Welcome to

4. Encrypting the Network

Communication and Network Security 2024

Henrik Kramselund he/him han/ham hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Codeberg

4-Encrypting-the-Network-Layer.tex in the repo security-courses

Goals for today

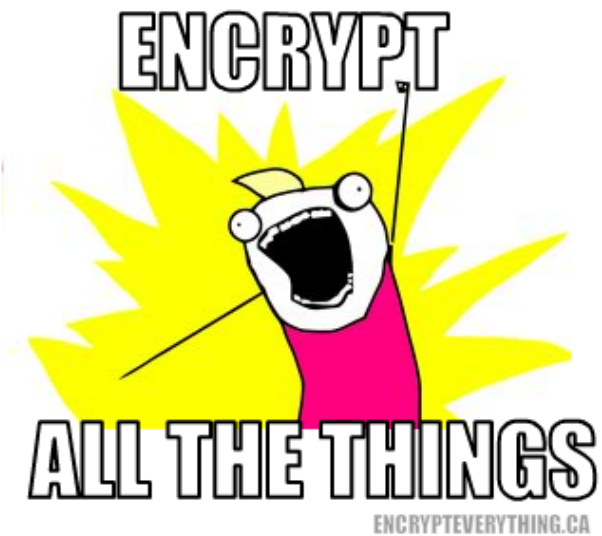
- Introduce Encryption
- Present the common algorithms, protocols, and tools used
- Start focus on various sub projects related to encryption in organisations

Photo by Thomas Galler on Unsplash

Plan for today

Subjects

- Basic cryptography - Encryption Decryption - Hashing
- Symmetric Cryptosystems
- Data Encryption Standard (DES) / Advanced Encryption Standard (AES)
- Public Key Cryptography
- Stream and Block Ciphers
- Example cryptosystems OpenPGP, IPsec, Transport Layer Security (TLS)
- Authentication and Password security, NIST guidelines
- Short introduction to algorithms RSA, AES
- Diffie Hellman exchange and Transport Layer Security (TLS)



Exercises

- sslscan scan various sites for TLS settings, Qualys SSLLabs https://www.ssllabs.com/ and sslscan locally on Kali
- Try Nmap with TLS scans
- Try ssl scanners, similar to ssh scanners
- Discuss HTTPS/TLS with Nginx as example

Time schedule

- 17:00 - 18:15
  Introduction and basics
- 30min break
- 18:45 - 19:30
- 15min break
- 19:45 -20:30 45min

Reading Summary

> It is not clear that the link layer is the right one for security. In a coffeeshop, the security association is terminated by the store: is there any reason you should trust the shopkeeper? Perhaps link-layer security makes some sense in a home, where you control both the access point and the wireless machines. However, we prefer end-to-end security at the network layer or in the applications.

Source: Cheswick-chap2.pdf Firewalls and Internet Security: Repelling the Wily Hacker , Second Edition, William R. Cheswick, Steven M. Bellovin, and Aviel D. Rubin

- Read: PPA chapters 7,8,9
- PPA chapter 7: Network Layer Protocols
- PPA chapter 8: Transport Layer Protocols
- PPA chapter 9: Common Upper-Layer Protocols
- Skim: table of contents of RFC5246 The TLS Protocol Version 1.2
- wikipedia page https://en.wikipedia.org/wiki/Transport_Layer_Security

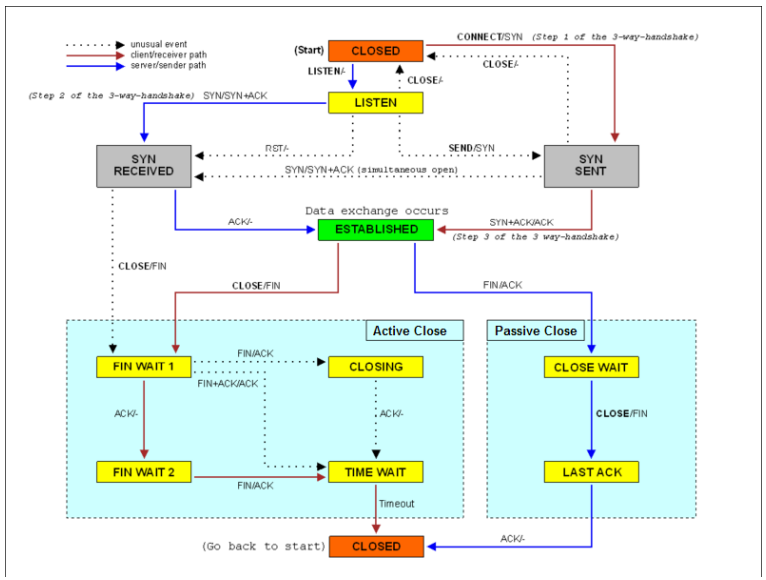
Reading Summary, continued

```
user@Projects:~$ ping -s 1472 -M do 91.102.91.18
PING 91.102.91.18 (91.102.91.18) 1472(1500) bytes of data.
1480 bytes from 91.102.91.18: icmp_seq=1 ttl=244 time=7.43 ms
1480 bytes from 91.102.91.18: icmp_seq=2 ttl=244 time=7.20 ms
...
user@Projects:~$ ping -s 1474 -M do 91.102.91.18
PING 91.102.91.18 (91.102.91.18) 1474(1502) bytes of data.
ping: local error: Message too long, mtu=1500
ping: local error: Message too long, mtu=1500
^C
--- 91.102.91.18 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 1025ms
```

PPA chapter 7: Network Layer Protocols

- What is normal network traffic?
- Great reference chapter for basic protocols
- Plus basic IPv6
- Re PATH MTU etc. Linux MTU 1500 check `ping -s 1472 -M do`

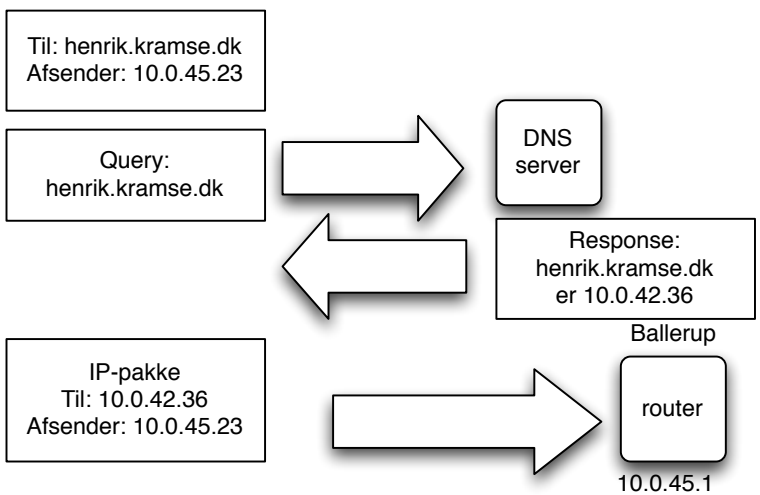
Reading Summary, continued



PPA chapter 8: Transport Layer Protocols

- Again the TCP 3-way handshake is described Note: can be done in 4 packets
- Closed TCP returns Reset (RST) packet, closed UDP returns ICMP port unreachable

Reading Summary, continued



PPA chapter 9: Common Upper-Layer Protocols

- Dynamic Host Configuration Protocol (DHCP) port 67/udp and 68/udp, and options
- Domain Name System (DNS) common query types A, NS, CNAME, MX, TXT, AAAA, AXFR, IXFR

Reading, Putty CVE-2024-31497

PuTTY SSH client flaw allows recovery of cryptographic private keys

The vulnerability tracked as CVE-2024-31497 was discovered by Fabian Bäumer and Marcus Brinkmann of the Ruhr University Bochum and is caused by how PuTTY generates ECDSA nonces (temporary unique cryptographic numbers) for the NIST P-521 curve used for SSH authentication.

...

Brinkmann explained on X that **attackers require 58 signatures** to calculate a **target's private key**, which they can acquire either by **collecting them from logins to an SSH server they control or is compromised, or from signed Git commits.**

Source: https://www.bleepingcomputer.com/news/security/putty-ssh-client-flaw-allows-recovery-of-cryptographic-private-keys/

Reading, Putty CVE-2024-31497

2024-04-15 PuTTY 0.81 released

PuTTY 0.81, released today, fixes a critical vulnerability CVE-2024-31497 in the use of 521-bit ECDSA keys (ecdsa-sha2-nistp521). If you have used a 521-bit ECDSA private key with any previous version of PuTTY, consider the private key compromised: remove the public key from `authorized_keys` files, and generate a new key pair.

However, this only affects that one algorithm and key size. No other size of ECDSA key is affected, and no other key type is affected.

Source: The Putty download page at https://www.chiark.greenend.org.uk/~sgtatham/putty/

Random number generation and use is hard!

https://nvd.nist.gov/vuln/detail/CVE-2024-31497

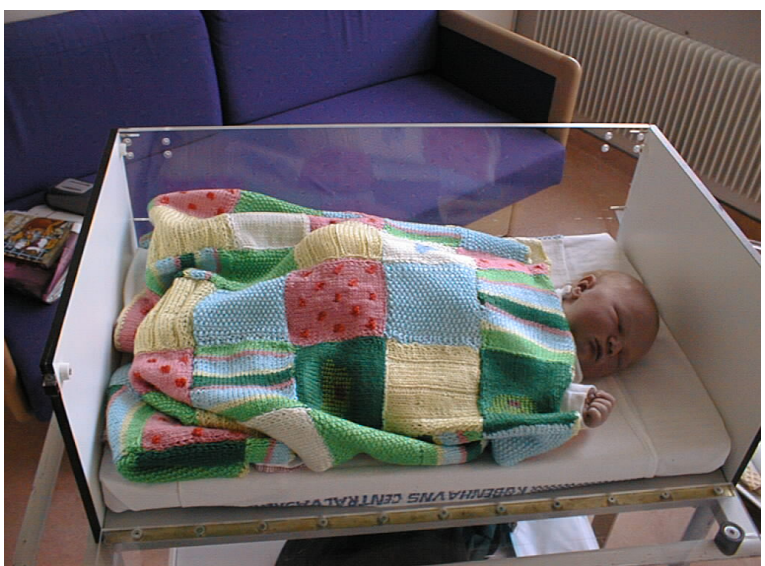
Another recent SSH related vulnerability which require active MiTM attack https://terrapin-attack.com/

What is data?

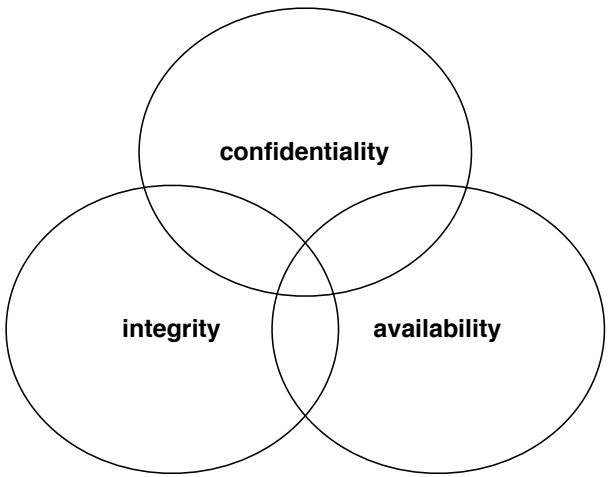
Personal data you dont want to loose:

- Wedding pictures
- Pictures of your children
- Sextapes
- Personal finances

Source: picture of my son less than 24 hours old - precious!



Confidentiality Integrity Availability



We want to protect something

Confidentiality - data holdes hemmelige

Integrity - data ændres ikke uautoriseret

Availability - data og systemet er tilgængelige når de skal bruges

Why think of security?

> Privacy is necessary for an open society in the electronic age. Privacy is not secrecy. A private matter is something one doesn't want the whole world to know, but a secret matter is something one doesn't want anybody to know. Privacy is the power to selectively reveal oneself to the world. A Cypherpunk's Manifesto by Eric Hughes, 1993

Copied from https://cryptoparty.org/wiki/CryptoParty

Solidaritetskryptering

Hvorfor skal vi kryptere?

```
      Køn
                      Seksualitet
Tro religion       hatecrimes

Politisk overbevisning, eller blot aktiv

Whistleblowers              soldater      diplomater
```

Du bestemmer ikke hvem der diskrimineres eller trues i andre lande

Når vi krypterer hjælper vi andre! **Solidaritetskryptering**

Person in the middle attacks

ARP spoofing, ICMP redirects, the classics

Used to be called Man in The Middle MiTM

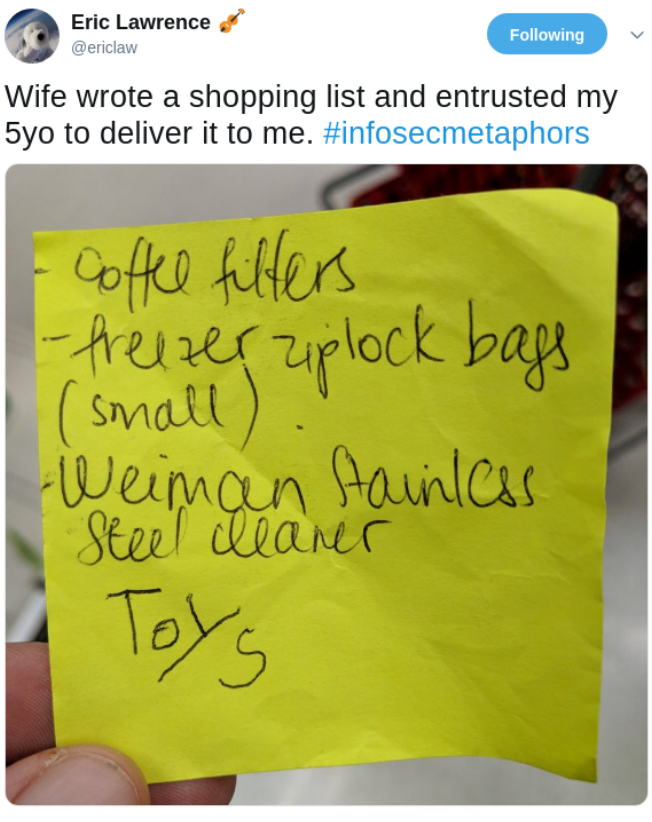
- ICMP redirect
- ARP spoofing
- Wireless listening and spoofing higher levels like airpwn-ng https://github.com/ICSec/airpwn-ng

Usually aimed at unencrypted protocols

Today we only talk about getting the data, not how to perform higher level attacks

Basic cryptography

- Confidentiality - data holdes hemmelige
- Integrity - data ændres ikke uautoriseret
- A common attack category is children intercepting messages
- or MiTM Mini in the Middle in this case



Cryptography

Cryptography or cryptology is the practice and study of techniques for secure communication

Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary

Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key, to ensure confidentiality, example algorithm AES

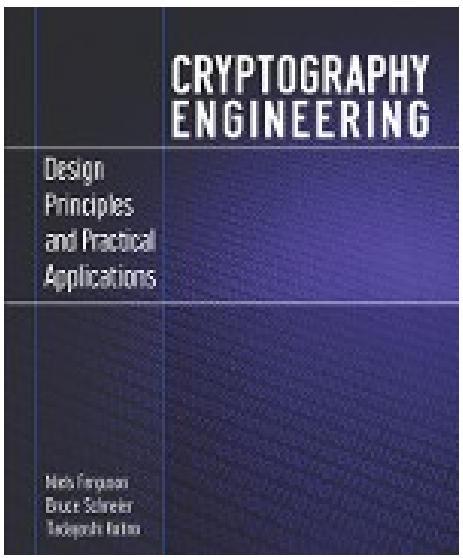
Public-key cryptography (like RSA) uses two related keys, a key pair of a public key and a private key. This allows for easier key exchanges, and can provide confidentiality, and methods for signatures and other services

Source: https://en.wikipedia.org/wiki/Cryptography

Kryptografi er svært

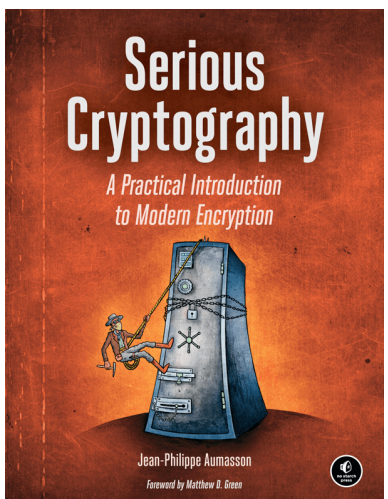
Åbent kursus på Stanford
http://crypto-class.org/

Kryptering: Cryptography Engineering



Cryptography Engineering by Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno, 2010
https://www.schneier.com/book-ce.html

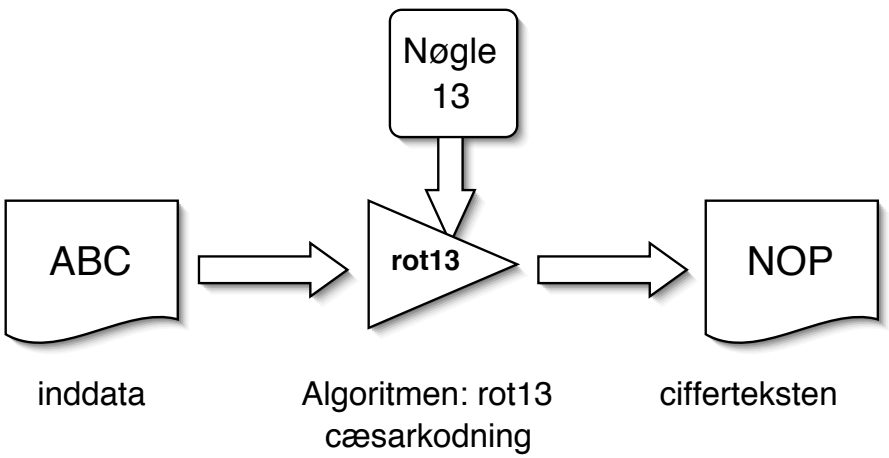
Serious Cryptography



Serious Cryptography A Practical Introduction to Modern Encryption by Jean-Philippe Aumasson November 2017, 312 pp. ISBN-13: 978-1-59327-826-7 https://nostarch.com/seriouscrypto

Encryption Decryption

Kryptografi



Kryptografi er læren om, hvordan man kan kryptere data

Kryptografi benytter algoritmer som sammen med nøgler giver en ciffertekst

- der kun kan læses ved hjælp af den tilhørende nøgle

Kryptografiske principper

Algoritmerne er kendte

Nøglerne er hemmelige

Nøgler har en vis levetid - de skal skiftes ofte

Et successfuldt angreb på en krypto-algoritme er enhver genvej som kræver mindre arbejde end en gennemgang af alle nøglerne

Nye algoritmer, programmer, protokoller m.v. skal gennemgås nøje!

Se evt. Snake Oil Warning Signs: Encryption Software to Avoid https://www.schneier.com/crypto-gram/archives/1999/0215.html#snakeoil

Wikipedia Snake oil description https://en.wikipedia.org/wiki/Snake_oil_(cryptography)

DES, Triple DES og AES

AES

Advanced Encryption Standard

DES kryptering - gammel og pensioneret!

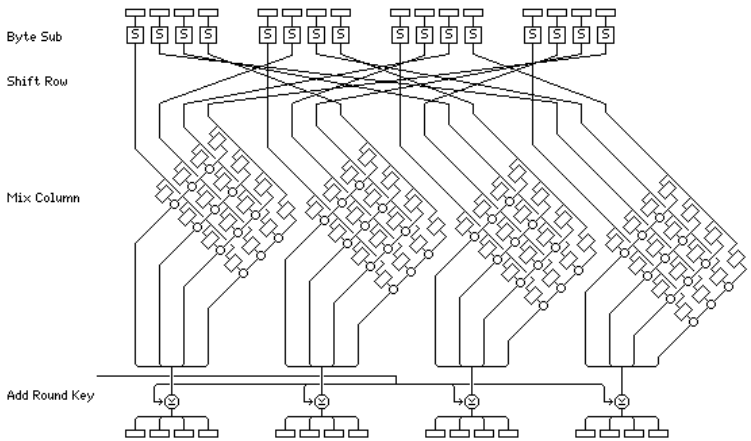
Der blev i 2001 vedtaget en ny standard algoritme Advanced Encryption Standard (AES) som afløser Data Encryption Standard (DES)

Algoritmen hedder Rijndael og er udviklet af Joan Daemen og Vincent Rijmen.

Se også https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Findes animationer (med fejl) https://www.youtube.com/watch?v=mIzxpkdXP58

AES Advanced Encryption Standard



- The official Rijndael web site displays this image to promote understanding of the Rijndael round transformation [8].
- Key sizes 128,192,256 bit typical
- Some extensions in cryptosystems exist: XTS-AES-256 really is 2 instances of AES-128 and 384 is two instances of AES-192 and 512 is two instances of AES-256
- https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

RSA

> RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems and is widely used for secure data transmission. ... In RSA, this asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem". The acronym RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1978.

- Key sizes 1,024 to 4,096 bit typical
- Quote from: https://en.wikipedia.org/wiki/RSA_(cryptosystem)

RSA operation

The RSA algorithm involves four steps: key generation, key distribution, encryption and decryption.

A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integers m (with $0m < n$):

$(m^e)^d \equiv m \pmod{n} (m^e)^d \equiv m \pmod{n}$

and that knowing e and n, or even m, it can be extremely difficult to find d. The triple bar ( ) here denotes modular congruence.

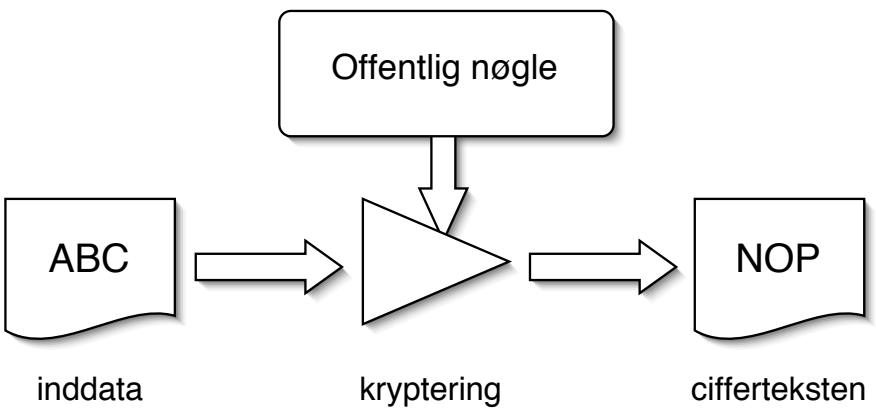
In addition, for some operations it is convenient that the order of the two exponentiations can be changed and that this relation also implies:

$(m^d)^e \equiv m \pmod{n} (m^d)^e \equiv m \pmod{n}$

RSA involves a public key and a private key. The public key can be known by everyone, and it is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount of time by using the private key. The public key is represented by the integers n and e; and, the private key, by the integer d (although n is also used during the decryption process. Thus, it might be considered to be a part of the private key, too). m represents the message (previously prepared with a certain technique explained below).

Source: https://en.wikipedia.org/wiki/RSA_(cryptosystem)

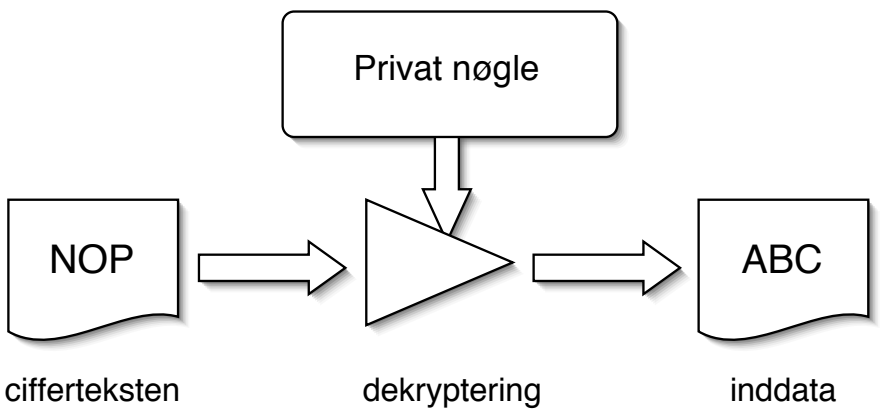
Public key kryptografi - 1



privat-nøgle kryptografi (eksempelvis AES) benyttes den samme nøgle til kryptering og dekryptering

offentlig-nøgle kryptografi (eksempelvis RSA) benytter to separate nøgler til kryptering og dekryptering

Public key kryptografi - 2



offentlig-nøgle kryptografi (eksempelvis RSA) bruger den private nøgle til at dekryptere

man kan ligeledes bruge offentlig-nøgle kryptografi til at signere dokumenter
- som så verificeres med den offentlige nøgle

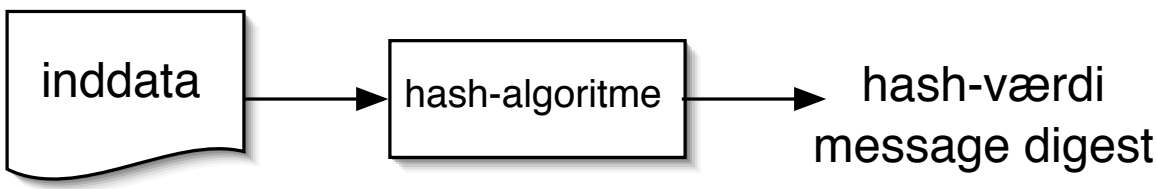
NB: Kryptering alene sikrer ikke anonymitet

Elliptic Curve

> Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-EC cryptography (based on plain Galois fields) to provide equivalent security.[1]

- Today we use https://en.wikipedia.org/wiki/Elliptic-curve_cryptography

Hashing - MD5 message digest funktion



HASH algoritmer giver en næsten unik værdi baseret på input

værdien ændres radikalt selv ved små ændringer i input

MD5 er blandt andet beskrevet i RFC-1321: The MD5 Message-Digest Algorithm

Algoritmen MD5 er baseret på MD4, begge udviklet af Ronald L. Rivest

Både MD5 og SHA-1 er idag gamle og skal generelt ikke bruges mere

Idag benyttes eksempelvis https://en.wikipedia.org/wiki/PBKDF2

Base64

In computer science, Base64 is a group of binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation. The term Base64 originates from a specific MIME content transfer encoding. Each Base64 digit represents exactly 6 bits of data. Three 8-bit bytes (i.e., a total of 24 bits) can therefore be represented by four 6-bit Base64 digits.

- Encoding data is not secure, can be decoded
- https://en.wikipedia.org/wiki/Base64

Cracking passwords

- Hashcat is the world's fastest CPU-based password recovery tool.
- oclHashcat-plus is a GPGPU-based multi-hash cracker using a brute-force attack (implemented as mask attack), combinator attack, dictionary attack, hybrid attack, mask attack, and rule-based attack.
- oclHashcat-lite is a GPGPU cracker that is optimized for cracking performance. Therefore, it is limited to only doing single-hash cracking using Markov attack, Brute-Force attack and Mask attack.
- John the Ripper password cracker old skool men stadig nyttig

Source:
http://hashcat.net/wiki/
http://www.openwall.com/john/

Encryption key length - who are attacking you

Encryption key lengths & hacking feasibility

Type of Attacker	*Budget*	Tool	*Time & Cost/Key 40 bit*	*Time & Cost/Key 56 bit*
Regular User	Minimal $400	Scavenged computer time FPGA	1 week 5 hours ($.08)	Not feasible 38 years ($5,000)
Small Business	$10,000	FPGA [1]	12 min.($.08)	556 days ($5,000)
Corporate Department	$300,000	FPGA ASIC [2]	24 sec. ($.08) 0.18 sec. ($.001)	19 days ($5,000) 3 hours ($38)
Large Corporation	$10M	ASIC	0.005 sec.($0.001)	6 min. ($38)
Intelligence Agency	$300M	ASIC	0.0002 sec.($0.001)	12 sec. ($38)

Source: http://www.mycrypto.net/encryption/encryption_crack.html

More up to date: In 1998, the EFF built Deep Crack for less than $250,000

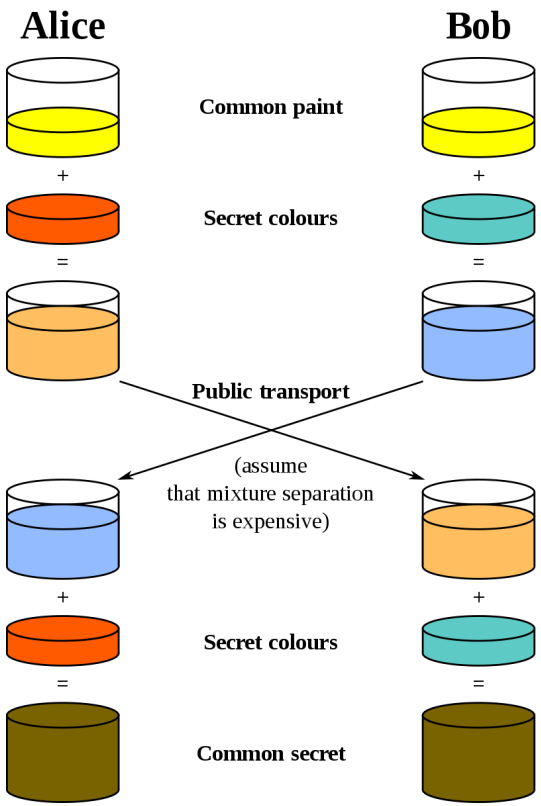
https://en.wikipedia.org/wiki/EFF_DES_cracker

FPGA Based UNIX Crypt Hardware Password Cracker - 100 EUR in 2006

http://www.sump.org/projects/password/

Encryption on the web – Diffie Helman exchange

Diffie–Hellman key exchange (DH)[nb 1] is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as originally conceptualized by Ralph Merkle and named after Whitfield Diffie and Martin Hellman.[1][2] DH is one of the earliest practical examples of public key exchange implemented within the field of cryptography. ... The scheme was first published by Whitfield Diffie and Martin Hellman in 1976



- Quote from: https://en.wikipedia.org/wiki/Diffie-Hellman_key_exchange
- Today we use https://en.wikipedia.org/wiki/Elliptic-curve_cryptography

Transport Layer Security (TLS)

Oprindeligt udviklet af Netscape Communications Inc.

Secure Sockets Layer SSL er idag blevet adopteret af IETF og kaldes derfor også for Transport Layer Security TLS TLS er baseret på SSL Version 3.0

RFC-2246 The TLS Protocol Version 1.0 fra Januar 1999

RFC-3207 SMTP STARTTLS

Det er svært!

Stanford Dan Boneh udgiver en masse omkring crypto
https://crypto.stanford.edu/~dabo/cryptobook/

Which TLS version

Recent versions of TLS are more secure than older versions. The oldest three versions of TLS, SSL 1.0, SSL 2.0 and SSL 3.0 cannot be used securely. The most recent version of TLS, TLS 1.3 offers the best protection.

Version	*Status*
TLS 1.3	Good (3; 4)
TLS 1.2	Sufficient (3; 4)
TLS 1.1	Phase out (3; 4)
TLS 1.0	
SSL 3.0	Insufficient (3; 4)
SSL 2.0	
SSL 1.0	

Table 1 – Versions

Source: *IT Security Guidelines for Transport Layer Security (TLS)* v2.1, National Cyber Security Centre, 2021

https://english.ncsc.nl/publications/publications/2021/january/19/it-security-guidelines-for-transport-layer-security-2.1

- Danish datatilsynet and CfCS.dk also has some guidance

TLS version 1.2 and 1.3 details

TLS 1.2	TLS 1.3
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	ECDHE RSA TLS_AES_256_GCM_SHA384

	Key exchange	Certificate verification	Bulk encryption	Hashing
Good	ECDHE	ECDSA RSA	AES_256_GCM CHACHA20_POLY1305 AES_128_GCM	(HMAC-)SHA-384 (HMAC-)SHA-256
Sufficient	DHE		AES_256_CBC AES_128_CBC	(HMAC-)SHA-1
Phase out	RSA*		3DES-CBC+	

* Written as TLS_RSA_WITH_...

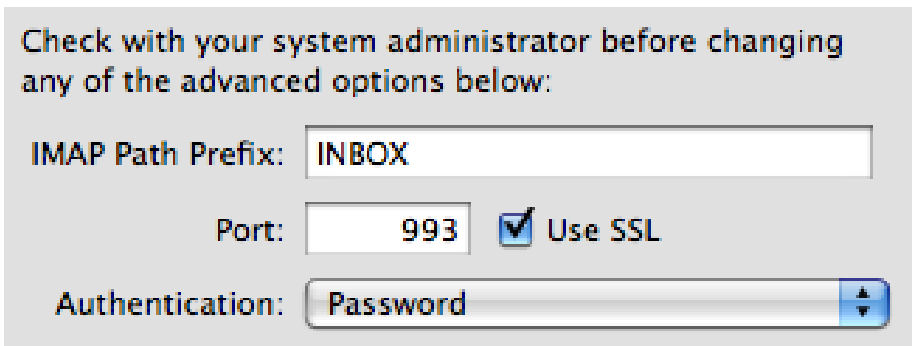
\+ Written as 3DES_EDE_CBC or DES_CBC3

Figure 2 – Cipher suite notation in TLS 1.2 and TLS 1.3. The table summarizes algorithm selections and their security level.
Not included in the (old) cipher suite notation are: versions; hash functions for certificate verification; hash functions for key exchange; key sizes & choice of groups; and options.
These can be found in their respective sections. For ordering, refer to the section Prefer faster and safer algorithms.

Source: *IT Security Guidelines for Transport Layer Security (TLS)* v2.1, National Cyber Security Centre, 2021

https://english.ncsc.nl/publications/publications/2021/january/19/it-security-guidelines-for-transport-layer-security-2.1

SSL/TLS udgaver af protokoller



Mange protokoller findes i udgaver hvor der benyttes SSL

HTTPS vs HTTP

IMAPS, POP3S, osv.

Bemærk: nogle protokoller benytter to porte IMAP 143/tcp vs IMAPS 993/tcp

Andre benytter den samme port men en kommando som starter:

SMTP STARTTLS RFC-3207

Secure protocols

Securing e-mail

- Pretty Good Privacy - Phil Zimmermann
- OpenPGP = e-mail security

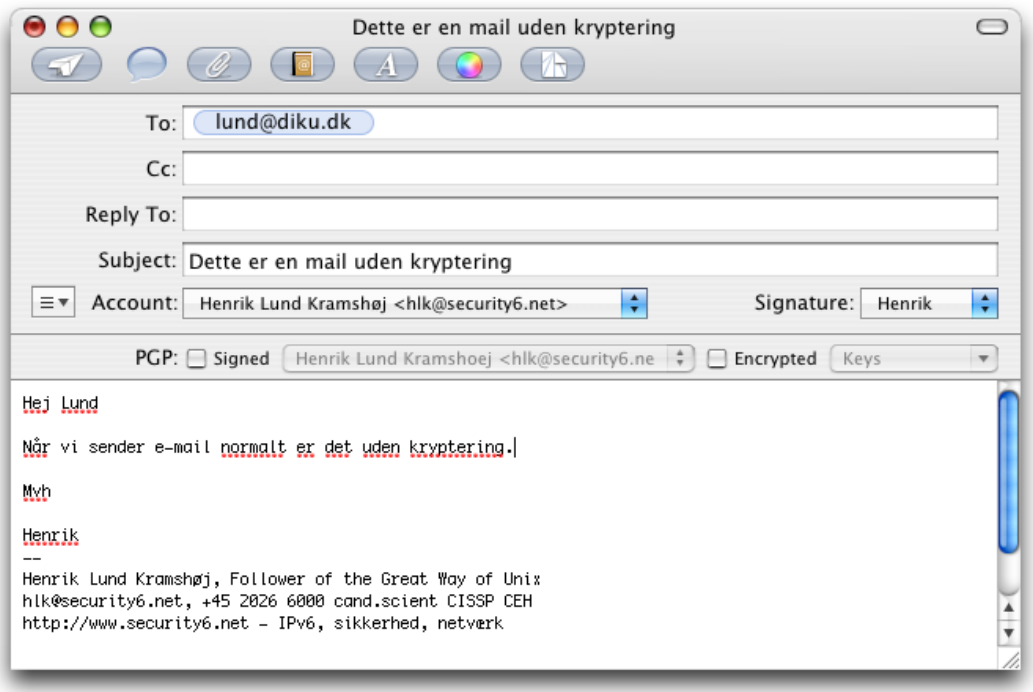
Network sessions use SSL/TLS

- Secure Sockets Layer SSL / Transport Layer Services TLS
- Encrypting data sent and received
- SSL/TLS already used for many protocols as a wrapper: POP3S, IMAPS, SSH, SMTP+TLS m.fl.

Encrypting traffic at the network layer - Virtual Private Networks VPN

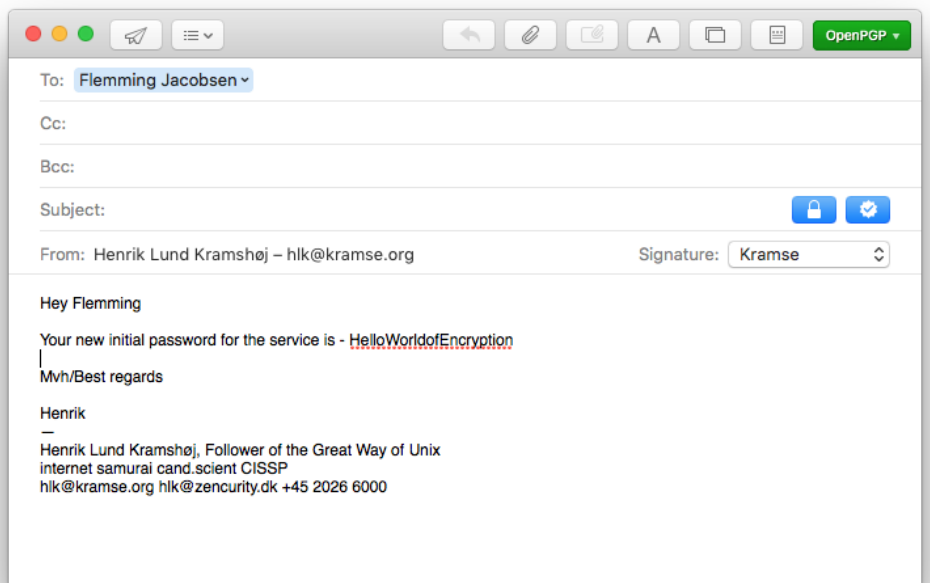
- IPsec IP Security Framework, se også L2TP
- PPTP Point to Point Tunneling Protocol - dårlig og usikker, brug den ikke mere!
- OpenVPN uses SSL/TLS across TCP or UDP

Email er usikkert



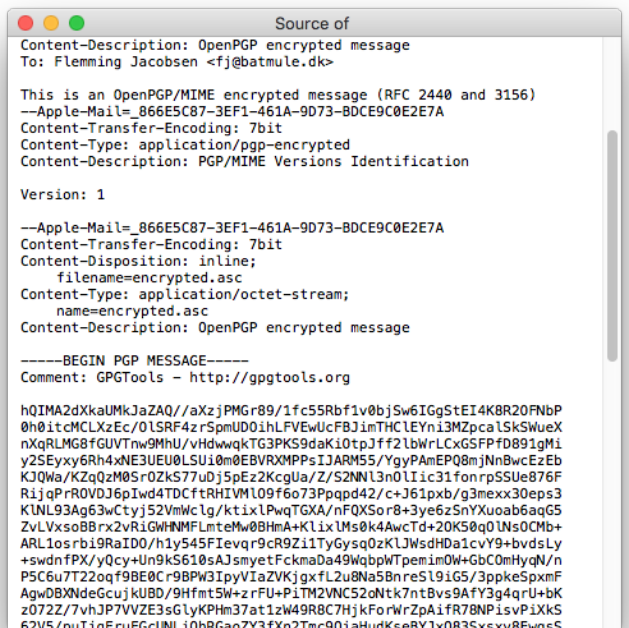
Email uden kryptering - er som et postkort

Email med OpenPGP kryptering - afsendelse



En sikker krypteret email er ikke sværere at sende

Krypteret OpenPGP Email under transporten



En sikker krypteret email er beskyttet undervejs

Fokus 2024: TLS og VPN indstillinger

```
ssl_prefer_server_ciphers on;
ssl_protocols TLSv1 TLSv1.1 TLSv1.2; # not possible to do exclusive
ssl_ciphers 'EDH+CAMELLIA:EDH+aRSA:EECDH+aRSA+AESGCM:EECDH+aRSA+SHA384:EECDH+\
  \aRSA+SHA256:EECDH:+CAMELLIA256:+AES256:+CAMELLIA128:+AES128:+SSLv3:!aNULL:!\
  \eNULL:!LOW:!3DES:!MD5:!EXP:!PSK:!DSS:!RC4:!SEED:!ECDSA:CAMELLIA256-SHA:AES256\
  \-SHA:CAMELLIA128-SHA:AES128-SHA';
add_header Strict-Transport-Security max-age=15768000; # six months
# use this only if all subdomains support HTTPS!
# add_header Strict-Transport-Security "max-age=15768000; includeSubDomains";
```

Listing 2.6: *SSL settings for nginx*
[configuration/Webservers/nginx/default]

Old settings for Nginx, don't use!

- De fleste har https nu, men er det konfigureret optimalt
- Vi bruger også VPN til at forbinde sites, kontorer
- Anbefaler at alle indstillingerne gennemgås mindst en gang om året!

Opportunistic TLS

Today most email is protected using Opportunistic TLS

> Opportunistic TLS (Transport Layer Security) refers to extensions in plain text communication protocols, which offer a way to upgrade a plain text connection to an encrypted (TLS or SSL) connection instead of using a separate port for encrypted communication. Several protocols use a command named "STARTTLS"for this purpose. It is a form of opportunistic encryption and is primarily intended as a countermeasure to passive monitoring.
>
> The STARTTLS command for IMAP and POP3 is defined in RFC 2595, for SMTP in RFC 3207, for XMPP in RFC 6120 and for NNTP in RFC 4642. For IRC, the IRCv3 Working Group has defined the STARTTLS extension. FTP uses the command "AUTH TLS"defined in RFC 4217 and LDAP defines a protocol extension OID in RFC 2830. HTTP uses an upgrade header.

Source: https://en.wikipedia.org/wiki/Opportunistic_TLS

- Easy to setup using Lets Encrypt
- SMTP MTA Strict Transport Security (MTA-STS) also exist, enforce and only accept email using encryption

Nmap efter SSL og TLS

Example Usage

```
nmap -sV -sC <target>
```

Script Output

```
443/tcp open   https   syn-ack
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_IDEA_128_CBC_WITH_MD5
|     SSL2_RC2_CBC_128_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC2_CBC_128_CBC_WITH_MD5
|_    SSL2_RC4_128_EXPORT40_WITH_MD5
```

Nu vi har lært Kali og Nmap at kende

- Find nemt alle ssl version 2 og 3
  `nmap --script ssl-enum-ciphers`
- Brug ssllabs https://www.ssllabs.com/

sslscan

```
root@kali:~# sslscan --ssl2 web.kramse.dk
Version: 1.10.5-static
OpenSSL 1.0.2e-dev xx XXX xxxx

Testing SSL server web.kramse.dk on port 443
...
  SSL Certificate:
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength:    2048

Subject:  *.kramse.dk
Altnames: DNS:*.kramse.dk, DNS:kramse.dk
Issuer:   AlphaSSL CA - SHA256 - G2
```

Source: Originally sslscan from http://www.titania.co.uk but use the version on Kali

SSLscan can check your own sites, while Qualys SSLLabs only can test from hostname

Example Weak DH paper

Weak Diffie-Hellman and the Logjam Attack

Good News! Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports `DHE_EXPORT` ciphers, and affects all modern web browsers. 8.4% of the Top 1 Million domains were initially vulnerable.

2. **Threats from state-level adversaries.** Millions of HTTPS, SSH, and VPN servers all use the same prime numbers for Diffie-Hellman key exchange. Practitioners believed this was safe as long as new key exchange messages were generated for every connection. However, the first step in the number field sieve—the most efficient algorithm for breaking a Diffie-Hellman connection—is dependent only on this prime. After this first step, an attacker can quickly break individual connections.

   We carried out this computation against the most common 512-bit prime used for TLS and demonstrate that the Logjam attack can be used to downgrade connections to 80% of TLS servers supporting `DHE_EXPORT`. We further estimate that an academic team can break a 768-bit prime and that a nation-state can break a 1024-bit prime. Breaking the single, most common 1024-bit prime used by web servers would allow passive eavesdropping on connections to 18% of the Top 1 Million HTTPS domains. A second prime would allow passive decryption of connections to 66% of VPN servers and 26% of SSH servers. A close reading of published NSA leaks shows that the agency's attacks on VPNs are consistent with having achieved such a break.

Source: https://weakdh.org/ and
https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf

Every year in different SSL/TLS implementations there have been problems.

FREAK March 2015

> "A group of cryptographers at INRIA, Microsoft Research and IMDEA have discovered some serious vulnerabilities in OpenSSL (e.g., Android) clients and Apple TLS/SSL clients (e.g., Safari) that allow a 'man in the middle attacker' to downgrade connections from 'strong' RSA to 'export-grade' RSA. These attacks are real and exploitable against a shocking number of websites – including government websites. Patch soon and be careful."

Source: Matthew Green, cryptographer and research professor at Johns Hopkins Univ

http://blog.cryptographyengineering.com/2015/03/attack-of-week-freak-or-factoring-nsa.html

https://www.smacktls.com/ https://freakattack.com/

OpenSSL, LibreSSL, Apple SSL flaw exit exit exit!, Android SSL, certs certs!!!111, SSLv3, Heartbleed, MS TLS

PS From now on its TLS! Not SSL anymore, any SSLv2, SSLv3 is old and vulnerable

Heartbleed CVE-2014-0160

The Heartbleed Bug

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.

Nok det mest kendte SSL/TLS exploit

Source: http://heartbleed.com/

Heartbleed hacking

```
06b0: 2D 63 61 63 68 65 0D 0A 43 61 63 68 65 2D 43 6F  -cache..Cache-Co
06c0: 6E 74 72 6F 6C 3A 20 6E 6F 2D 63 61 63 68 65 0D  ntrol: no-cache.
06d0: 0A 0D 0A 61 63 74 69 6F 6E 3D 67 63 5F 69 6E 73  ...action=gc_ins
06e0: 65 72 74 5F 6F 72 64 65 72 26 62 69 6C 6C 6E 6F  ert_order&billno
06f0: 3D 50 5A 4B 31 31 30 31 26 70 61 79 6D 65 6E 74  =PZK1101&payment
0700: 5F 69 64 3D 31 26 63 61 72 64 5F 6E 75 6D 62 65  _id=1&card_numbe
0710: XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX   r=4060xxxx413xxx
0720: 39 36 26 63 61 72 64 5F 65 78 70 5F 6D 6F 6E 74  96&card_exp_mont
0730: 68 3D 30 32 26 63 61 72 64 5F 65 78 70 5F 79 65  h=02&card_exp_ye
0740: 61 72 3D 31 37 26 63 61 72 64 5F 63 76 6E 3D 31  ar=17&card_cvn=1
0750: 30 39 F8 6C 1B E5 72 CA 61 4D 06 4E B3 54 BC DA  09.l..r.aM.N.T..
```

- Obtained using Heartbleed proof of concepts - Gave full credit card details
- "can XXX be exploited- yes, clearly! PoCs ARE needed
  without PoCs even Akamai wouldn't have repaired completely!
- The internet was ALMOST fooled into thinking getting private keys from Heartbleed was not possible - scary indeed.

Key points after heartbleed

Source: picture source

https://www.duosecurity.com/blog/heartbleed-defense-in-depth-part-2

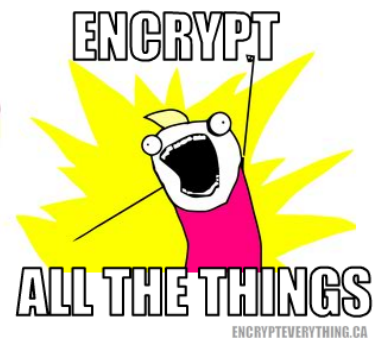
- Writing SSL software and other secure crypto software is hard
- Configuring SSL is hard
  check you own site https://www.ssllabs.com/ssltest/
- SSL is hard, finding bugs "all the time"http://armoredbarista.blogspot.dk/2013/01/a-brief-chronology-of-ssltls-attacks.html

HTTPS Everywhere

HTTPS Everywhere is a Firefox extension produced as a collaboration between The Tor Project and the Electronic Frontier Foundation. It encrypts your communications with a number of major websites.

http://www.eff.org/https-everywhere

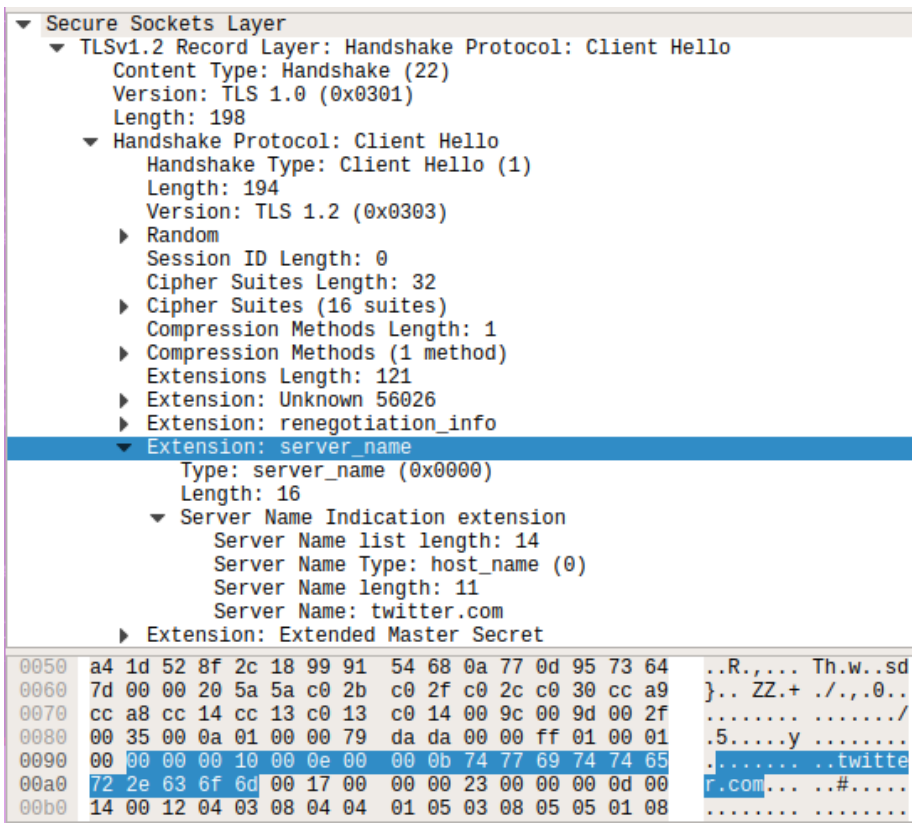
TLS Server Name Indication extension



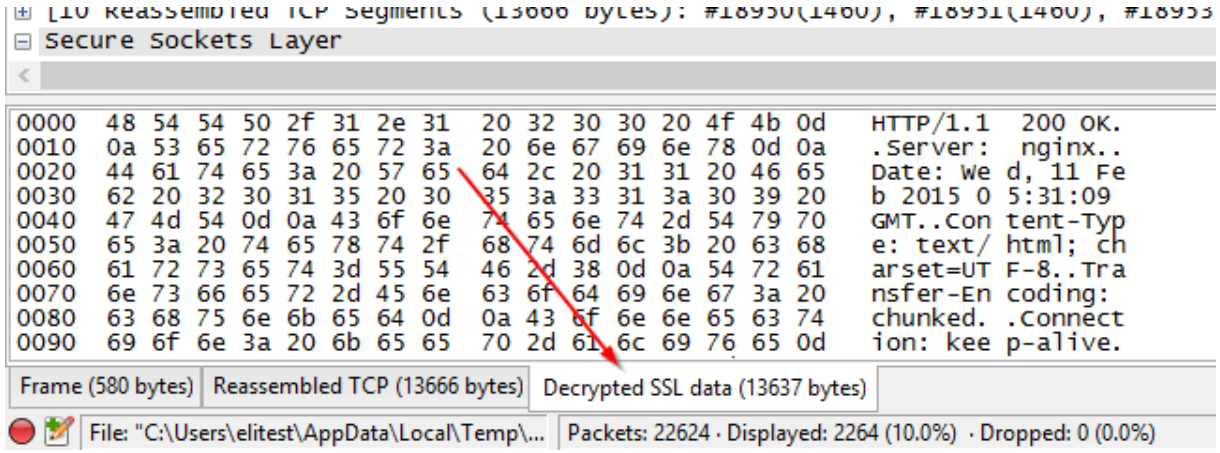
Vi skal kryptere, men desværre så skjuler vores HTTPS ikke hvad site vi tilgår.

- HTTPS er idag TLS Transport Layer Security
- Verifikation sker med certifikater der præsenteres af server
- Der kan være flere sites på en enkelt IP - med SNI
- Desværre vælges det rigtige certifikat før krypteringen starter

TLS Server Name Indication example



Decrypting TLS Browser Traffic With Wireshark

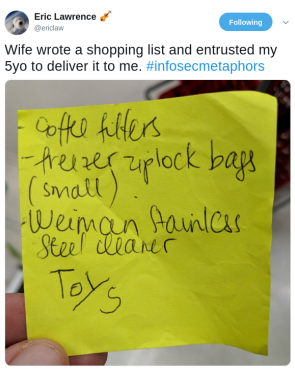


- Firefox and Chrome both support logging the symmetric session key used to encrypt TLS traffic to a file
- Wireshark can read this file - and decrypt sessions - Nifty trick
- Another option is to use Burp

Source: great blog article about the features used

https://jimshaver.net/2015/02/11/decrypting-tls-browser-traffic-with-wireshark-the-easy-way/

Basic MiTM



- A common attack category is children intercepting messages
- or MiTM Mini in the Middle in this case
- Note: sslstrip and other MiTM tools for TLS/SSL are becoming outdated!

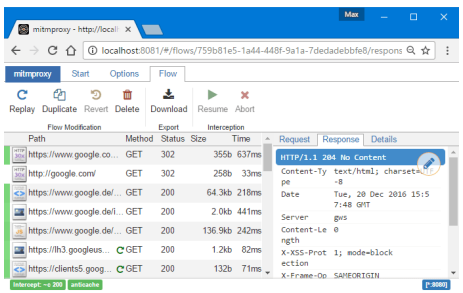
sslstrip - transparently hijack HTTP

> This tool provides a demonstration of the HTTPS stripping attacks that I presented at Black Hat DC 2009. It will transparently hijack HTTP traffic on a network, watch for HTTPS links and redirects, then map those links into either look-alike HTTP links or homograph-similar HTTPS links. It also supports modes for supplying a favicon which looks like a lock icon, selective logging, and session denial. For more information on the attack, see the video from the presentation below.

https://moxie.org/software/sslstrip/

- *First, arpspoof convinces a host that our MAC address is the router's MAC address, and the target begins to send us all its network traffic.*
- *supplying a favicon which looks like a lock icon*

mitmproxy - interactive HTTPS proxy



https://mitmproxy.org/

- Command line, Web interface, Python API
- Intercept HTTP and HTTPS requests and responses and modify them on the fly
- Reverse proxy mode to forward traffic to a specified server
- Make scripted changes to HTTP traffic using Python
- SSL/TLS certificates for interception are generated on the fly

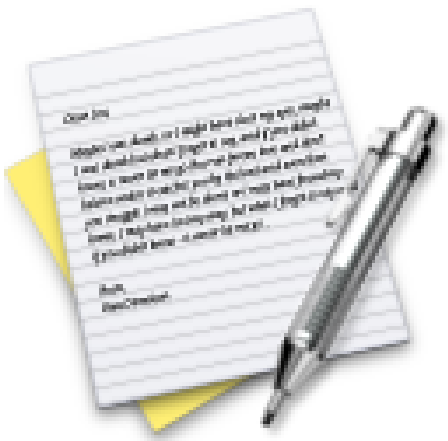
sslsplit - transparent SSL/TLS interception

Overview
SSLsplit is designed to transparently terminate connections that are redirected to it using a network address translation engine. SSLsplit then terminates SSL/TLS and initiates a new SSL/TLS connection to the original destination address, while logging all data transmitted. Besides NAT based operation, SSLsplit also supports static destinations and using the server name indicated by SNI as upstream destination. SSLsplit is purely a transparent proxy and cannot act as a HTTP or SOCKS proxy configured in a browser.

https://www.roe.ch/SSLsplit

- SSLsplit implements a number of defences against mechanisms which would normally prevent MitM attacks or make them more difficult
- SSLsplit can deny OCSP requests in a generic way. For HTTP and HTTPS connections, SSLsplit mangles headers to prevent server-instructed public key pinning (HPKP), avoid strict transport security restrictions (HSTS), avoid Certificate Transparency enforcement (Expect-CT) and prevent switching to QUIC/SPDY, HTTP/2 or WebSockets (Upgrade, Alternate Protocols)

Exercise



Now lets do the exercise

⚠ SSL/TLS scanners 15 min

which is number 33 in the exercise PDF.

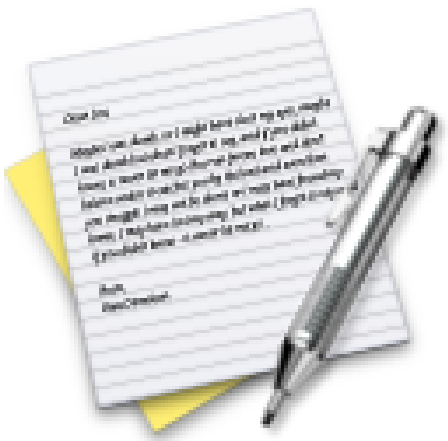
Exercise

Now lets do the exercise

⚠ Internet scanners 15 min

which is number **34** in the exercise PDF.

Exercise

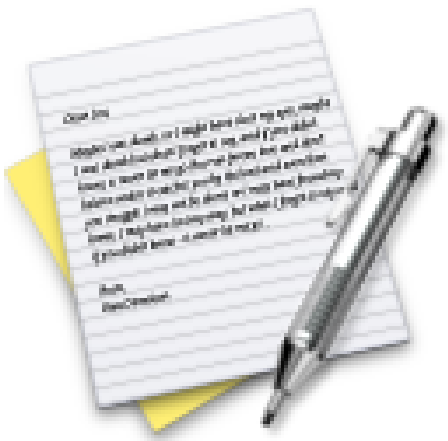


Now lets do the exercise

❶ Nginx as a Transport Layer Security (TLS) endpoint 40 min

which is number **35** in the exercise PDF.

Exercise

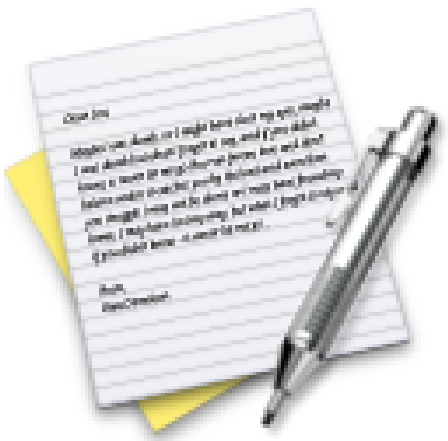


Now lets do the exercise

⚠ Nginx logging 20 min

which is number **36** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Nginx filtering 40 min

which is number **37** in the exercise PDF.

Other crypto related stuff

PRNG

Debian OpenSSL [edit]

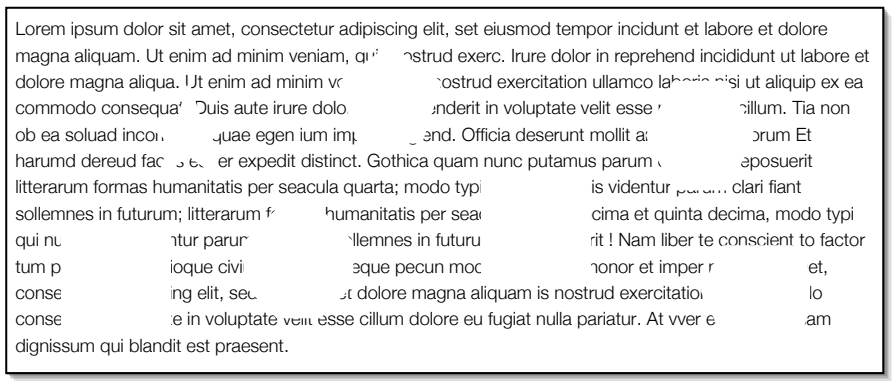
In May 2008, security researcher Luciano Bello revealed his discovery that changes made in 2006 to the random number generator in the version of the OpenSSL package distributed with Debian GNU/Linux and other Debian-based distributions, such as Ubuntu, dramatically reduced the entropy of generated values and made a variety of security keys vulnerable to attack.[10][11] The security weakness was caused by changes made to the openssl code by a Debian developer in response to compiler warnings of apparently redundant code.[12] This caused a massive worldwide regeneration of keys, and despite all attention the issue got, it could be assumed many of these old keys are still in use. Key types affected include SSH keys, OpenVPN keys, DNSSEC keys, key material for use in X.509 certificates and session keys used in SSL/TLS connections. Keys generated with GnuPG or GNUTLS are not affected as these programs used different methods to generate random numbers. Non-Debian-based Linux distributions are also unaffected. This security vulnerability was promptly patched after it was reported.

https://en.wikipedia.org/wiki/Random_number_generator_attack#Debian_OpenSSL

The random number generator is VITAL for crypto security

Check out modern CPUs and Linux response to this
https://en.wikipedia.org/wiki/RdRand

Is your data secure - data at rest



Stolen laptop, tablet, phone - can anybody read your data?

Do you trust "remote wipe"

How do you in fact wipe data securely off devices, and SSDs?

Encrypt disk and storage devices before using them in the first place!

Circumvent security - single user mode boot

Unix systems often allows boot into singleuser mode
press command-s when booting Mac OS X

Laptops can often be booted using PXE network or CD boot

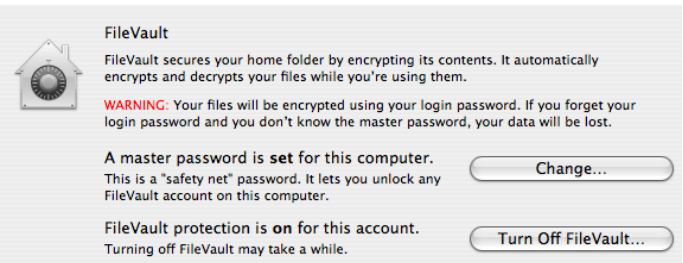
Mac computers can become a Firewire disk
hold t when booting - firewire target mode

Unrestricted access to un-encrypted data

Moving hard drive to another computer is also easy

Physical access is often - **game over**

Encrypting hard disk



Becoming available in the most popular client operating systems

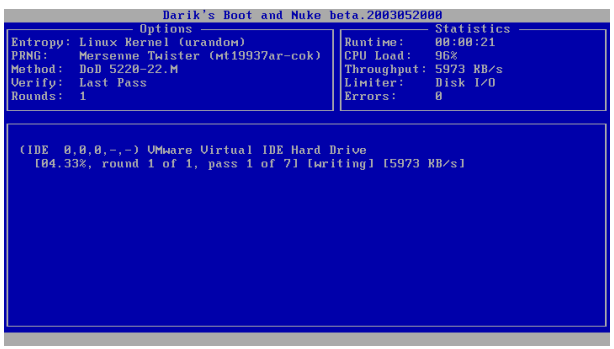
- Microsoft Windows Bitlocker
- Apple Mac OS X - FileVault
- FreeBSD GEOM og GBDE - encryption framework
- Linux LUKS distributions like Ubuntu ask to encrypt home dir during installation
- Some vendors have BIOS passwords, or disk passwords

Attacks on disk encryption

- Firewire, DMA & Windows, Winlockpwn via FireWire
  Hit by a Bus: Physical Access Attacks with Firewire Ruxcon 2006
- Removing memory from live system - data is not immediately lost, and can be read under some circumstances
  Lest We Remember: Cold Boot Attacks on Encryption Keys
  http://citp.princeton.edu/memory/
- This is very CSI or Hollywoord like - but a real threat
- VileFault decrypts encrypted Mac OS X disk image files
  https://code.google.com/p/vilefault/
- FileVault Drive Encryption (FVDE) (or FileVault2) encrypted volumes
  https://code.google.com/p/libfvde/

So perhaps use both hard drive encryption AND turn off computer after use?

... and deleting data



Getting rid of data from old devices is a pain

Some tools will not overwrite data, leaving it vulnerable to recovery

Even secure erase programs might not work on SSD
- due to reallocation of blocks

I have used Darik's Boot and Nuke ("DBAN")
http://www.dban.org/

2018 attack

Drive	1	2	3	4	5	6	7	8	9	Impact
Crucial MX100 (all form factors)	✗	✗	✗							✗ Compromised
Crucial MX200 (all form factors)	✗	✗	✗							✗ Compromised
Crucial MX300 (all form factors)	✓	✓	✓		✗	✓	✓	✓	✓	✗ Compromised
Samsung 840 EVO (SATA)	✗	✓	✓		✓	✓	✓	✗	✓	∼ Depends
Samsung 850 EVO (SATA)	✗	✓	✓		✓	✓	✓	✓	✓	∼ Depends
Samsung T3 (USB)				✗						✗ Compromised
Samsung T5 (USB)				✗						✗ Compromised

[1] Cryptographic binding in ATA Security (High mode)
[2] Cryptographic binding in ATA Security (Max mode)
[3] Cryptographic binding in TCG Opal
[4] Cryptographic binding in proprietary standard
[5] No single key for entire disk
[6] Randomized DEK on sanitize
[7] Sufficient random entropy
[8] No wear leveling related issues
[9] No DEVSLP related issues

self-encrypting deception: weakness in the encryption of solid state drives (SSDs)
https://www.ru.nl/publish/pages/909282/draft-paper.pdf

PGP/GPG verifikation af integriteten

Pretty Good Privacy PGP

Gnu Privacy Guard GPG

Begge understøtter OpenPGP - fra IETF RFC-2440

Når man har hentet og verificeret en nøgle kan man fremover nemt checke integriteten af software pakker

```
hlk@bigfoot:postfix$ gpg --verify  postfix-2.1.5.tar.gz.sig
gpg: Signature made Wed Sep 15 17:36:03 2004 CEST using RSA key ID D5327CB9
gpg: Good signature from "wietse venema <wietse@porcupine.org>"
gpg:                 aka "wietse venema <wietse@wzv.win.tue.nl>"
```

Secure Shell - SSH og SCP

SSH er oprindeligt udviklet af Tatu Ylönen i Finland, https://en.wikipedia.org/wiki/Secure_Shell

OpenSSH er idag det mest udbredte og inkluderet i Mac OS X, Windows og de fleste Linux distributioner https://www.openssh.com/

SSH afløser en række protokoller som er usikre: Telnet, r* rsh/rcp/rlogin og FTP

SSH - de nye kommandoer er

kommandoerne er:

- ssh - Secure Shell
- scp - Secure Copy
- sftp - secure FTP

Husk: SSH er både navnet på protokollerne - version 1 og 2 samt programmet `ssh` til at logge ind på andre systemer

SSH tillader også port-forward, tunnel til usikre protokoller, eksempelvis X protokollen til UNIX grafiske vinduer

NB: Man bør idag bruge SSH protokol version 2!

SSH nøgler

I praksis benytter man nøgler fremfor kodeord

I kan lave jeres egne SSH nøgler med programmerne i Putty eller `ssh-keygen`

Hvilken del skal jeg have for at kunne give jer adgang til en server?

Hvordan får jeg smartest denne nøgle?

først skal der genereres et nøglepar `id_ed25519` og `id_ed25519.pub`

Den offentlige del, filen `id_ed25519.pub`, kopieres til serveren, brug `ssh-copy-id`

Der logges ind på serveren

Nice tool ssh-copy-id

Highly recommended to use the utility tool `ssh-copy-id` for copying the public key to the server. Install tool if not available using `apt` :

```
hlk@kunoichi:hlk$ ssh-copy-id -i /home/hlk/.ssh/id_rsa hlk@10.0.42.147
/usr/local/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/kramse.pub"
The authenticity of host '10.0.42.147 (10.0.42.147)' can't be established.
ECDSA key fingerprint is SHA256:DP6jqadDWEJW/3FYPY84cpTKmEW7XoQ4zDNf/RdTu6M.
Are you sure you want to continue connecting (yes/no)? yes
/usr/local/bin/ssh-copy-id: INFO: attempting to log in with the new key(s),
to filter out any that are already installed
/usr/local/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you
are prompted now it is to install the new keys
hlk@10.0.42.147's password:

Number of key(s) added:        1

Now try logging into the machine, with:   "ssh -o 'IdentitiesOnly yes' 'hlk@10.0.42.147'"
and check to make sure that only the key(s) you wanted were added.
```

OpenSSH konfiguration

Sådan anbefaler jeg at konfigurere OpenSSH SSHD

Det gøres i filen `sshd_config` typisk `/etc/ssh/sshd_config`

```
Port 22780    // eller anden tilfældig port
Protocol 2

PermitRootLogin no
PubkeyAuthentication yes
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no

UseDNS no
#X11Forwarding no
#X11DisplayOffset 10
#X11UseLocalhost yes
```

Det er en smagssag om man vil tillade *X11 forwarding*

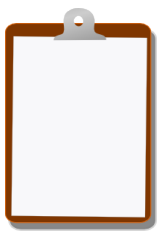
Exercise

Now lets do the exercise

Configure SSH keys for more secure access

which is number **30** in the exercise PDF.

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools