Welcome to

3. SDLC and Risk Ranking

KEA Kompetence VF1 Software Security

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

Slides are available as PDF, kramse@Codeberg

3-SDLC-and-risk-ranking.tex in the repo security-courses

Goals for today

Todays goals:

- Catchup on the last days – too much information?
- Talk about SSDL, SDLC – Secure Development *
- Look a few more examples of real vulnerabilities, can we read the advisories now?

Photo by Thomas Galler on Unsplash

Plan for today

Subjects

- Poor Use of Cryptography
- Basic Cryptography introduction
- Symmetric Cryptosystems
- Data Encryption Standard (DES) / Advanced Encryption Standard (AES)
- Public Key Cryptography
- Stream and Block Ciphers
- Software Development Lifecycle
- Secure Software Development Lifecycle
- Phases of SSDL
- Roles and Responsibilities

Exercises

- Choose a few real vulnerabilities, prioritize them

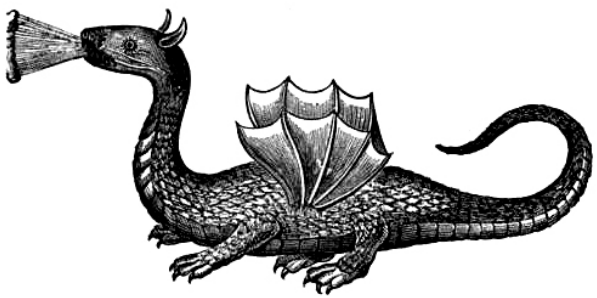
Reading Summary

AoST chapters 3: The Secure Software Development Lifecycle

AoST chapters 4: Risk-based Security Testing

AoST chapters 5: Shades of Analysis: white, Gray, and Black Box Testing

Goals:



Here be dragons

- Software is insecure
- How do we improve quality
- Higher quality is more stable, and more secure
- Make sure to test specifically for security issues

We talked about security design with Qmail and Postfix recently. This year has been bad for Exim mailserver: CVE-2019-10149, CVE-2019-13917 and CVE-2019-15846

Exim RCE CVE-2019-10149 June

VULNERABILITY PATCHED... BY ACCIDENT ...

This was only recently discovered by the Qualys team while auditing older Exim versions. Now, Qualys researchers are warning Exim users to update to the 4.92 version to avoid having their servers taken over by attackers. Per the same June 2019 report on email server market share, only 4.34% of all Exim servers run the latest 4.92 release.

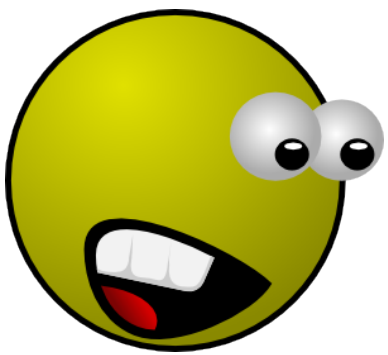
In an email to Linux distro maintainers, Qualys said the vulnerability is "trivially exploitable"and expects attackers to come up with exploit code in the coming days.

This Exim flaw is currently tracked under the CVE-2019-10149 identifier, but Qualys refers to it under the name of "Return of the WIZard"because the vulnerability resembles the ancient WIZ and DEBUG vulnerabilities that impacted the Sendmail email server back in the 90s.

https://www.zdnet.com/article/new-rce-vulnerability-impacts-nearly-half-of-the-internets-email-servers/

See also detailed information from the finders:
https://www.qualys.com/2019/06/05/cve-2019-10149/return-wizard-rce-exim.txt

Exim RCE CVE-2019-10149 July



Issue: A local or remote attacker can execute programs with root privileges - if you've an unusual configuration. For details see below.

https://exim.org/static/doc/security/CVE-2019-13917.txt

Not enabled in default config!

Exim RCE CVE-2019-15846 September

The Exim mail transfer agent (MTA) software is impacted by a critical severity vulnerability present in versions 4.80 up to and including 4.92.1.

The bug allows local or unauthenticated remote attackers to execute programs with root privileges on servers that accept TLS connections.

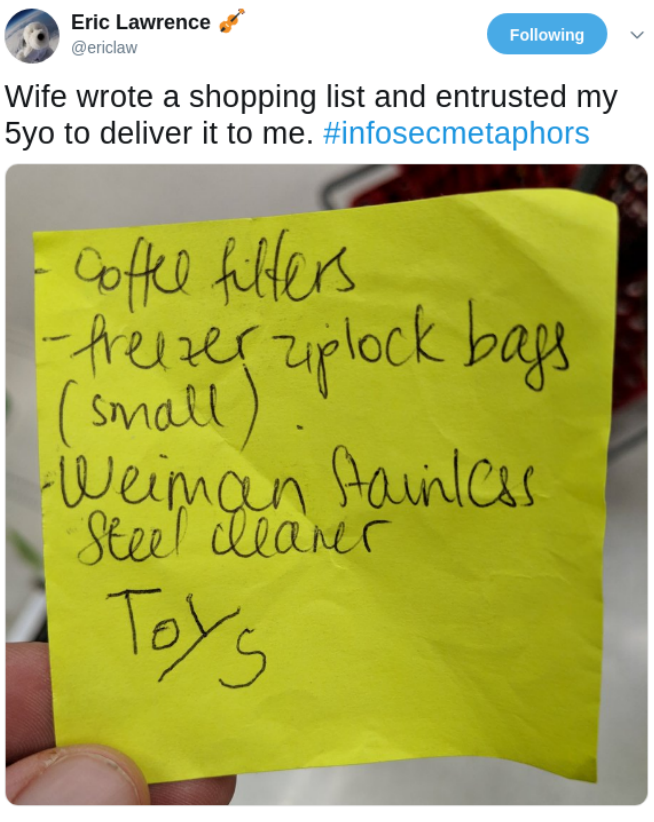
The flaw tracked as CVE-2019-15846 — initially reported by 'Zerons' on July 21 and analyzed by Qualys' research team — is "exploitable by sending an SNI ending in a backslash-null sequence during the initial TLS handshake"which leads to RCE with root privileges on the mail server.

https://www.bleepingcomputer.com/news/security/critical-exim-tls-flaw-lets-attackers-remotely-execute-commands-as-r

https://git.exim.org/exim.git/blob_plain/2600301ba6dbac5c9d640c87007a07ee6dcea1f4:/doc/doc-txt/cve-2019-15846/cve.txt

Basic cryptography

- Confidentiality - data ket secret from prying eyes
- Integrity - data is not changed by unauthorized programs pr people
- A common attack category is children intercepting messages
- often called MiTM (Man in the middle) – Mini in the Middle in this case



WEP design major cryptographic errors

weak keying - 24 bit already known - 128-bit = 104 bit really

small initialisation vector (IV) - only 24 bit, every IV will be reused more often

CRC-32 integrity check NOT *strong* enough cryptographically

Authentication gives pad - if you get one *encryption pad* for one IV you can produce packets forever

Source: *Secure Coding: Principles and Practices*, Mark G. Graff and Kenneth R. van Wyk, O'Reilly, 2003

Example of a technology that people depended upon, https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy

Poor Use of Cryptography

Common pitfalls

- Creating Your Own Cryptography
  Its easy to create something you cannot break, but that is not neccessarily secure
- Choosing the Wrong Cryptography - book recommend FIPS, lots of internet resources recommend NOT to use FIPS!
  Times changes, follow and read up
- Relying on Security by Obscurity
- Hard-Coded Secrets / Mishandling Private Information - if your mobile app binary contains a private key, and is being distributed to millions of users, is it really private - no

Cryptography is hard!

A Graduate Course in Applied Cryptography By Dan Boneh and Victor Shoup
https://toc.cryptobook.us/
https://crypto.stanford.edu/~dabo/cryptobook/BonehShoup_0_4.pdf

Basic Cryptography introduction

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

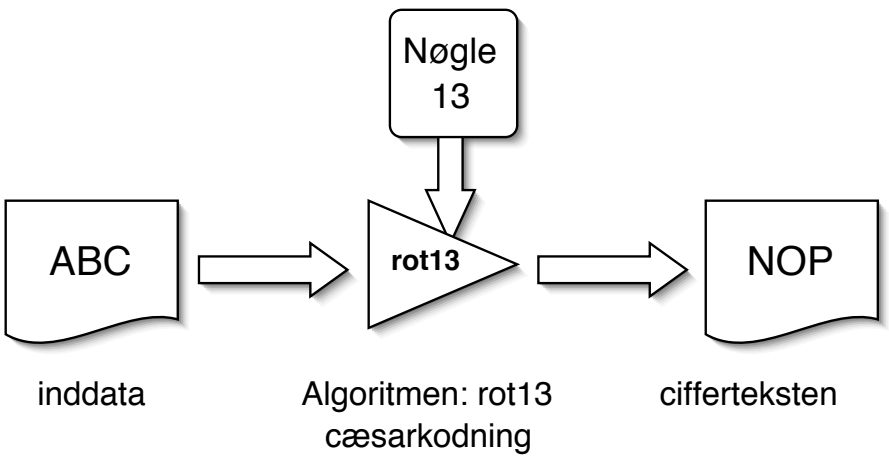
Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key, to ensure confidentiality, example algorithm AES

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

Encryption Decryption

Kryptografi

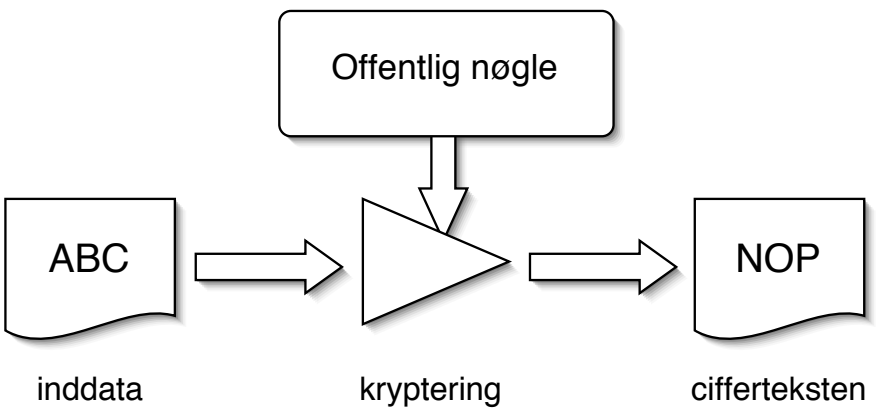


Kryptografi er læren om, hvordan man kan kryptere data

Kryptografi benytter algoritmer som sammen med nøgler giver en ciffertekst

- der kun kan læses ved hjælp af den tilhørende nøgle

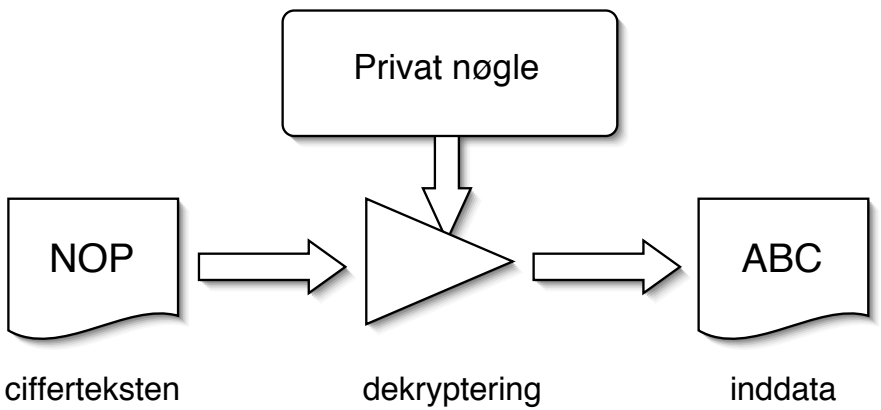
Public key kryptografi - 1



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

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!

DES, Triple DES og AES

AES

Advanced Encryption Standard

DES kryptering - gammel og pensioneret!

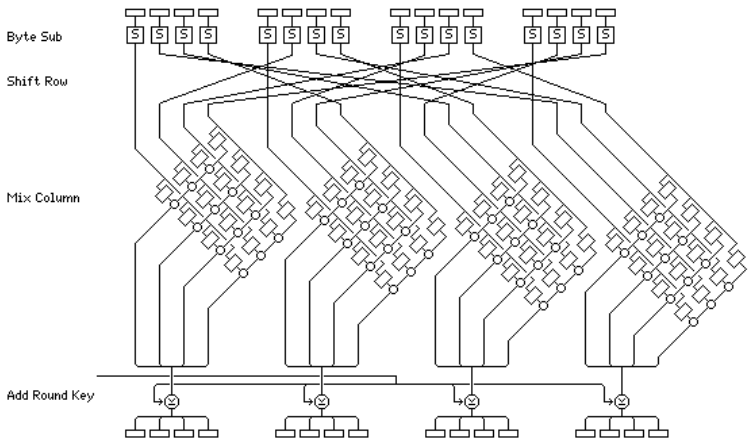
Der blev i 2001 vedtaget en ny standard algoritme Advanced Encryption Standard (AES) som afløser Data Encryption Standard (DES)

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



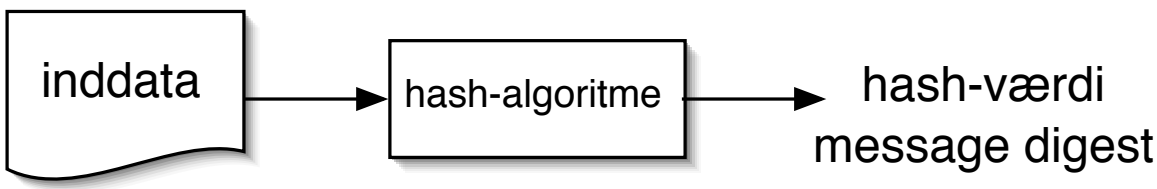
- The official Rijndael web site displays this image to promote understanding of the Rijndael round transformation [8].
- Key sizes 128,192,256 bit typical
- Some extensions in cryptosystems exist: XTS-AES-256 really is 2 instances of AES-128 and 384 is two instances of AES-192 and 512 is two instances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

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)

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

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

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

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/

Pass the hash

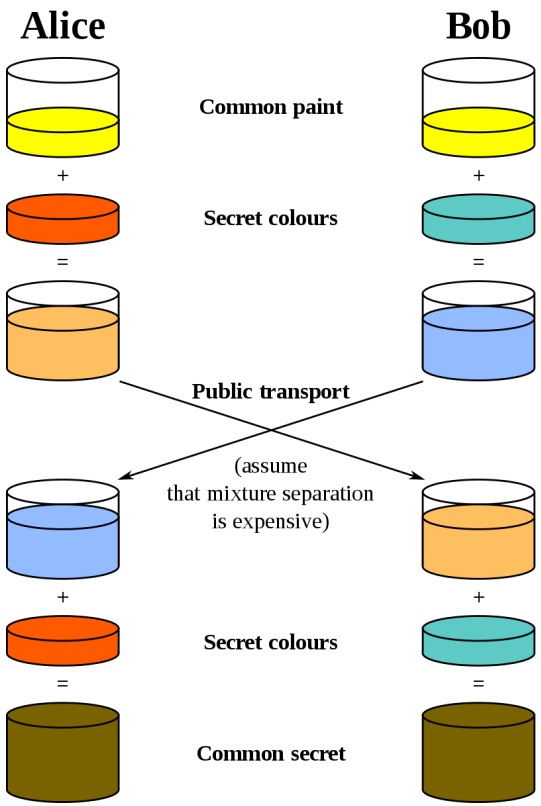
Lots of tools in pentesting pass the hash, reuse existing credentials and tokens *Still Passing the Hash 15 Years Later* http://passing-the-hash.blogspot.dk/2013/04/pth-toolkit-for-kali-interim-status.html

> If a domain is built using only modern Windows OSs and COTS products (which know how to operate within these new constraints), and configured correctly with no shortcuts taken, then these protections represent a big step forward.

Source:
http://www.harmj0y.net/blog/penetesting/pass-the-hash-is-dead-long-live-pass-the-hash/ https://samsclass.info/lulz/pth-8.1.htm

Diffie Hellman 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 also use elliptic curves with DH
  https://en.wikipedia.org/wiki/Elliptic-curve_cryptography

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.

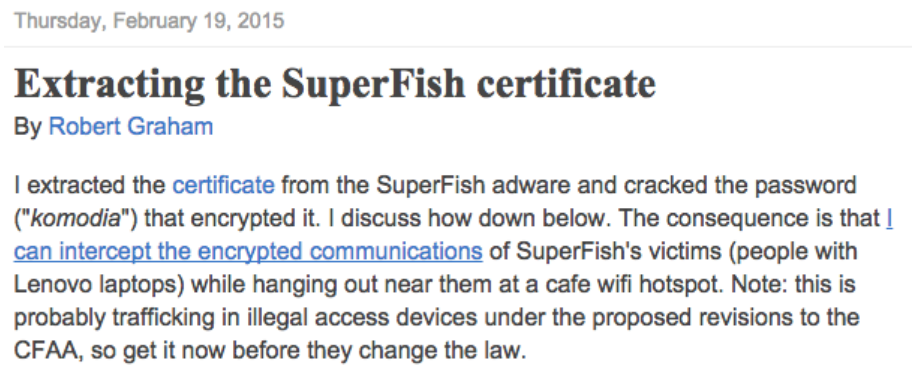
2. **Threats from state-level adversaries.** Millions of HTTPS, SSH, and VPN servers all use the same prime numbers for Diffie-Hellman key exchange. Practitioners believed this was safe as long as new key exchange messages were generated for every connection. However, the first step in the number field 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.

Why?, because things like Superfish February 2015



Thursday, February 19, 2015

Extracting the SuperFish certificate

By Robert Graham

I extracted the certificate from the SuperFish adware and cracked the password ("*komodia*") that encrypted it. I discuss how down below. The consequence is that I can intercept the encrypted communications of SuperFish's victims (people with Lenovo laptops) while hanging out near them at a cafe wifi hotspot. Note: this is probably trafficking in illegal access devices under the proposed revisions to the CFAA, so get it now before they change the law.

Lenovo laptops included Adware, which did SSL/TLS Man in the Middle on connections. They had a root certificate installed on the Windows operating system, WTF!

Sources:

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

http://blog.erratasec.com/2015/02/extracting-superfish-certificate.html

http://www.version2.dk/blog/kibana4-superfish-og-emergingthreats-81610

https://www.eff.org/deeplinks/2015/02/further-evidence-lenovo-breaking-https-security-its-laptops

Elliptic Curve

Public-key cryptography is based on the intractability of certain mathematical problems. Early public-key systems are secure assuming that it is difficult to factor a large integer composed of two or more large prime factors. For elliptic-curve-based protocols, it is assumed that finding the discrete logarithm of a random elliptic curve element with respect to a publicly known base point is infeasible: this is the **"elliptic curve discrete logarithm problem"(ECDLP)**. The security of elliptic curve cryptography depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original and product points. The size of the elliptic curve determines the difficulty of the problem.

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]

- https://en.wikipedia.org/wiki/Elliptic-curve_cryptography
- Has very small key sizes

Transport Layer Security (TLS)

Originally from Netscape Communications Inc.

Secure Sockets Layer (SSL) was adopted by the IETF newer versions are called Transport Layer Security (TLS)

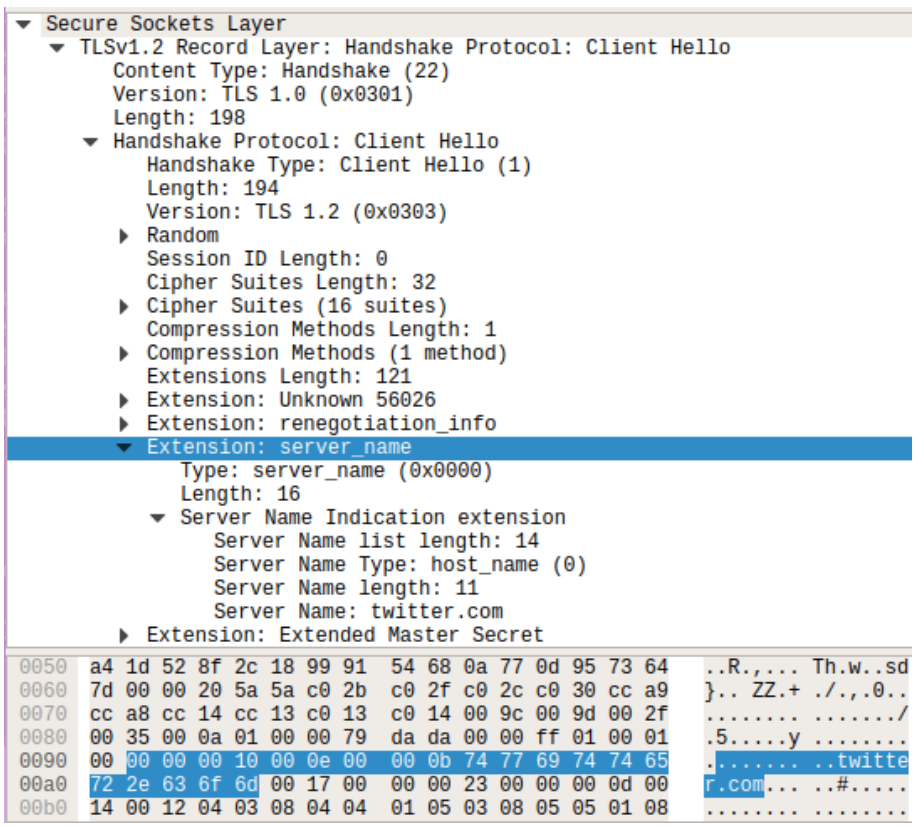
TLS 1.0 based on generalized version of SSL Version 3.0

RFC-2246 The TLS Protocol Version 1.0 from Januar 1999

RFC-3207 SMTP STARTTLS allows the use of TLS with SMTP mail protocols

Today most sites and servers support TLS Server Name Indication (SNI) name based certificates

TLS Server Name Indication example



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.

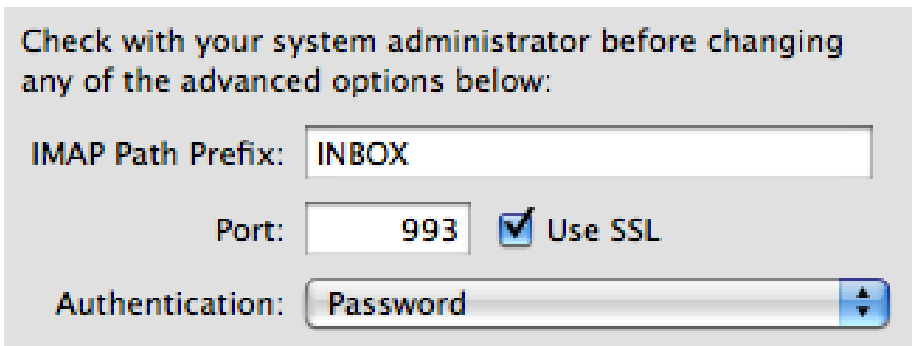
Key points after heartbleed

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

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

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

Exercise

Now lets do the exercise

ℹ SSL/TLS scanners 15 min

which is number **13** in the exercise PDF.

Software Development Lifecycle

> A full lifecycle approach is the only way to achieve secure software.
> –Chris Wysopal

- Often security testing is an afterthought
- Vulnerabilities emerge during design and implementation
- Before, during and after approach is needed

Secure Software Development Lifecycle

- SSDL represents a structured approach toward implementing and performing secure software development
- Security issues evaluated and addressed early
- During business analysis
- through requirements phase
- during design and implementation

Functional specification needs to evaluate security

- Completeness
- Consistency
- Feasibility
- Testability
- Priority
- Regulations

Source: The Art of Software Security Testing Identifying Software Security Flaws Chris Wysopal ISBN: 9780321304865

Phases of SSDL

- Phase 1: Security Guidelines, Rules, and Regulations
- Phase 2: Security requirements: attack use cases
- Phase 3: Architectural and design reviews/threat modelling
- Phase 4: Secure coding guidelines
- Phase 5: Black/gray/white box testing
- Phase 6: Determining exploitability

Secure deployment comes next after this.

Phase 1: Security Guidelines, Rules, and Regulations

- *Umbrella requirement*
- Government regulations Sarbanes-Oxley Act (SOX)
- Payment regulations Payment Card Industry (PCI)
- OWASP, HIPAA, FISMA, BASEL II, ...
- ISO/IEC 27001 - information security management system standards http://en.wikipedia.org/wiki/ISO/IEC_27001
- SSAE 16 No. 16, Reporting on Controls at a Service Organization Statement on Standards for Attestation Engagements (SSAE) http://ssae16.com/
- ISAE 3402 Assurance Reports on Controls at a Service Organization International Standard on Assurance Engagements (ISAE) http://isae3402.com/

Phase 2: Security requirements: attack use cases

MITRE ATT&CK™ is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. The ATT&CK knowledge base is used as a foundation for the development of specific threat models and methodologies in the private sector, in government, and in the cybersecurity product and service community.

With the creation of ATT&CK, MITRE is fulfilling its mission to solve problems for a safer world — by bringing communities together to develop more effective cybersecurity. ATT&CK is open and available to any person or organization for use at no charge.

ATT&CK™

- Does application store personal and/or sensitive information, health, HIPAA, GDPR
- MITRE ATT&CK framework may help https://attack.mitre.org/

Phase 3: Architectural and design reviews/threat modelling

- Help avoid insecure architectures and low-security design
- Threat modelling - a whole subject in itself
- Identify security critical parts of the application

Phase 4: Secure coding guidelines

- Plan use of static and dynamic analysis tools
- Train for secure coding
- Lay down rules for coding, dont use `strcpy` only `strlcpy`

Secure Coding Best Practices Handbook from Veracode

- **#01 Verify for Security Early and Often**
- #02 Parameterize Queries
- #03 Encode Data
- #04 Validate All Inputs
- #05 Implement Identity and Authentication Controls
- #06 Implement Access Controls
- #07 Protect Data
- #08 Implement Logging and Intrusion Detection
- #09 Leverage Security Frameworks and Libraries
- #10 Monitor Error and Exception Handling

https://info.veracode.com/secure-coding-best-practices-hand-book-guide-resource.html

Phase 5: Black/gray/white box testing

- Plan for testing
- Allow time for testing - critical part
- Continous integration may help to avoid pitfalls like, we are out of time – we will skip security testing

Phase 6: Determining exploitability

Ideally every vulnerability would be fixed

Determining exploitability is a factor in estimating risk associated

- Access needed to attempt exploitation
- Level of access or privilege yielded by successful exploitation
- The time or work factor required to exploit the vulnerability
- The exploits potential reliability
- The repeatability of exploit attempts

Deploying Applications Securely

- Having secure defaults helps
- Good initial file permissions
- Make sure application can be patched
- Track and prioritize identified vulnerabilities
- Make it easy to report vulnerabilities to the organization

Roles and Responsibilities

- Make it clear who has responsibility for security at various phases
- Program or product manager should write the security policies
- Product or project manager also responsible for certification processes
- Architects and developers are responsible for providing design and implementation
- QA/testers drive critical analyses of the system and build tests
- Security process managers oversee threat modelling, security assessments, and secure coding training

- Not an exhaustive list!

Risk-Based Security Testing

> Focus testing on areas where difficulty of attack is least and the impact is highest.
> –Chris Wysopal

Time and resources are constrained

Software development must be prioritized

Threat modelling / risk modelling exist to help this

- Identify threat paths
- Identify threats
- Identify vulnerabilities
- Rank/prioritize the vulnerabilities

Sounds easy enough, harder to do

DREAD

> DREAD is part of a system for risk-assessing computer security threats previously used at Microsoft and although currently used by OpenStack and other corporations[citation needed] it was abandoned by its creators [1]. It provides a mnemonic for risk rating security threats using five categories.

The categories are:

- Damage – how bad would an attack be?
- Reproducibility – how easy is it to reproduce the attack?
- Exploitability – how much work is it to launch the attack?
- Affected users – how many people will be impacted?
- Discoverability – how easy is it to discover the threat?

Source: https://en.wikipedia.org/wiki/DREAD_(risk_assessment_model)

but was abandoned by Microsoft

Microsoft Secure Development Lifecycle

There are five major threat modeling steps:

- Defining security requirements.
- Creating an application diagram.
- Identifying threats.
- Mitigating threats.
- Validating that threats have been mitigated. Threat modeling should be part of your routine development lifecycle, enabling you to progressively refine your threat model and further reduce risk.

Sources:
https://www.microsoft.com/en-us/securityengineering/sdl
https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

Example applications from Microsoft

Microsoft has released sample applications.

> Secure Development Documentation Learn how to develop and deploy secure applications on Azure with our sample apps, best practices, and guidance.
>
> Get started Develop a secure web application on Azure

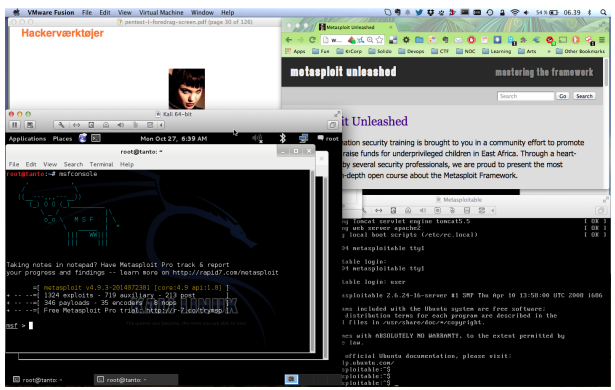
Source: https://docs.microsoft.com/en-us/azure/security/develop/

Yes, this describes how to run Alpine Linux on their Azure Cloud.

Blackbox, greybox og whitebox

- Forudsætninger og forudgående kendskab til miljøet
- Black Box testen involverer en sikkerhedstestning af et netværk uden nogen form for insider viden om systemet udover den IP-adresse, der ønskes testet. Dette svarer til den situation en fjendtlig hacker vil stå i og giver derfor det mest realistiske billede af netværkets sårbarhed overfor angreb udefra. Men er dårlig ressourceudnyttelse.
- I den anden ende af skalaen har vi White Box testen. I dette tilfælde har sikkerhedsspecialisten både før og under testen fuld adgang til alle informationer om det scannede netværk. Analysen vil derfor kunne afsløre sårbarheder, der ikke umiddelbart er synlige for en almindelig angriber. En White Box test er typisk mere omfattende end en Black Box test og forudsætter en højere grad af deltagelse fra kundens side, men giver en meget detaljeret og tilbundsgående undersøgelse.
- En Grey Box test er som navnet siger et kompromis mellem en White Box og en Black Box test. Typisk vil sikkerhedsspecialisten udover en IP-adresse være i besiddelse af de mest grundlæggende systemoplysninger: Hvilken type af server der er tale om (mail-, webserver eller andet), operativsystemet og eventuelt om der er opstillet en firewall foran serveren.

Testing Labs



- Sniffers Wireshark and similar tools
- Proxies and fuzzers
- Debuggers
- Virtualisation - can also emulate ARM on Intel etc.
- Laptops and network hardware - dont use a HUB! Cheap managed switch with mirror port is better

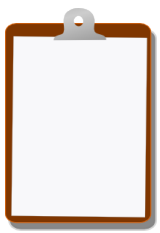
Exercise

Now lets do the exercise

⚠ Real Vulnerabilities up to 30min

which is number **14** 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