. separates ☞ Handout zu „Meltdown und Spectre“.

Gesamtkonzept

- IT-Sicherheit bei vorhandenen Installationen etablieren (?) oder altes Konzept überarbeiten („mit möglichst wenig Änderungen“ ?)
- IT-Sicherheit als Teil eines Gesamtkonzeptes bei der Planung berücksichtigen
- Rechtslage
- Hilfestellungen
- (kommerzielles) IT-Consulting

HOWTO?

Normen und Zertifizierungen

☞ IT-Sicherheitsmanagement: internationale ☞ ISO/IEC-27000-Reihe

Im deutschsprachigen Raum: ☞ IT-Grundschutz

Evaluierung und Zertifizierung von IT-Produkten und -systemen: ☞ ISO/IEC 15408 (Common Criteria)

Hilfestellungen und Richtlinien

Hierfür ist in Deutschland das Bundesamt für Sicherheit in der Informationstechnik zuständig, das in Form von Sicherheitswarnungen, Analysen, Informationsportalen und teils SEHR umfangreichen Richtlinien-Dokumenten eine beratende Funktion sowohl für Privatanwender als auch für Unternehmen jeder Größe übernimmt. (☞ Gesetzliche Grundlage „BSI-Gesetz“

Allerdings nimmt das BSI keine „Aufträge als IT-Dienstleister“ an, und steht nicht in Konkurrenz zu Firmen aus dem Sicherheitsbereich.

http://bsi.bund.de/

BSI-Dokumente (neue Standards)

- ☞ Grundschutz-Kompendium (Mindmap)
- ☞ BSI Standard 200-1 „Managementsysteme für Informationssicherheit“
- ☞ BSI Standard 200-2 „IT Grundschutz - Vorgehensweise“
- ☞ BSI Standard 200-3 „Risikomanagement“

CERT

Während die bisher genannten Dokumente und Einrichtungen bei der Einrichtung sicherer Infrastrukturen unterstützen sollen, und präventiv wirken, ist bei akuten Vorfällen oft keine Zeit mehr, umfangreiche Dokumente zu lesen.

Das Computer Emergency Response Team ☞ Definition auf Wikipedia ist für schnelle Hilfe, Informations-Weiterreichung und akute Warnungen im Bereich der IT-Sicherheit zuständig. Hierfür gibt es je nach Zielgruppe unterschiedliche Einrichtungen, z.B. ☞ CERT-Bund als Teilbereich des BSI.

Gesetzliche Grundlagen

Während die Grundschutz-Handreichungen des BSI das Unternehmen **freiwillig** darin unterstützen sollen, sich selbst gegen Cyber-Angriffe zu schützen, ist mit dem 2015 verabschiedeten IT-Sicherheits Gesetz eine Verflichtung verbunden, andere / unternehmensexterne Personen und Einrichtungen vor Angriffen zu schützen, die durch die Ausnutzung eigener Sicherheitslücken entstehen können. Hiermit ist die kontrover diskutierte Verpflichtung zu einer mehr oder weniger weit gehenden Protokollierung von Nutzer-Aktivitäten und Netzverbindungen verbunden, z.B. bei ISPs (Internet Service Providern), die die Zugänge ihrer Kunden überwachen und Protokolle eine gewisse Zeit aufbewahren und für Strafverfolgungsbehörden auf Anfrage zugänglich machen müssen.

Auch eine Meldepflicht von sicherheitsrelevanten Vorfällen für Betreiber „kritischer Infrastrukturen" ist Bestandteil des Gesetzes.

☞ IT-Sicherheitsgesetz (mit Zusammenfassung vom BSI) (☞ FAQ)

Anlässlich des 10. Geburtstages von Bitcoin...

Am Anfang war die Blockchain - 10 Jahre Bitcoin (Heise)
Proof-of-Key Day (Behalte die Kontrolle über deine Schlüssel) (BTC-Echo)

Sicherheit von Blockchain-Anwendungen

S.a. separaten Foliensatz „Bitcoin“ als Beispiel für eine *verteilte* Blockchain-Anwendung.
Übersicht:

- Zentral kontrollierte vs. offentliche/verteilte Blockchains,
- Konsens-Verfahren: 51% Attacke bei „Proof of Work“, Angriffszenarien auf das Verfahren an sich (Verfügbarkeit), Replay-Attacken oder Double Spending bei fehlerhaften Algorithmen,
- Risiko: Sichere Aufbewahrung der *geheimen Schlüssel* für die Signatur von Transaktionen

Privater Schlüssel (Signatur)

Gespeichert werden die kritischen privaten bzw. geheimen Schlüssel in eine(m/r) sog. digitalen Wallet („Geldbörse“).

- Sicherheitskriterien bei Erzeugung: Gute Zufallszahlen, hohe Schlüssellänge
- Aufbewahrung: An „sicherem“ Ort, z.B. Offline („Cold Wallet“),
- Maßnahmen gegen Diebstahl:
  - Verschlüsselung der Wallet,
  - Wegschließen (Paper-Wallet z.B. mit ☞ bitaddress.org erzeugt, im Safe oder versteckt)
  - „Auswendig merken“ eines Key oder Master-Seed (bei Hierarchical Deterministic Wallets (HD-Wallet)),
  - Hardware-Wallet: Schlüssel und Signaturalgorithmus in einem „Smart Device“, signiert Transaktionen auch sicher in kompromittierten Umgebungen, allerdings

Authentifizierung: Challenge Response (1)

Hier wird mit Hilfe einer benutzerspezifischen Tabelle ein Challenge Response Verfahren umgesetzt, das auf einer nur dem Besitzer zugänglichen Tabelle oder Rechenvorschrift beruht. Das „Schloss“ gibt bei jedem Öffnungsversuch einen zeitlich begrenzt gültigen Code aus, der mit Hilfe der Rechenvorschrift vom Benutzer in ein gültiges Passwort „übersetzt“ werden muss, welches das Schloss dann öffnet.

Authentifizierung: Challenge Response (2)

Authentifizierung „as a service“: sog. Authentikatoren generieren zeitlich begrenzt gültige „Einmal-Passwörter“, die auf ein dem Benutzer eindeutig zugeordnetem Endgerät generiert werden. Diese Passwörter werden bei der Passwort-Abfrage eingegeben. Der authentigizierte Dienst fragt beim Online-Authentikator nach der Prüfsumme des aktuell gültigen Passworts, stimmt diese überein, ist der Benutzer für die Session authentifiziert.
Da dieses Verfahren bei Diebstahl des passwort-generierenden Endgerätes sehr schwach ist, wird er oft als ergänznde Authentifikation in einem Multi-Faktor Authentifikationsverfahren verwendet.

Zwei-Faktor / Mehr-Faktor Authentifizierung

Hierunter versteht man die erforderliche Kombination aus mindestens zwei verschiedenen aus einem Set von *n* möglichen Authentifikationsmechanismen, die alle erfolgreich sein müssen, damit die Authentifikation erfolgt, z.B. die Kombination aus

1. Einfaches Passwort,
2. Transaktionsnummer, die per SMS oder E-Mail an den Benutzer verschickt wird.

Beispiele: Online-Banking mit mTAN, Zugang zu Online-Börsen.
Vorteil: Zusätzlicher Schutz,
Nachteil: Aufwändiger, kein Zugang bei Verlust der komplementären Authentifikation.

Post Mortem: Schadsoftware gefunden!

Erste Maßnahmen:

- Rechner vom Netz trennen (Intranet+Internet),
- Nicht ausschalten (es könnte sein, dass man sich nicht mehr anmelden kann, ☞ Ransomware, Verschlüsselungs-/Erpressungs-Trojaner)
- Datensicherung aller RELEVANTEN Daten auf externen Datenträger,
- Kontrolle, dass die gesicherten Daten sich auch noch (auf einem anderen Computer) lesen lassen.

Bereinigung vs. Neuinstallation (1)

Bereinigung („Virenscanner“ , Quarantäne)

- ➪ Vorteil: Sind nur wenige, eindeutog identifizierbare Komponenten kompromittiert, ist nach deren Bereinigung das System wieder „wie gewohnt“ verfügbar.
- ➪ Nachteil: Da die Schadsoftware meist neuer ist als die Schadsoftware-Datenbank, ist es leicht möglich, infizierte Dateien zu übersehen ☞ Neuinfektion.
- ➪ Nachteil: Desinfektion erfordert „sauberes“ System (z.B. Live-Stick booten), da infizierte Dateien „gelockt“ oder unsichtbar sein können.

☞ Es ist sehr arbeitsaufwändig und erfordert viel Fachwissen und Recherche, um alle potenziellen kompromittierten Dateien, Bootloader, Systemkomponenten zu finden und durch „saubere“ Versionen zu ersetzen. Kommerzielle Schadsoftware-Scanner enthalten zwar oft sinnvolle Algorithmen, um dies zu teilautoma-

Bereinigung vs. Neuinstallation (2)

Neuinstallation

- Vorteil: Nach Löschen aller System- und Anwenderdateien und Neuinstallation von einem „sauberen“ Datenträger nebst Updates ist das System zunächst in einem einwandfreien Zustand.
- Vorteil: Das System kann bei dieser Gelegenheit auch fehlerbereinigt/aktualisiert werden.
- Nachteil: Die anschließend restaurierten Nutzerdaten könnten, trotz Virenscanner, noch Schadsoftware enthalten und das System erneut kompromittieren.
- Nachteil: Nach Neuinstallation sind ggf. Konfigurationsdaten, Zugangspasswörter etc. „vergessen“ und müssen neu eingetragen werden.

Aktuelle Meldungen hierzu

- ☞ Masterschlüssel für FilesLocker
- ...

Übung (18.1.2019): Stehlen von Bitcoins

Vorbereitung: Kurz-Einführung in Linux und TCP/IP, Werkzeuge **`nmap`**, **`putty`**.