# Security in Mind

www.security-in-mind.com

BY DANIEL ELLEBÆK, 2020

# Agenda

- Who am I?
- What is security?
- What is front- and back-end?
- Is there security on the front-end?
- Back-end security
- How to communicate security?
- Taint analysis simplified

Daniel Ellebæk, 40 år

**Education**

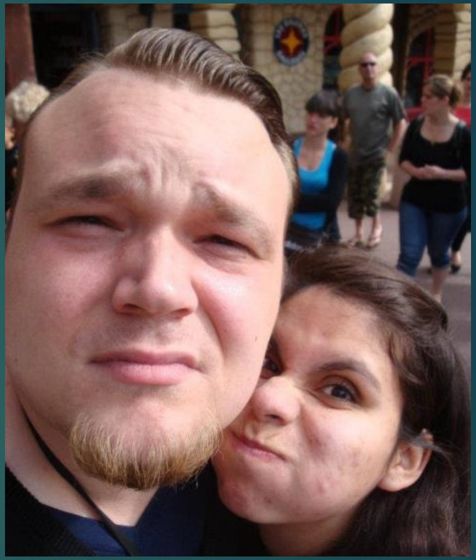
- Aalborg Universitet - Master i IT /Cyber Sikkerhed (present)

**Online**

- YouTube: https://bit.ly/3uEwwtH (Security In Mind)
- Web: www.security-in-mind.com

**Projects:**

- Incorportate IT Security in Danish educations



# What is security?

# Why secure software?

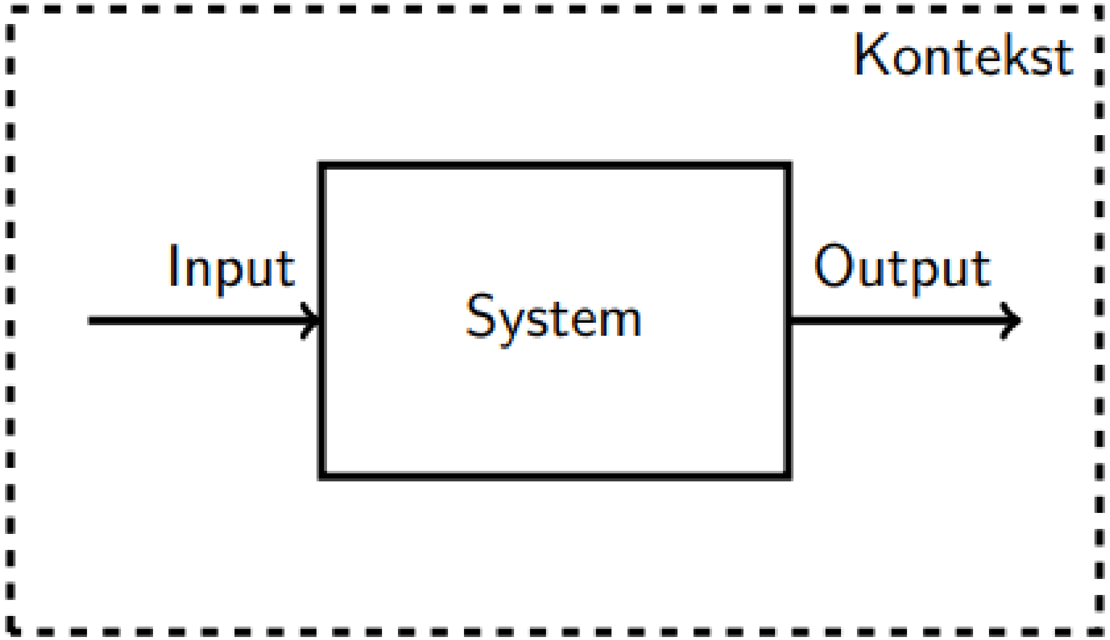
We wouldn’t have to spend so much time, money, and effort on network security if we didn’t have such bad software security.

— Bruce Schneier

# What security is…

- Network Analysis
- Penetration Testing
- Incident Response
- ISMS (information security management system)  Implementation
- Code Analysis
- Secure Programming

# What is security?



**Definition (Security)**
The ability of a **system** to satisfy its **goals** in the presence of an **adversary**

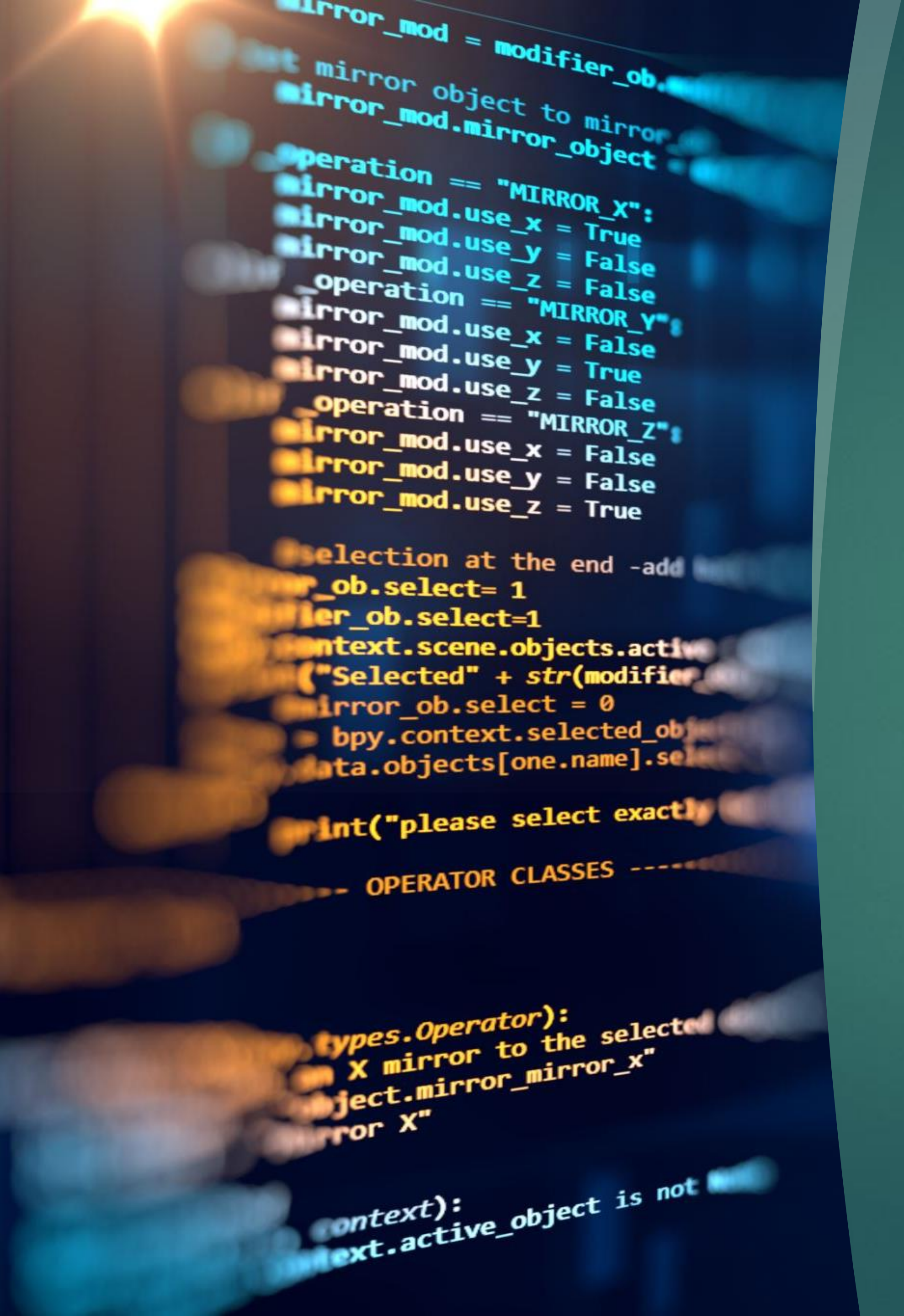
# What is front- and back-end?

# What is front-end?

- JavaScript
- HTML
- CSS
- Images
- Text

**Spend 2 minutes in groups of 2-5 and discuss the following:**

Does security exist on the front-end and why?



# Is there security on the front-end?

# Front-end security?

- Short answer, no.
  - The front-end is all about HTML, CSS, JS, Images, Text…
- Why is a JS solution not secure?
  - JS can be altered on the front-end
  - Never trust front-end data
  - Treat front-end data as tainted data
- Long answer, still no… however it is possible to achieve some XSS. (next page)

# Front-end XSS (a bit abstract)

- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-11022
  - This exact vulnerability can create what is called a front-end XSS vulnerability
    - An example of this could be a vulnerability in a JavaScript library where the "document.location" is used to generate the content of the site
    - Exploit example: https://site/page?parm=foo#xss-goes-here
  - A fix would be:
    - To sanitize the data in the JavaScript

Its not about running the exploit, its about knowing how to stop it.

# What is back-end?

- C#, PHP, Java, C, Python, etc…
- Database
- Server where stuff runs

**Spend 2 minutes in groups of 2-5 and discuss the following:**

Does security exist on the back-end and why?

# How to communicate security?

# Communication is hard

- C.I.A. triad – A way to communicate security

- Confidentiality
  - Keep them secrets secret…
  - Ex. passwords, PII (personal identifiable information)
- Integrity
  - Secure/preserve authenticity of data
  - Only authorized changes to data allowed
  - Ex. bank account, personal data
- Availability
  - Ensure that (legitimate) users can access data in a timely manner
  - Ex. NemID

# Taint analysis simplified

# What is Taint analysis?

- A simple way to analyse your code
  - To find untrusted sources
  - To write better code

A static taint analysis is done by hand/eye. It is a time-consuming process, so better do it while you write code.

https://psalm.dev/

# Taint analysis – 3 simple rules

## Source

- Input from front-end (URL variables, Form data, JS sent data etc…)

## Sanitizer or filter

- A function/method that cleans untrusted data. Makes it trustable.

## Sink

- Output to the user.
- Send data out of your own context.

# Source & sink example

```
<?php
echo $_GET[’name’];
```

- $_GET[‘name’] is direct input from the front-end
  - Source
- echo is outputting data away from your context
  - Sink

# Sanitizer / filter example

```
<?php
echo $_GET[’name’];
```

- The above is not safe.
- Use a function/method to sanitize/filter the data before outputting it.

```
<?php
echo sanitize($_GET[’name’]);
```

# Sanitizer / filter explanation

```
<?php
echo sanitize($_GET[’name’]);
```

- Replace the function sanitize() with the appropriate context-based output sanitizer
  - What is your output?
    - Internet Browser then convert your output to HTML Entities
    - Depending on what you output to you need to sanitize accordingly

Read more: https://www.php.net/manual/en/filter.filters.sanitize.php

# Taint analysis example

```
<?php // --taint-analysis

function getName() : string {
    return $_GET[’name'] ?? 'unknown';
}

function sayHello() : string {
    return 'Hello ' . getName();
}
?>
<!– Outputting to the users (front-end) -->
<h1><?= sayHello() ?></h1>
```

## Is the code tainted?

- If yes, why?
- Is no, why?

# Thanks for listening

# Resources

- https://tryhackme.com
- https://owasp.org/www-project-top-ten/
- https://www.nist.gov/cyberframework
- www.security-in-mind.com
- https://bit.ly/3uEwwtH (Security In Mind : YouTube)