Moduldokumentation

Modul Applikationssicherheit (apsi)

Simon Wächter

2019

Inhalt

[1 Einleitung 3](#_Toc21186590)

[1.1 Einleitung 3](#_Toc21186591)

[1.2 Lernziele 3](#_Toc21186592)

[1.3 Prüfungen 3](#_Toc21186593)

[2 Woche 1 4](#_Toc21186594)

[2.1 Agenda 4](#_Toc21186595)

[2.2 Administrative Issues 4](#_Toc21186596)

[2.3 Who is Your Lecturer (Wagner) 4](#_Toc21186597)

[2.4 whoami (Schläpfer) 5](#_Toc21186598)

[2.5 Why is Application Security Important? 5](#_Toc21186599)

[2.6 Lecture Overview (1) 6](#_Toc21186600)

[2.7 Lecture Overview (2) 6](#_Toc21186601)

[2.8 Lecture Overview (3) 7](#_Toc21186602)

[2.9 Example: Buffer Overflow 7](#_Toc21186603)

[2.10 How does a Buffer Overflow Work? 7](#_Toc21186604)

[2.11 Buffer Overflow Demo 8](#_Toc21186605)

[2.12 Not Really a Fix for Buffer Overflow: Stack Protection 8](#_Toc21186606)

[2.13 Fix for Buffer Overflow: Checked Buffers 8](#_Toc21186607)

[2.14 Example: Taint Checking in Perl (1) 9](#_Toc21186608)

[2.15 Example: Taint Checking (2) 9](#_Toc21186609)

[2.16 Example: Taint Checking (3) 10](#_Toc21186610)

[2.17 Example: Taint Checking (4) 10](#_Toc21186611)

[2.18 Example: Data Leakage by Behavior (1) 10](#_Toc21186612)

[2.19 Example: Data Leakage by Behavior (2) 11](#_Toc21186613)

[2.20 Example: Data Leakage by Error Output 11](#_Toc21186614)

[2.21 Human Factors 12](#_Toc21186615)

[2.22 Exercises and Preparation for Next Week 12](#_Toc21186616)

[2.23 Reserve 12](#_Toc21186617)

[3 Woche 2 13](#_Toc21186618)

[3.1 Agenda 13](#_Toc21186619)

[3.2 Recap: What do Attackers Want? 13](#_Toc21186620)

[3.3 Buffer Overflow II 13](#_Toc21186621)

[3.4 Buffer Overflow II 14](#_Toc21186622)

[3.5 Buffer Overflow II 14](#_Toc21186623)

[3.6 Buffer Overflow II 15](#_Toc21186624)

[3.7 Buffer Overflow II 15](#_Toc21186625)

[3.8 Attack Technique: Fuzzing 16](#_Toc21186626)

[3.9 Error Handling 16](#_Toc21186627)

[3.10 Case study – Handling too long input 17](#_Toc21186628)

[3.11 Case-study: Compression Bomb 17](#_Toc21186629)

[3.12 Error Handling 18](#_Toc21186630)

[3.13 Algorithmic Complexity Attacks 19](#_Toc21186631)

[3.14 Algorithmic Complexity: Sorting 19](#_Toc21186632)

[3.15 Algorithmic Complexity: Hashing 20](#_Toc21186633)

[3.16 Algorithmic Complexity: Regular Expressions 20](#_Toc21186634)

[4 Woche 3 21](#_Toc21186635)

[4.1 Agenda 21](#_Toc21186636)

[4.2 Secure Coding Principle: Test-before-use 21](#_Toc21186637)

[4.3 Input Validation 21](#_Toc21186638)

[4.4 Input Normalization 22](#_Toc21186639)

[4.5 Unix Permission Model (simplified) 22](#_Toc21186640)

[4.6 Chroot 23](#_Toc21186641)

[4.7 Mandatory Access Control (MAC) 23](#_Toc21186642)

[4.8 MAC Example: SELinux 24](#_Toc21186643)

[4.9 MAC Example: AppArmor 24](#_Toc21186644)

[4.10 Secure Coding Principle: Defense in Depth 25](#_Toc21186645)

[4.11 Privilege Separation 25](#_Toc21186646)

[4.12 Overall Idea 26](#_Toc21186647)

[4.13 Form of the Modules 26](#_Toc21186648)

[4.14 Example: OpenSSH 27](#_Toc21186649)

[4.15 OpenSSH: Privilege Separation Approach 27](#_Toc21186650)

[4.16 Example: Postfix MTA (Email Server) 27](#_Toc21186651)

[4.17 Postfix System View 28](#_Toc21186652)

[4.18 Postfix: Example Data-Flow 28](#_Toc21186653)

[5 Woche 4 29](#_Toc21186654)

# Einleitung

## Einleitung

Dieses Dokument stellt die Moduldokumentation für das Modul apsi dar. Allfällige Unterlagen sind im Modulordner zu finden.

## Lernziele

Das Modul beinhaltet folgende Lernziele:

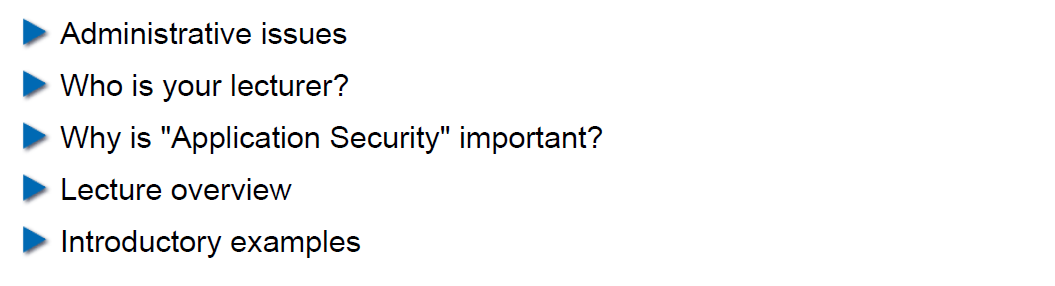
* Kennen typische Bedrohungen von Applikationen sowie ihren Daten und können abschätzen, wieweit diese für durch sie zu entwickelnde Software ein Sicherheitsrisiko darstellen
* Kennen typische Sicherheitsprobleme und können sie vermeiden
* Sind in der Lage, einschlägige Quellen zu konsultieren, um neu veröffentlichte Sicherheitsprobleme zu erfassen und zu verstehen, wie diese zu vermeiden sind
* Verfügen über Konzeptwissen, nicht nur Produktwissen, um das Gelernte auch in anderen Umfeldern und mit anderen Programmiersprachen anwenden zu können

## Prüfungen

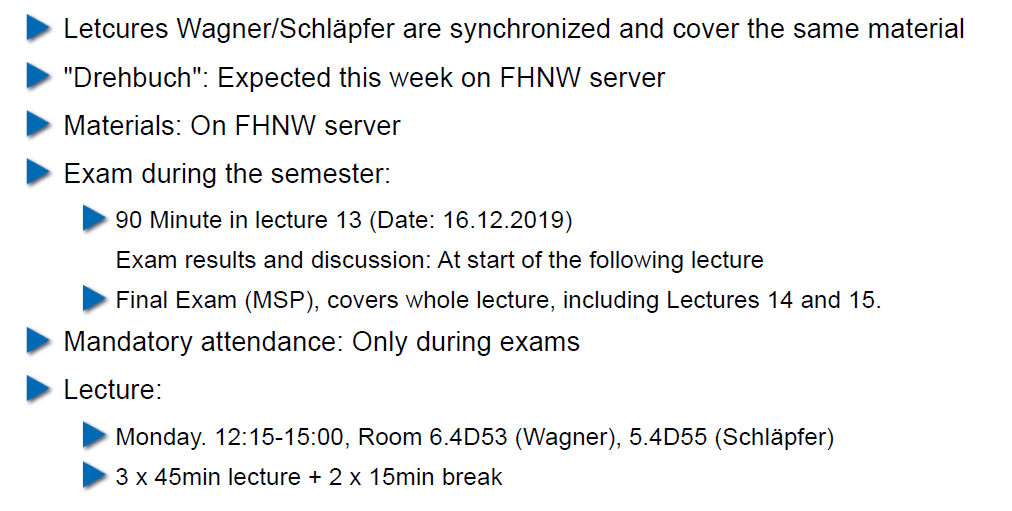
Die Modulnote setzt sich zu 100% aus einer Semesterprüfung und einer Modulschlussprüfung zu je 50% zusammen.

# Woche 1

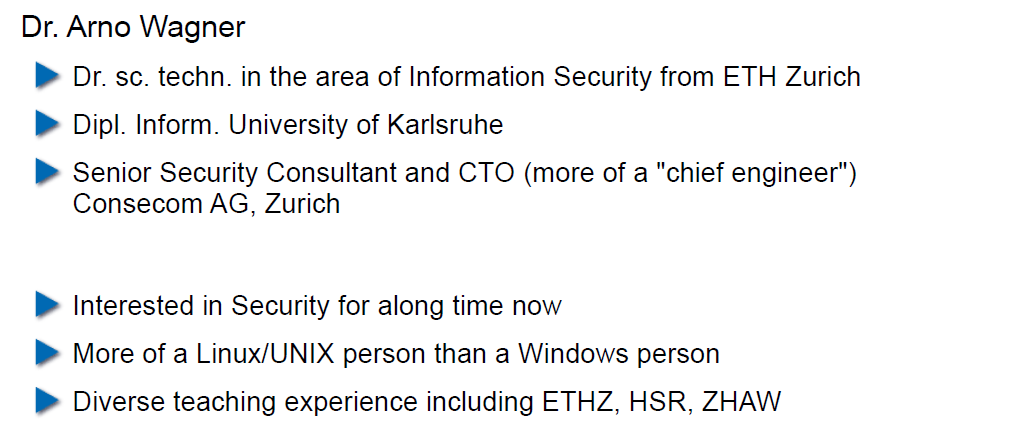
## Agenda



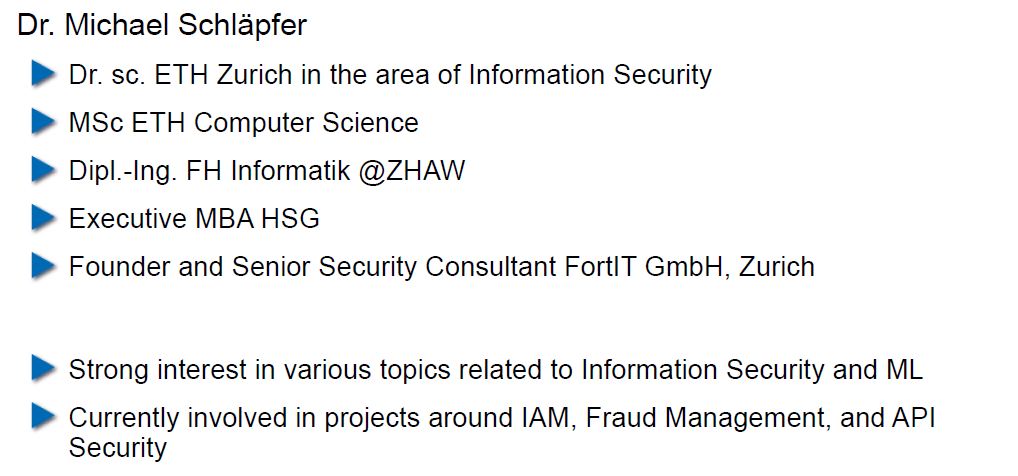
## Administrative Issues



## Who is Your Lecturer (Wagner)



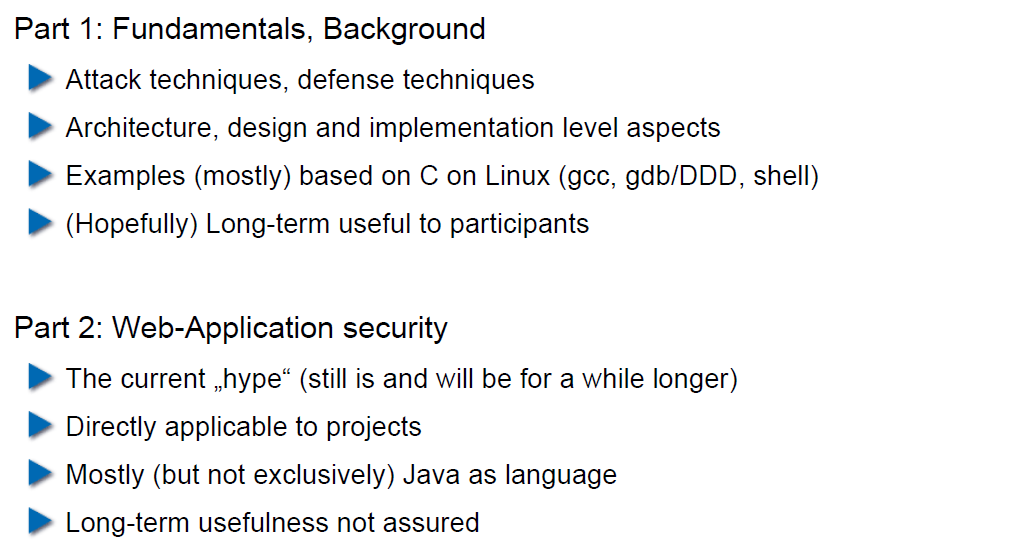
## whoami (Schläpfer)



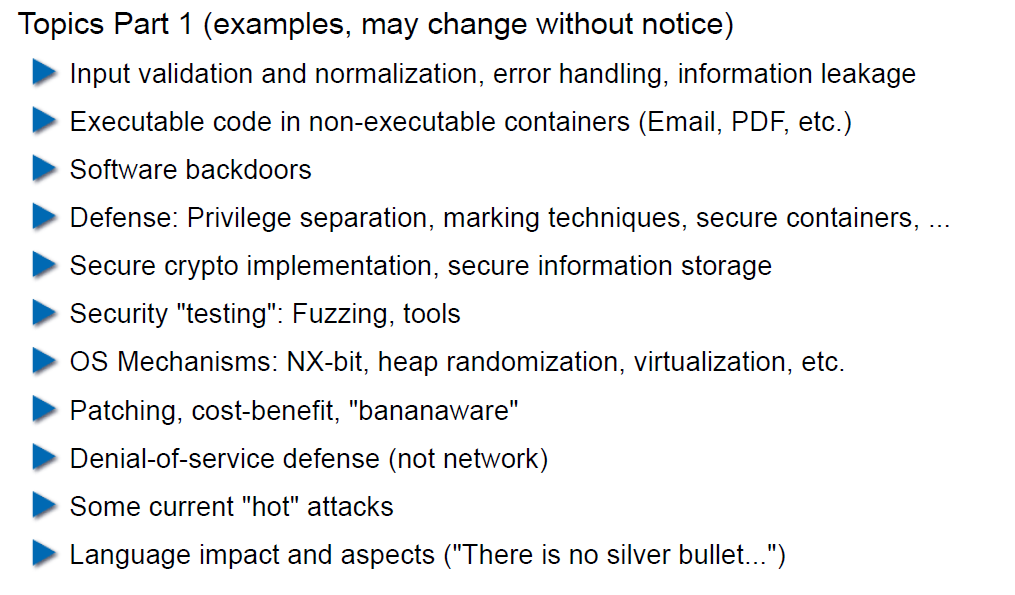
## Why is Application Security Important?



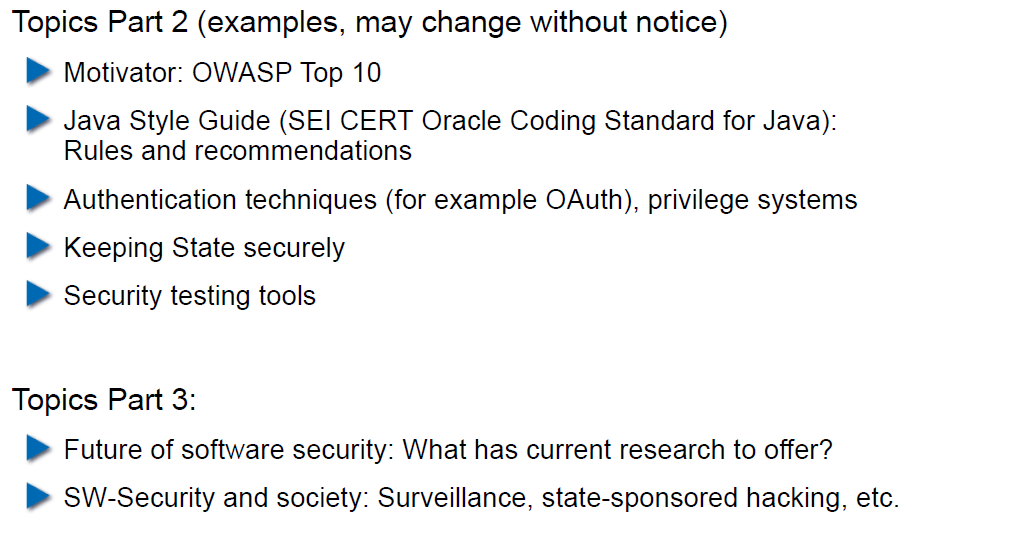
## Lecture Overview (1)



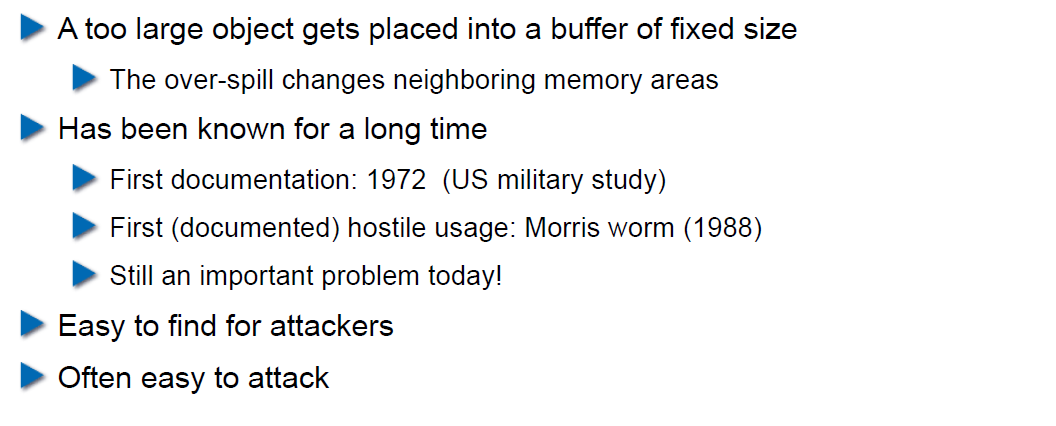
## Lecture Overview (2)



## Lecture Overview (3)



## Example: Buffer Overflow



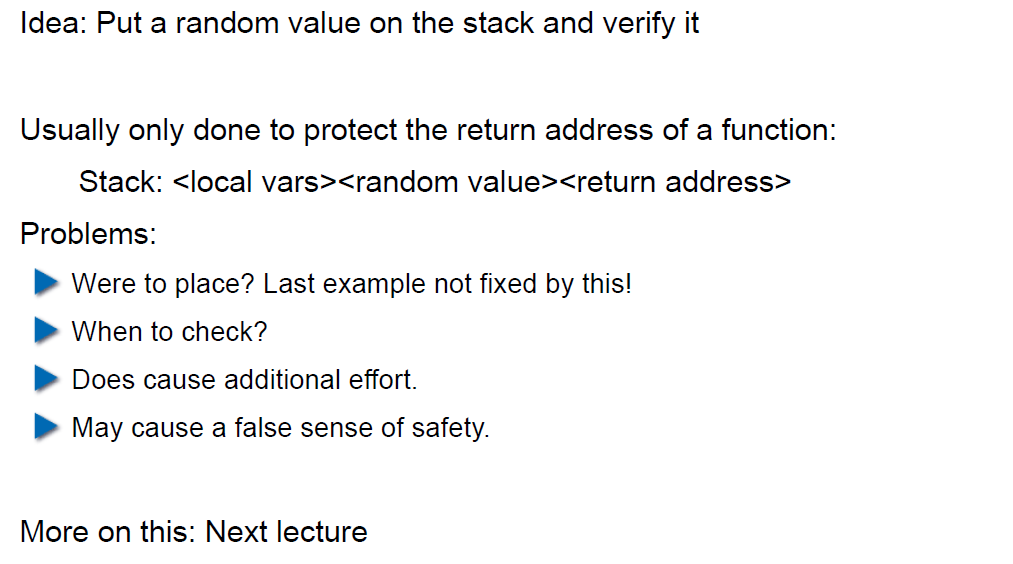
## How does a Buffer Overflow Work?



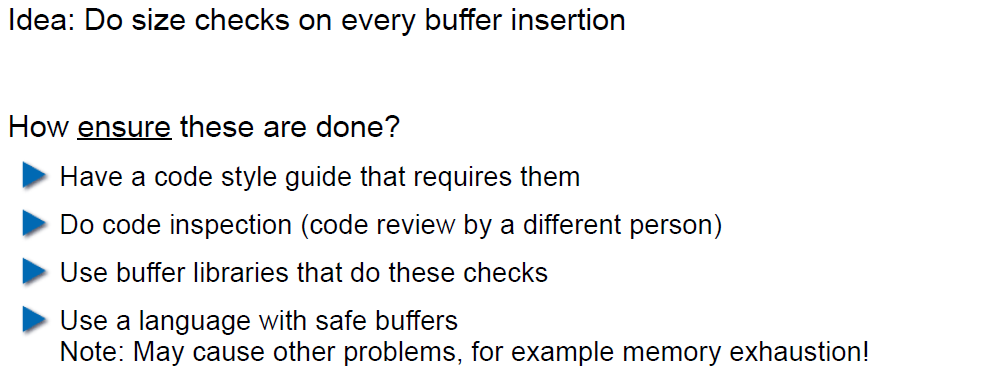
## Buffer Overflow Demo



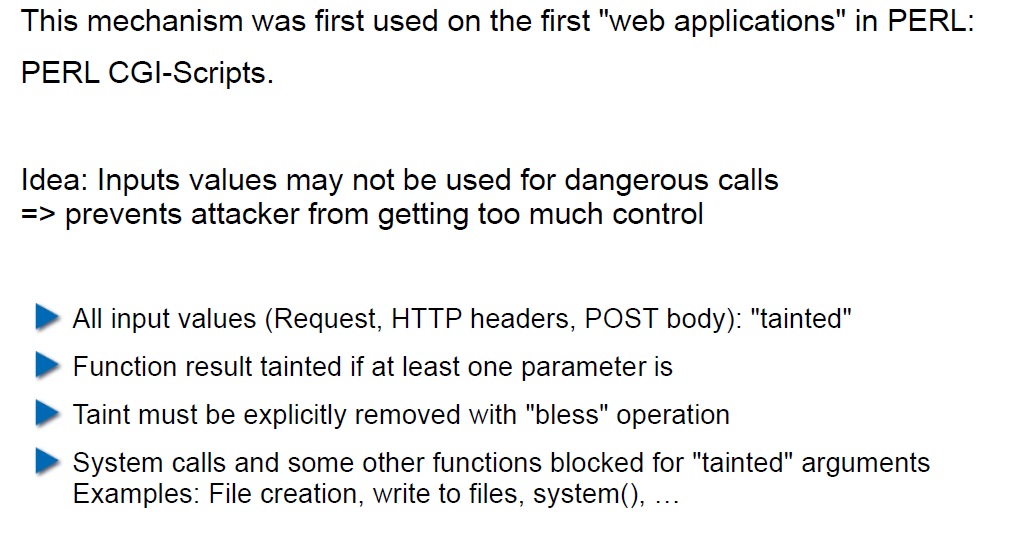
## Not Really a Fix for Buffer Overflow: Stack Protection



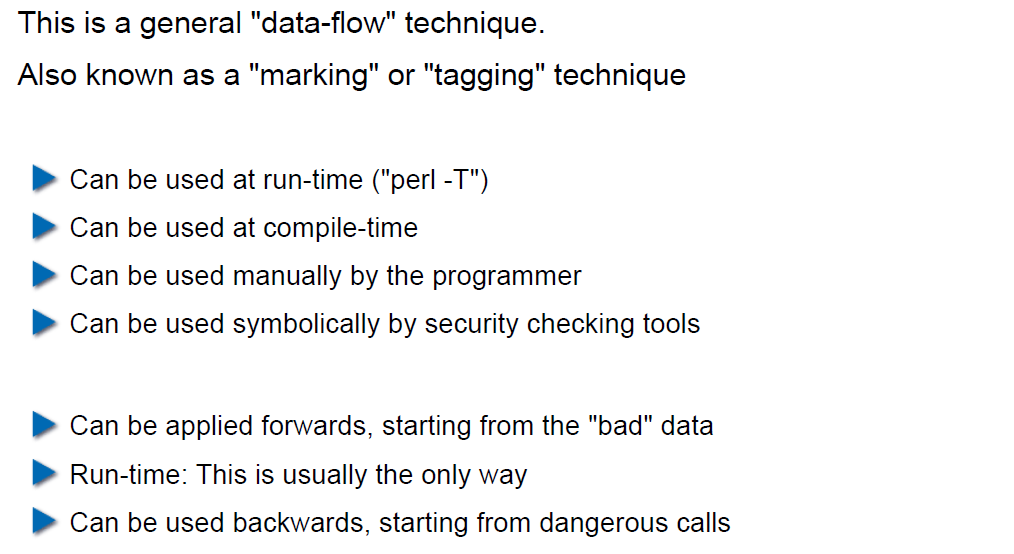
## Fix for Buffer Overflow: Checked Buffers



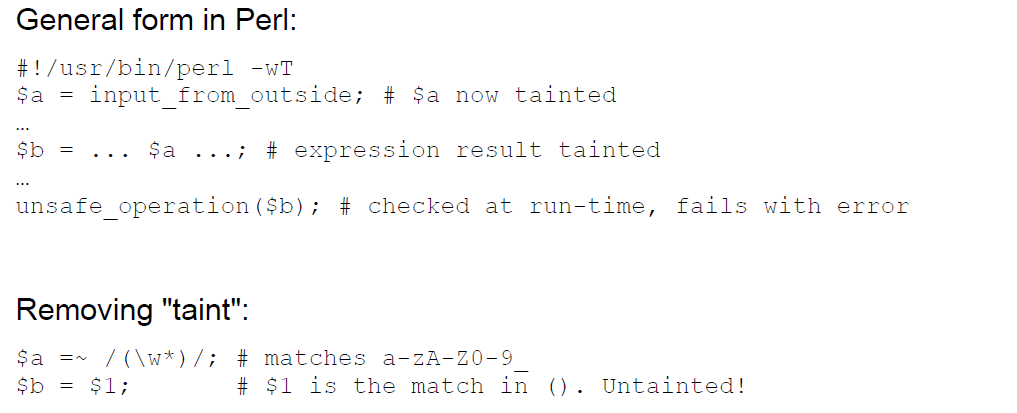
## Example: Taint Checking in Perl (1)



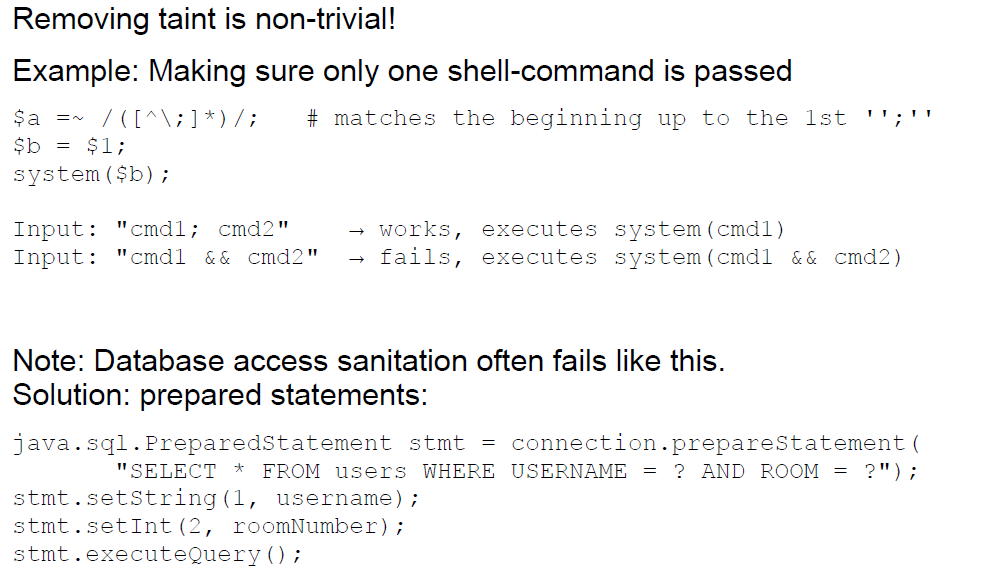
## Example: Taint Checking (2)



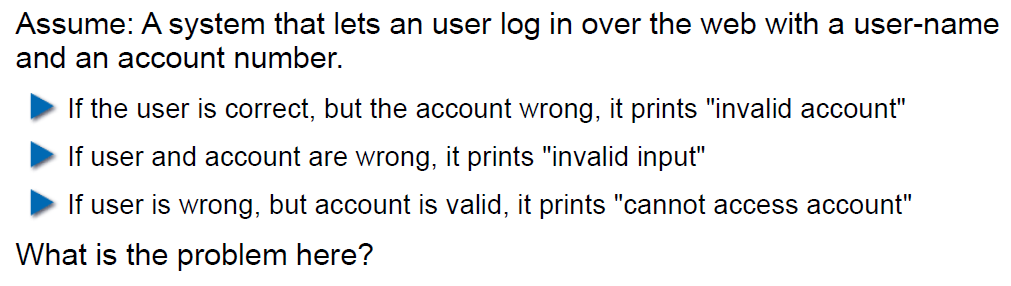
## Example: Taint Checking (3)



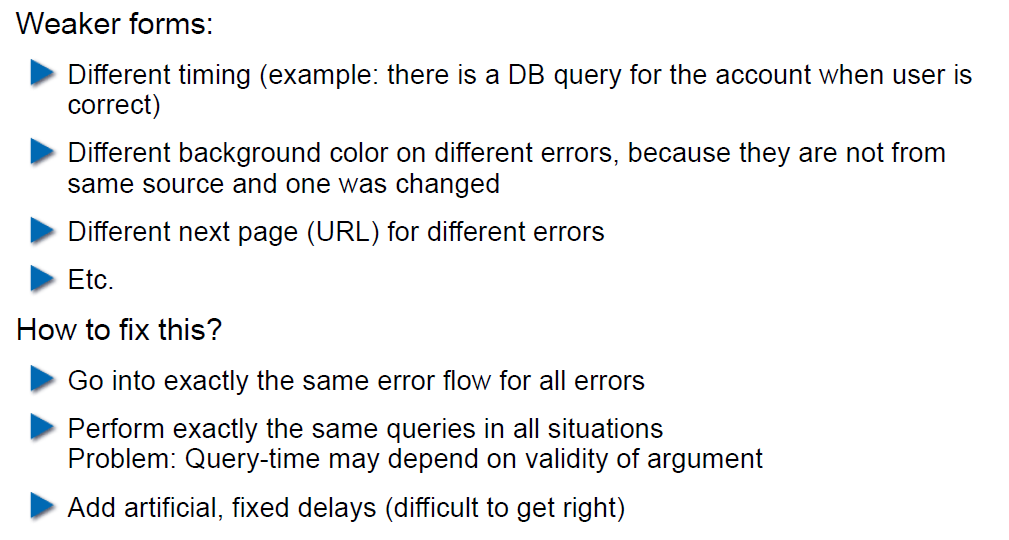
## Example: Taint Checking (4)



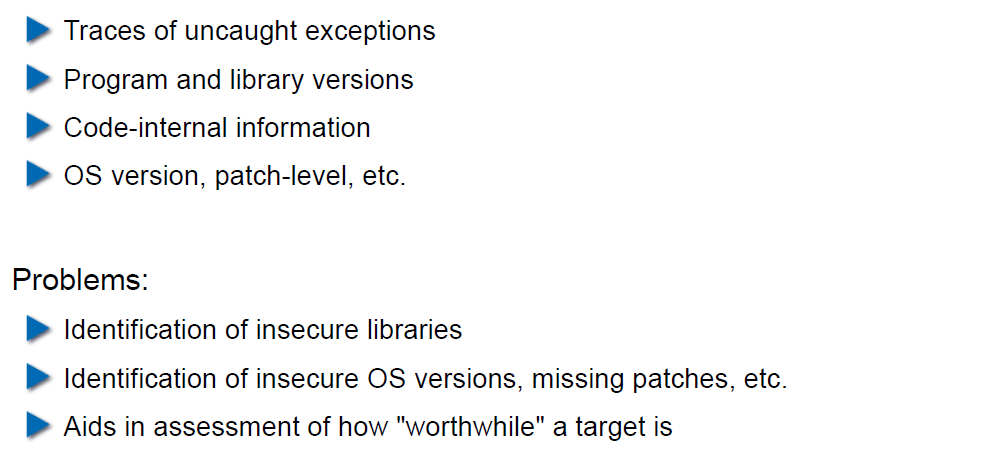
## Example: Data Leakage by Behavior (1)



## Example: Data Leakage by Behavior (2)



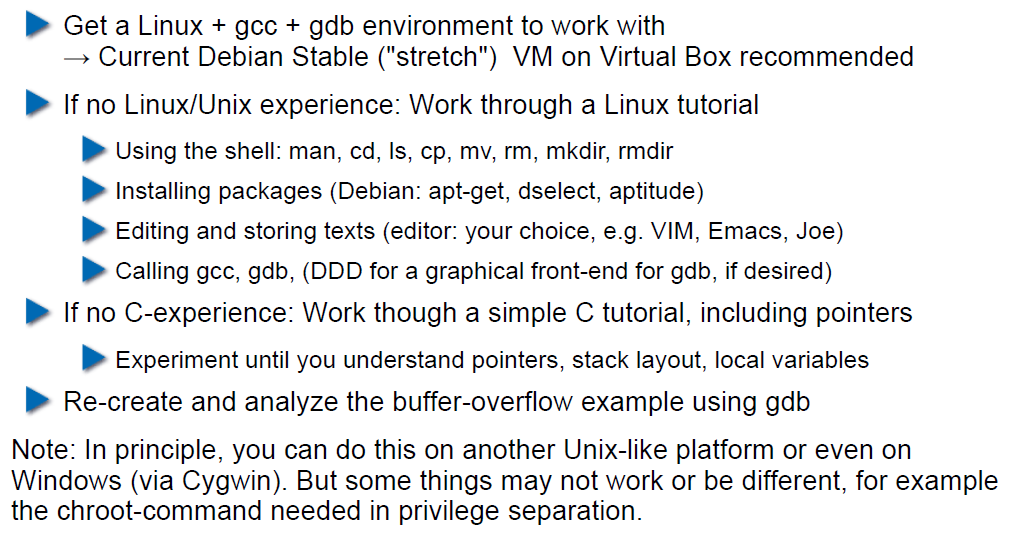
## Example: Data Leakage by Error Output



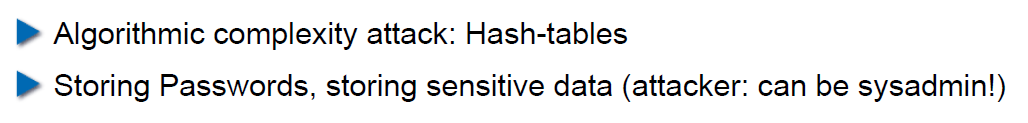
## Human Factors



## Exercises and Preparation for Next Week

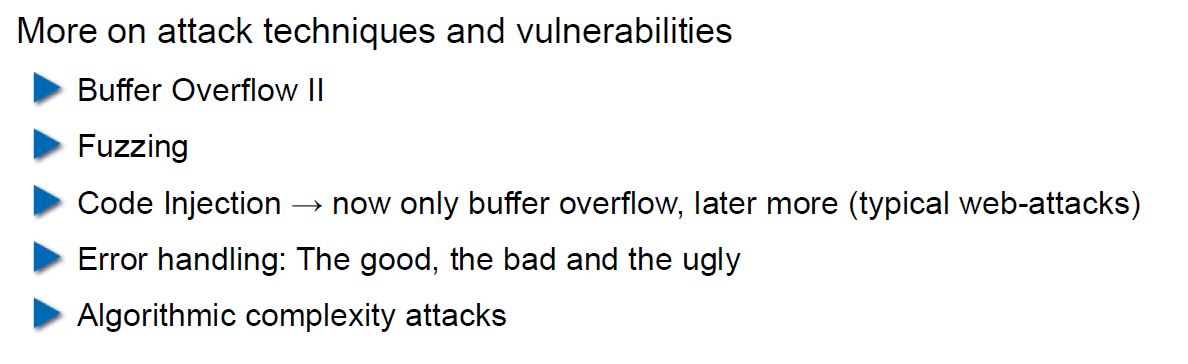


## Reserve

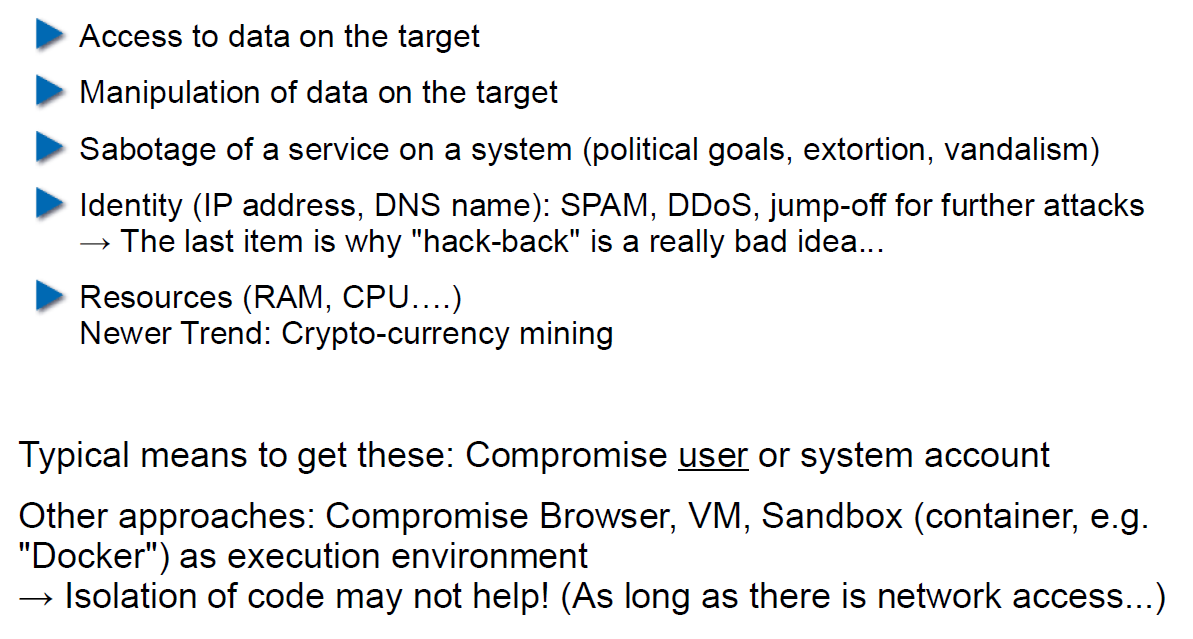


# Woche 2

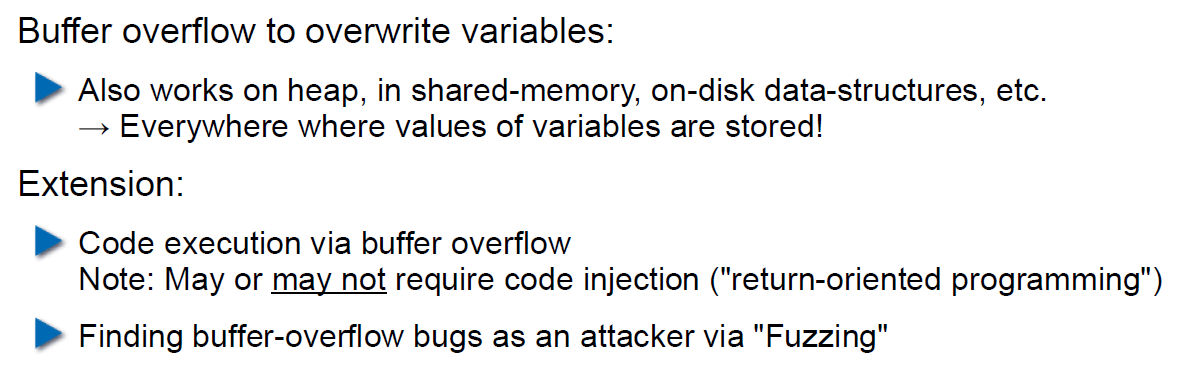
## Agenda



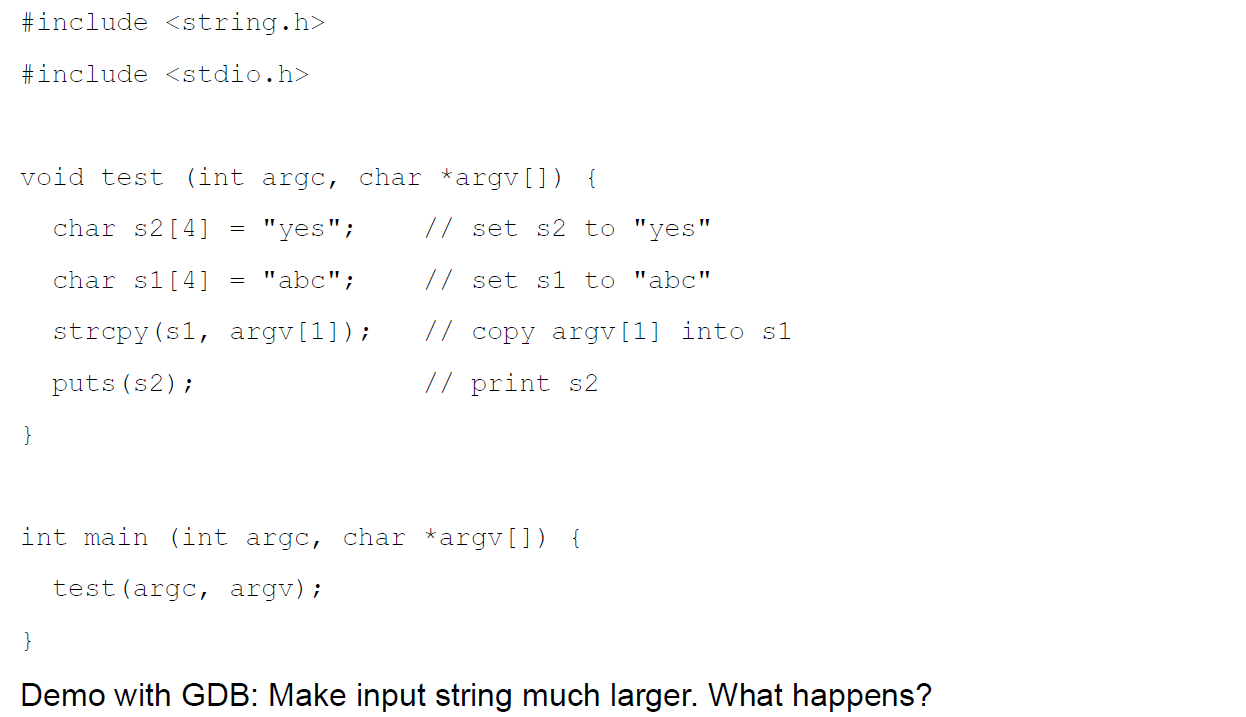
## Recap: What do Attackers Want?



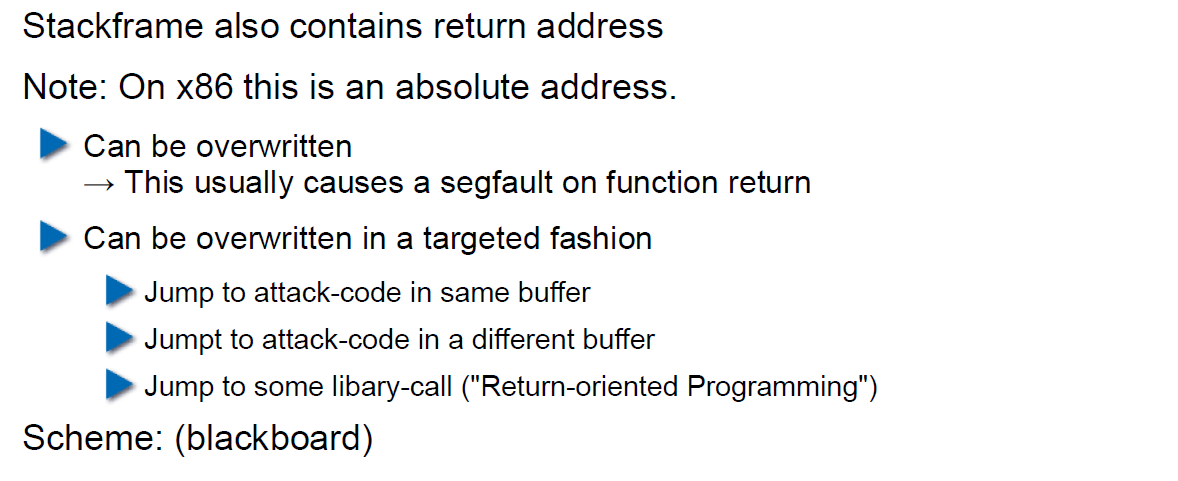
## Buffer Overflow II



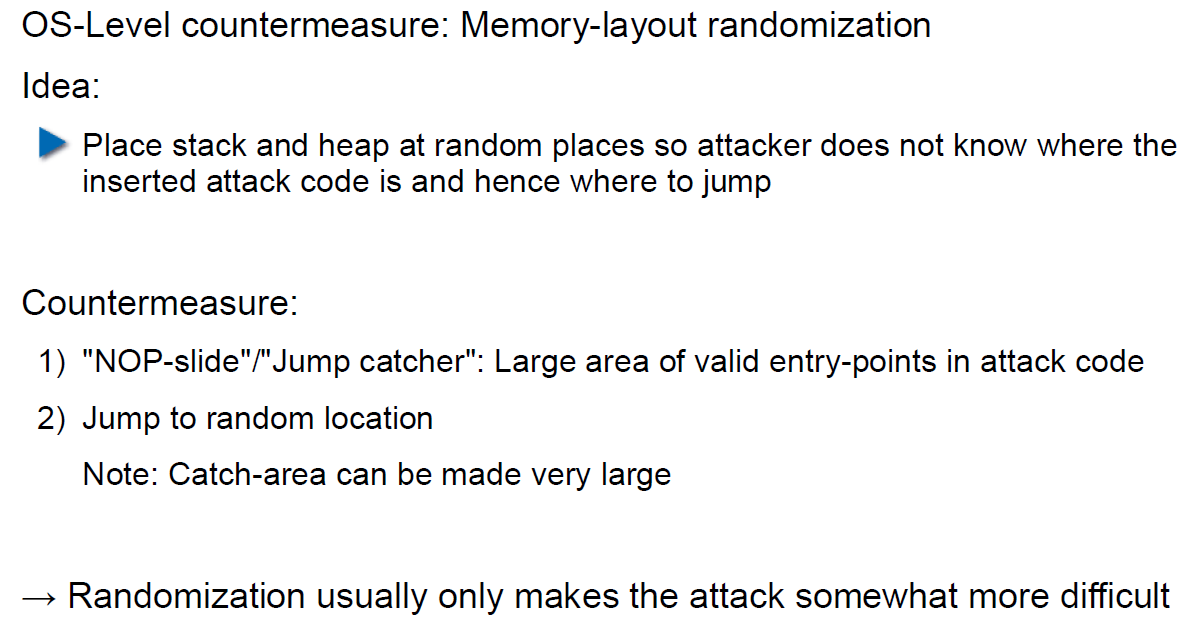
## Buffer Overflow II



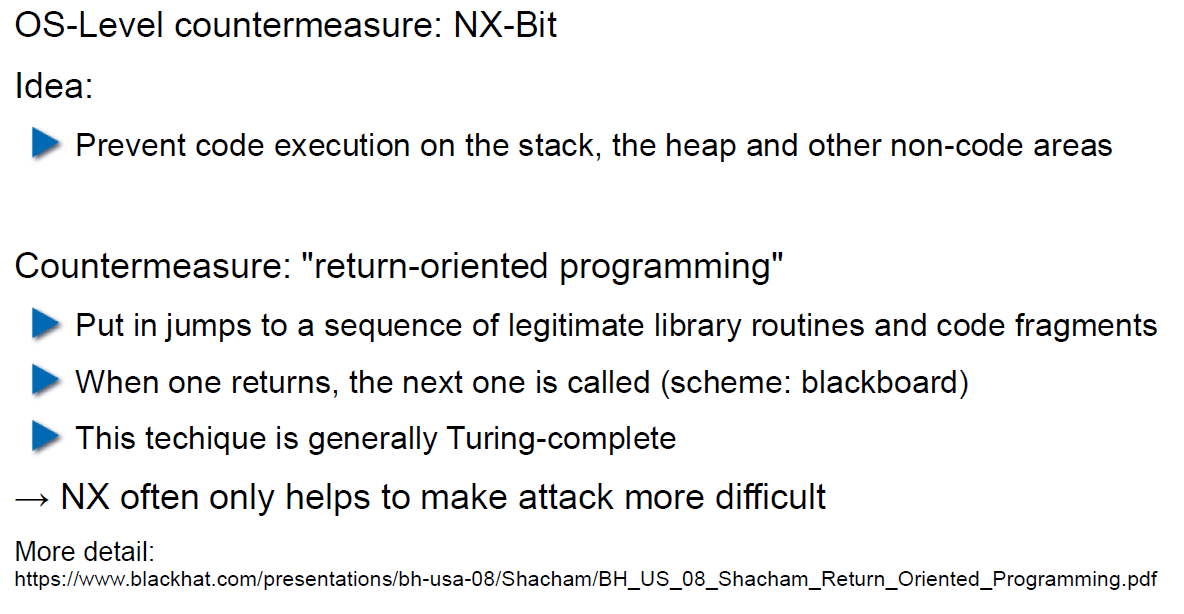
## Buffer Overflow II



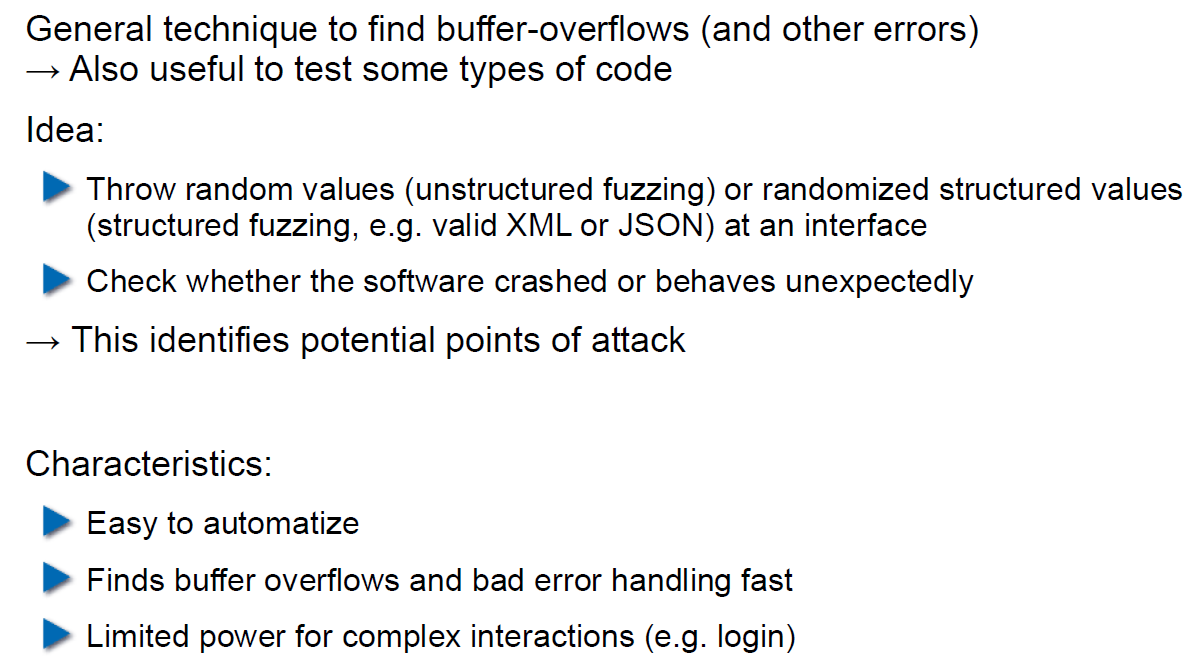
## Buffer Overflow II



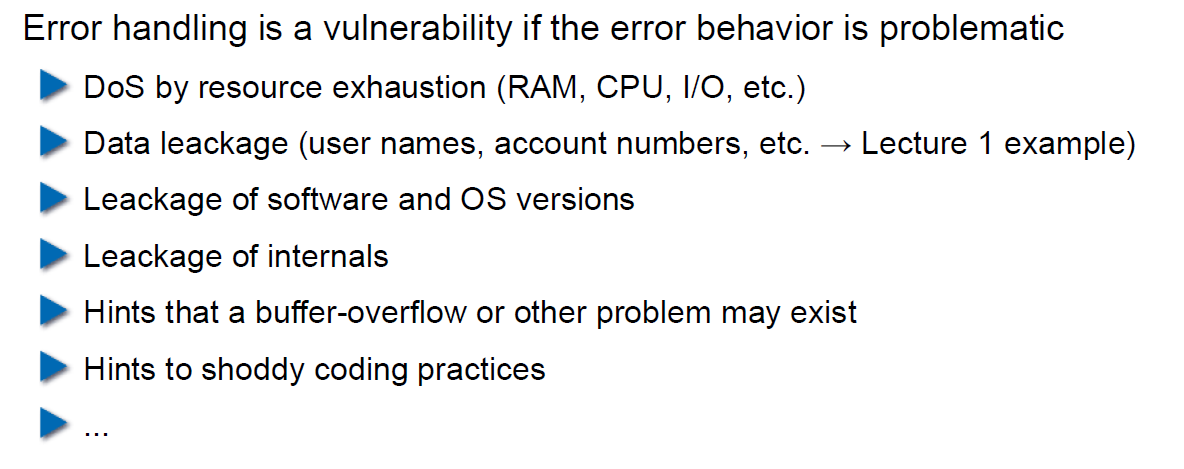
## Buffer Overflow II



## Attack Technique: Fuzzing



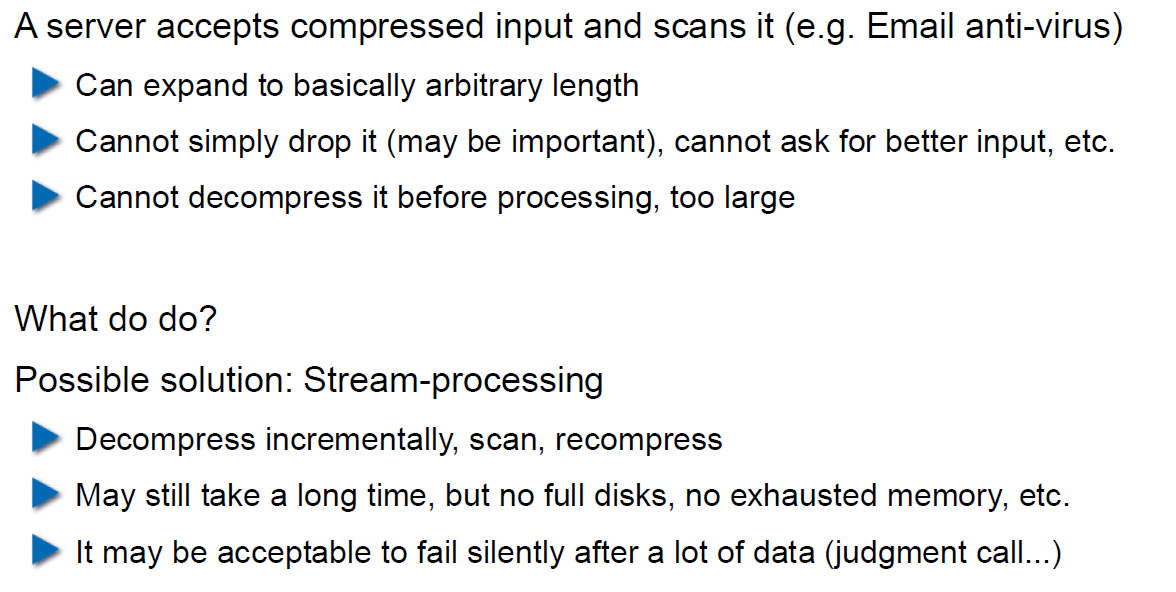
## Error Handling



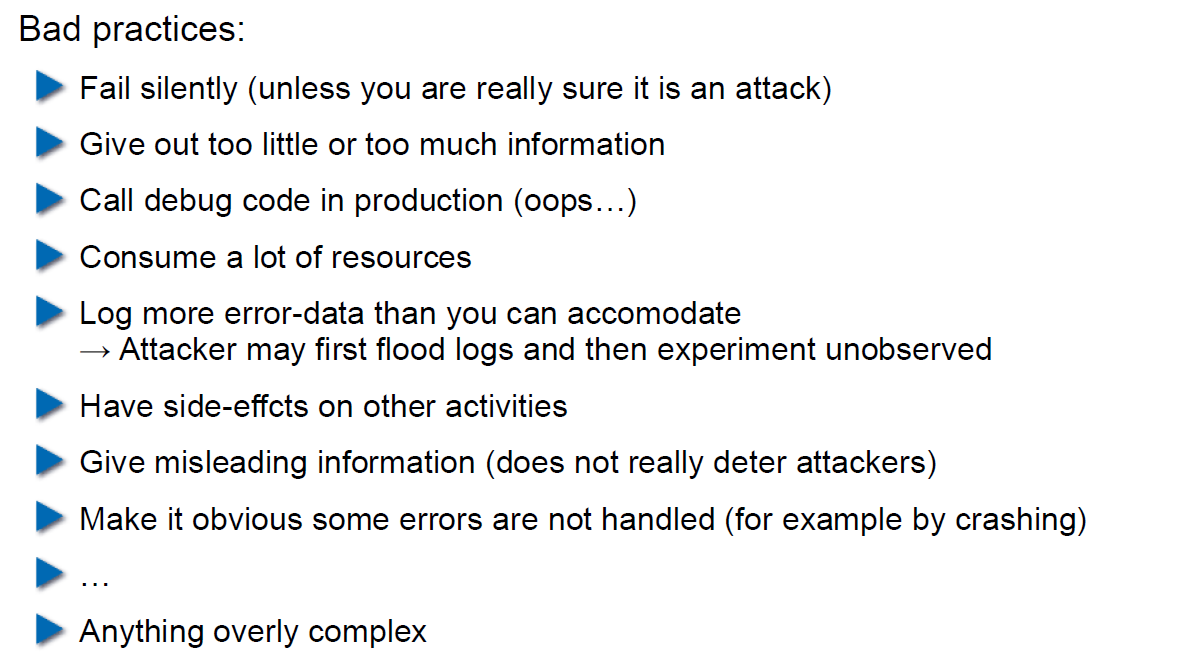
## Case study – Handling too long input



## Case-study: Compression Bomb

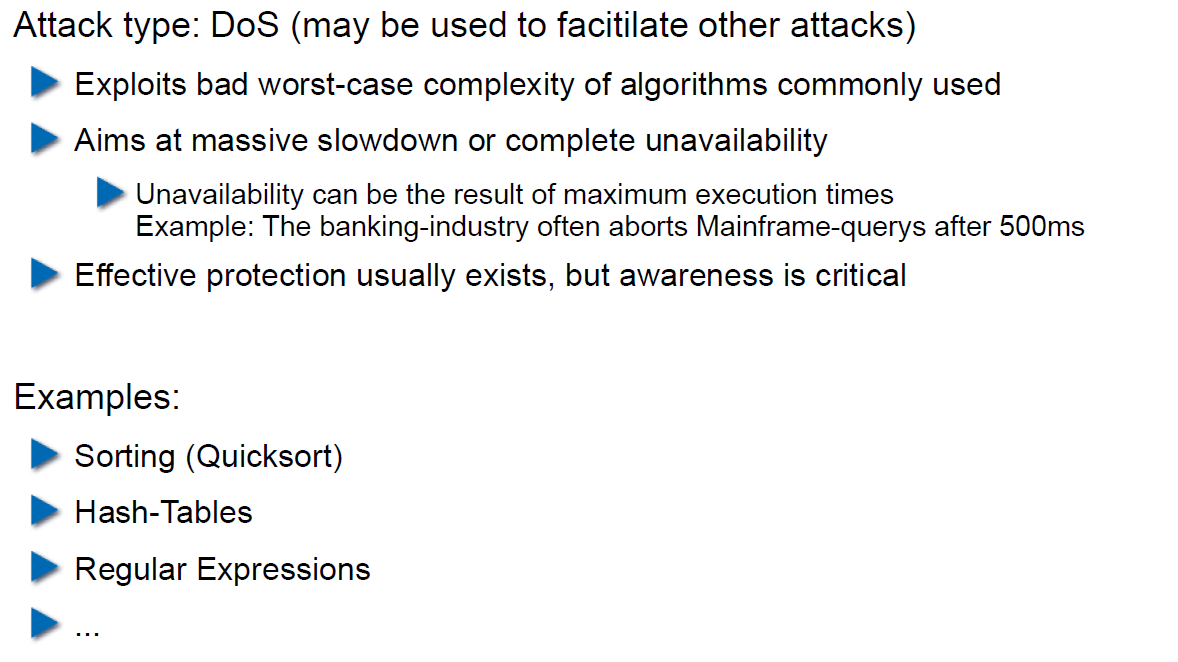


## Error Handling

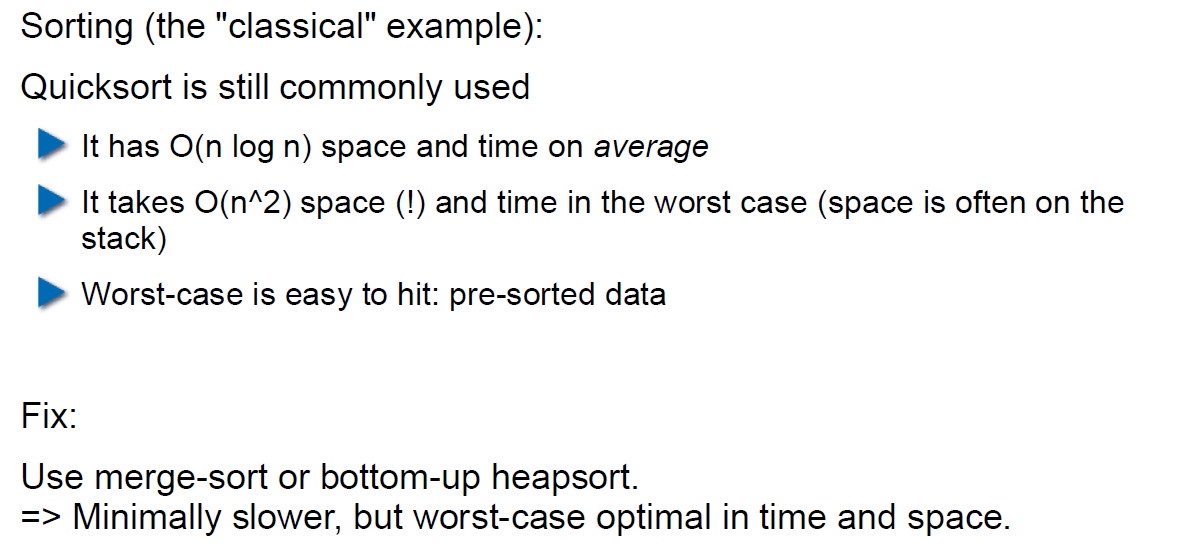




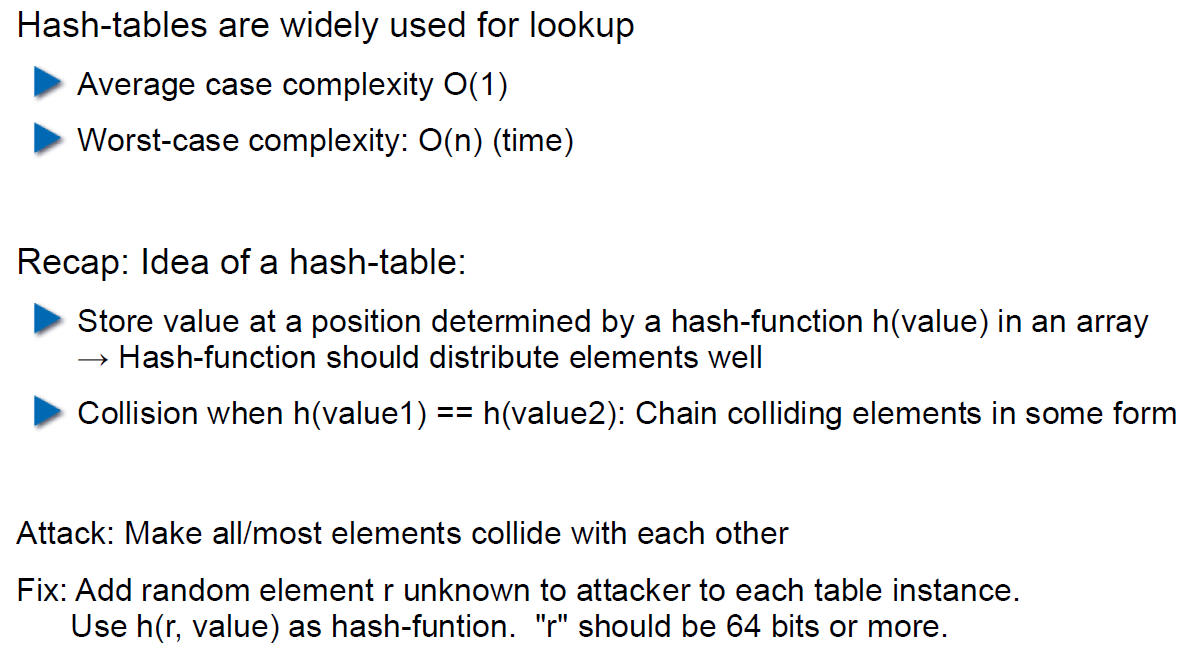
## Algorithmic Complexity Attacks



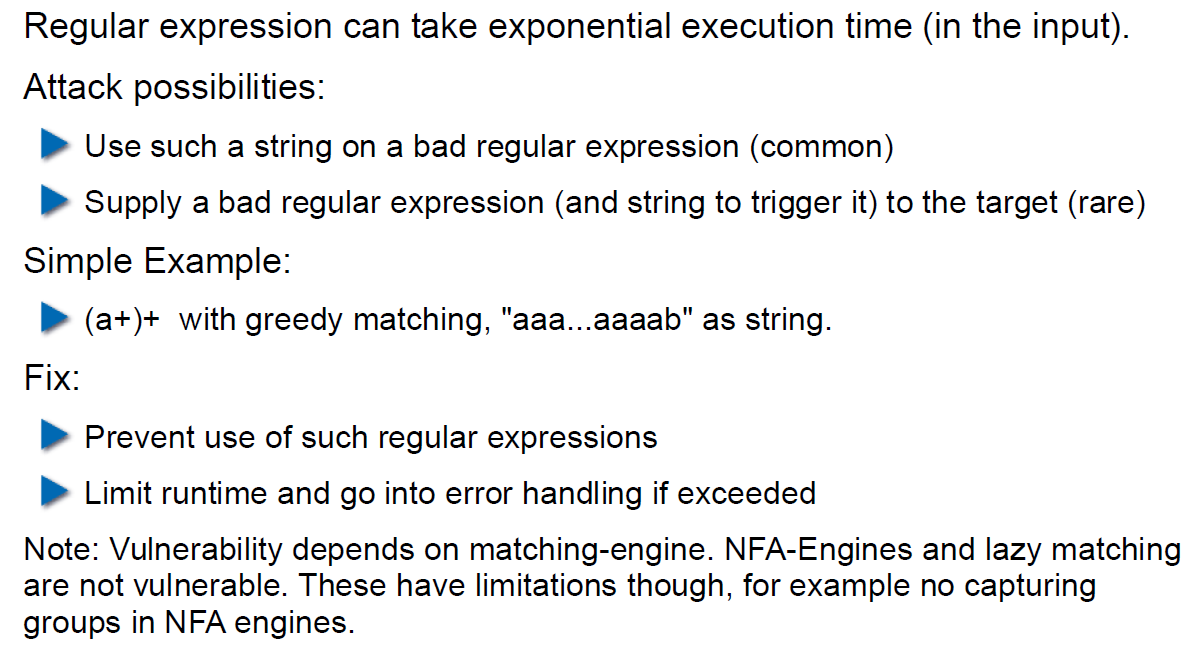
## Algorithmic Complexity: Sorting



## Algorithmic Complexity: Hashing

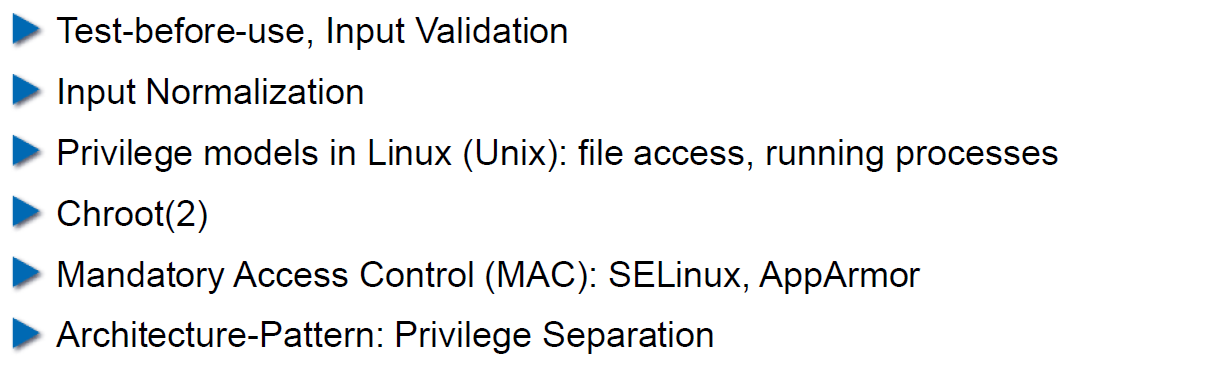


## Algorithmic Complexity: Regular Expressions

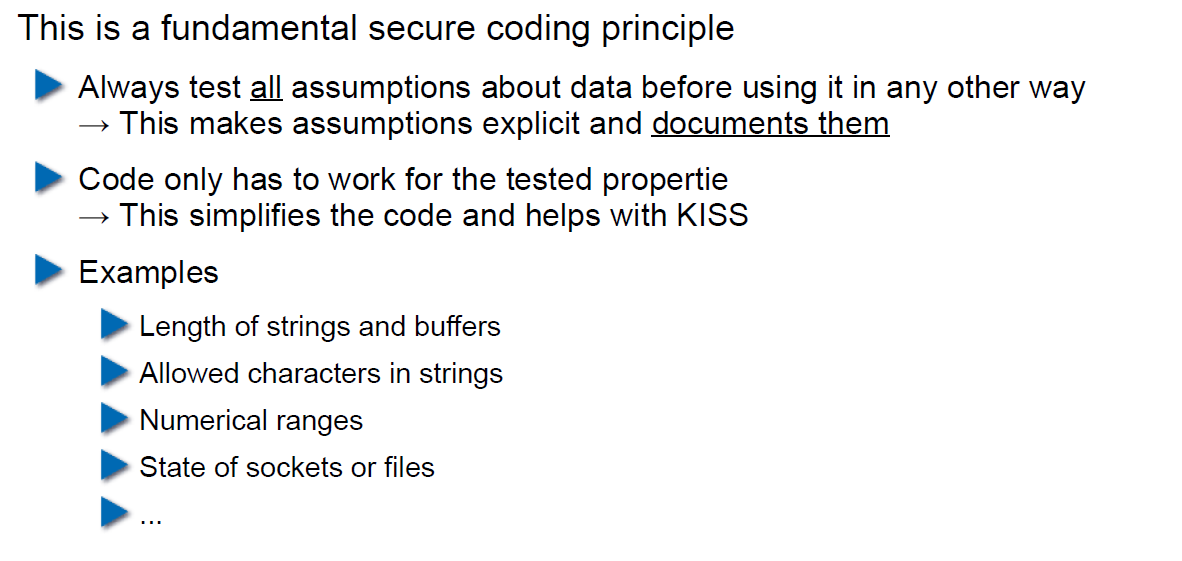


# Woche 3

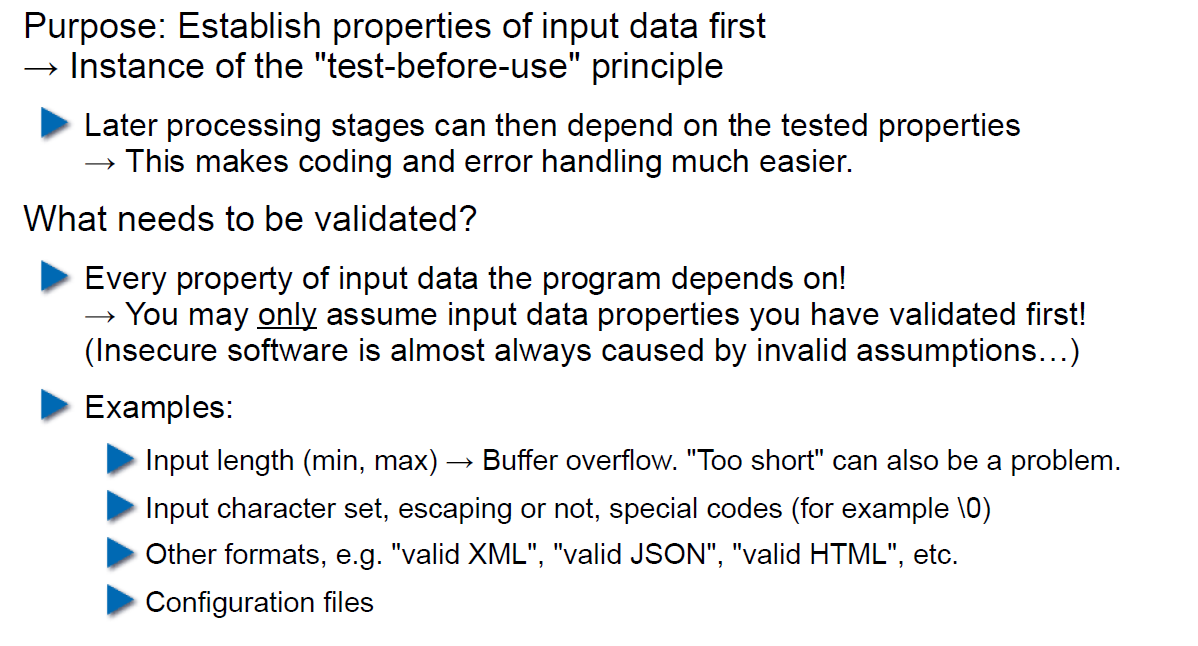
## Agenda



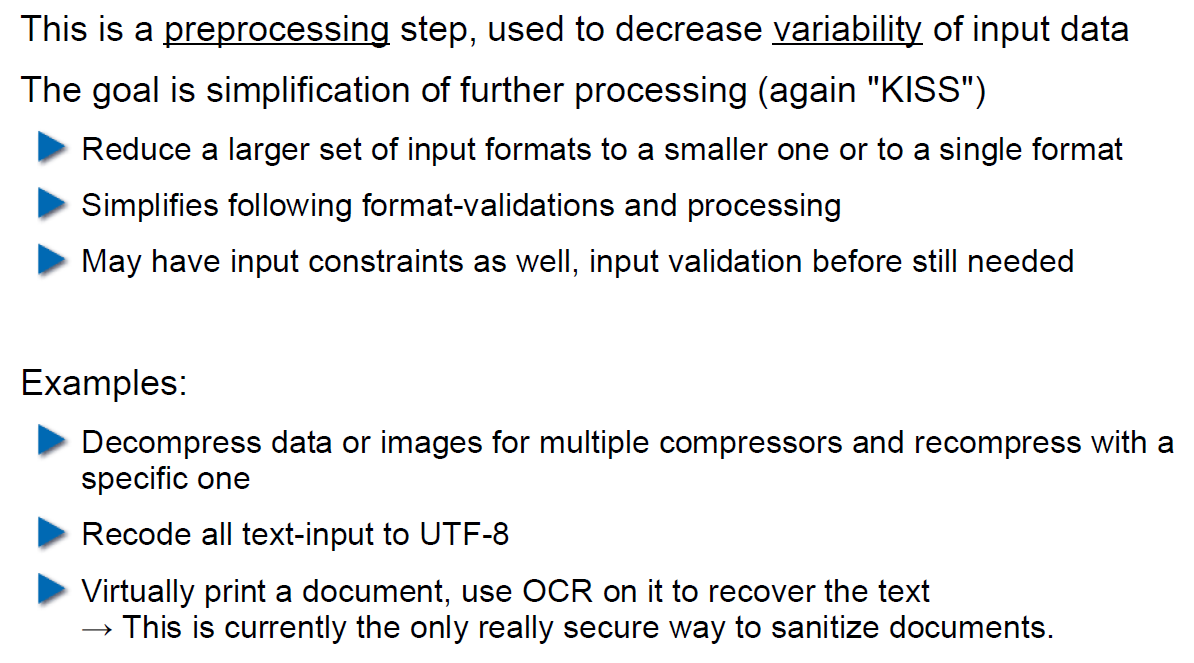
## Secure Coding Principle: Test-before-use



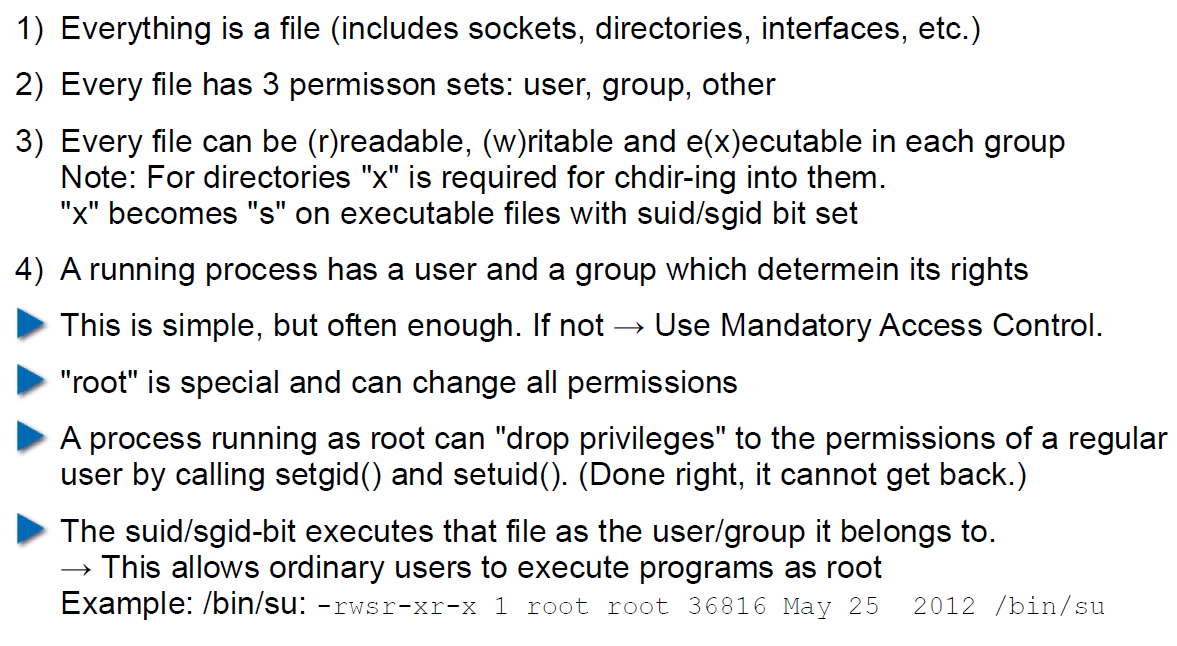
## Input Validation



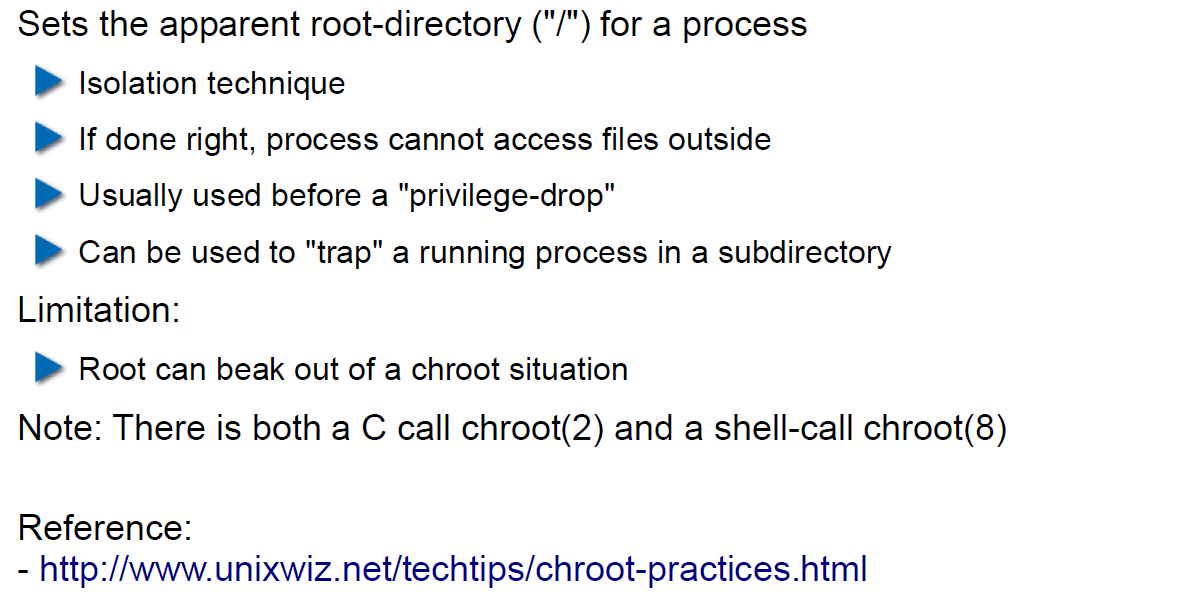
## Input Normalization



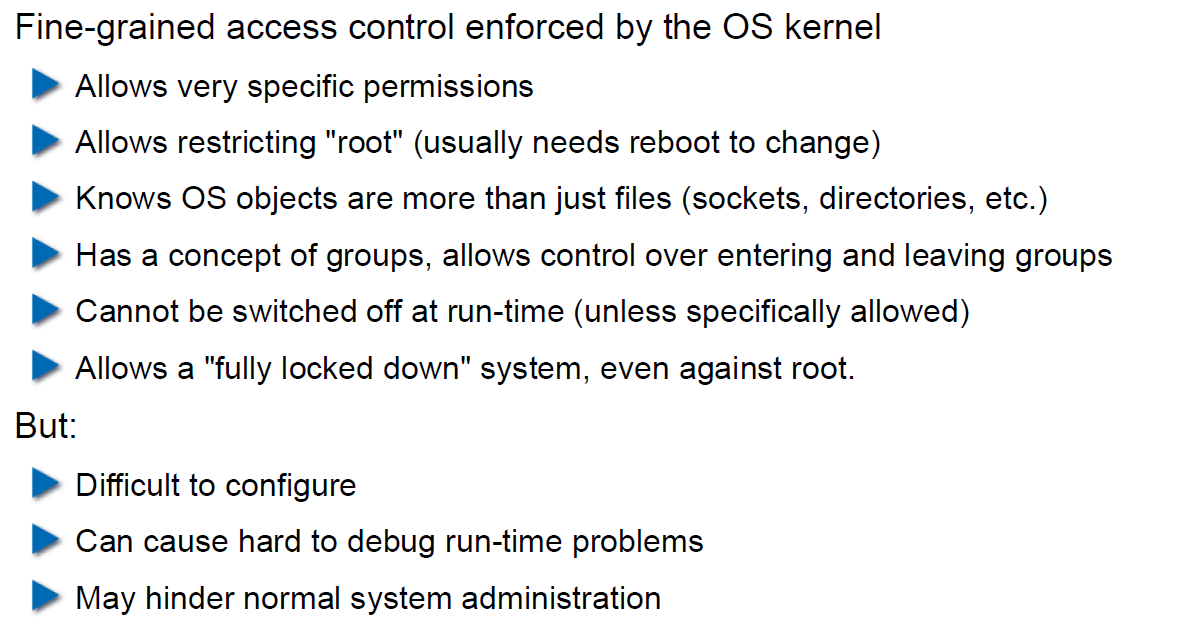
## Unix Permission Model (simplified)



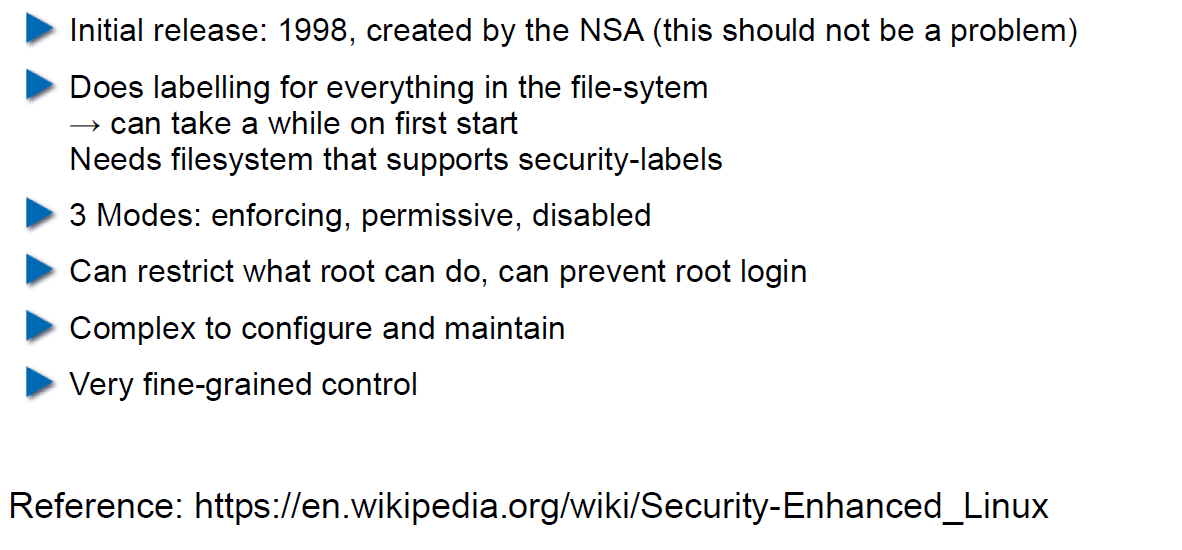
## Chroot



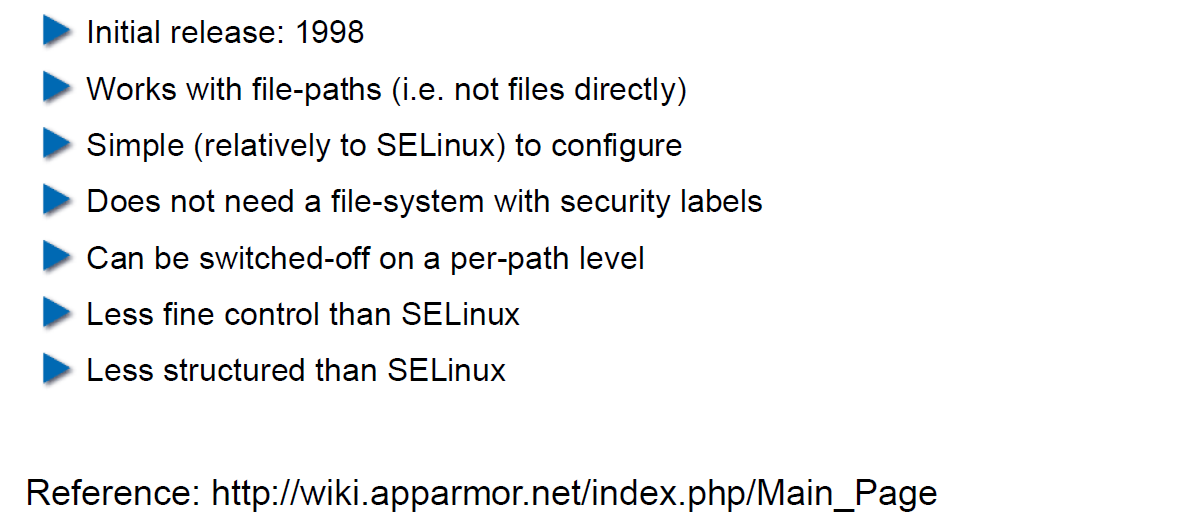
## Mandatory Access Control (MAC)



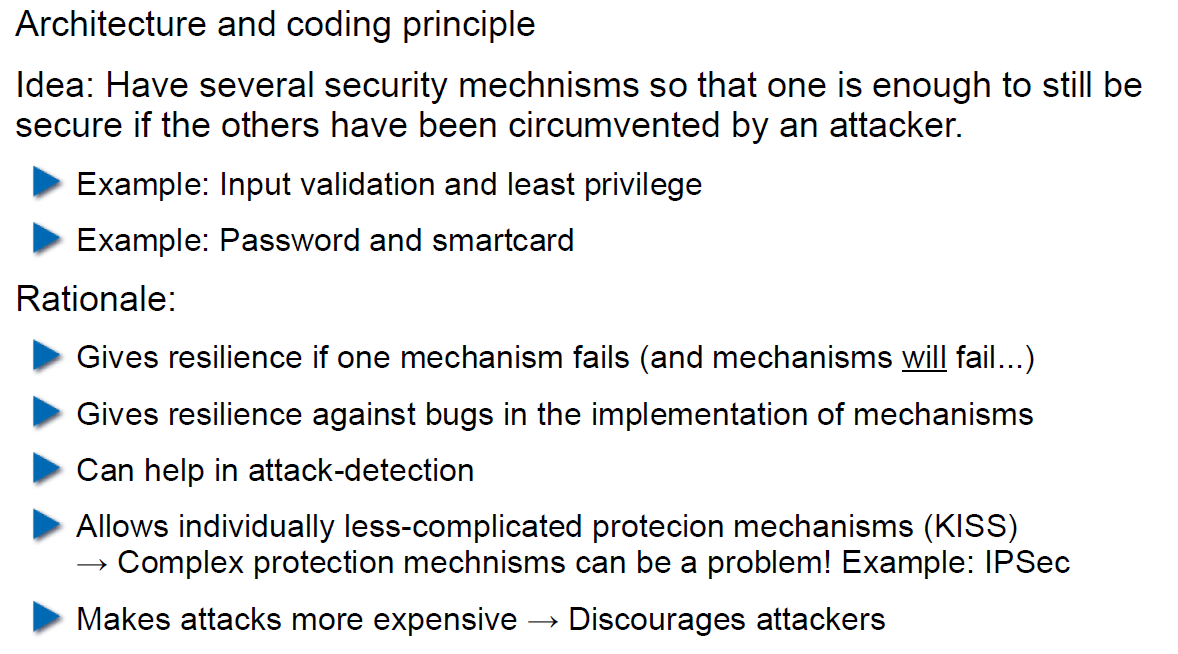
## MAC Example: SELinux



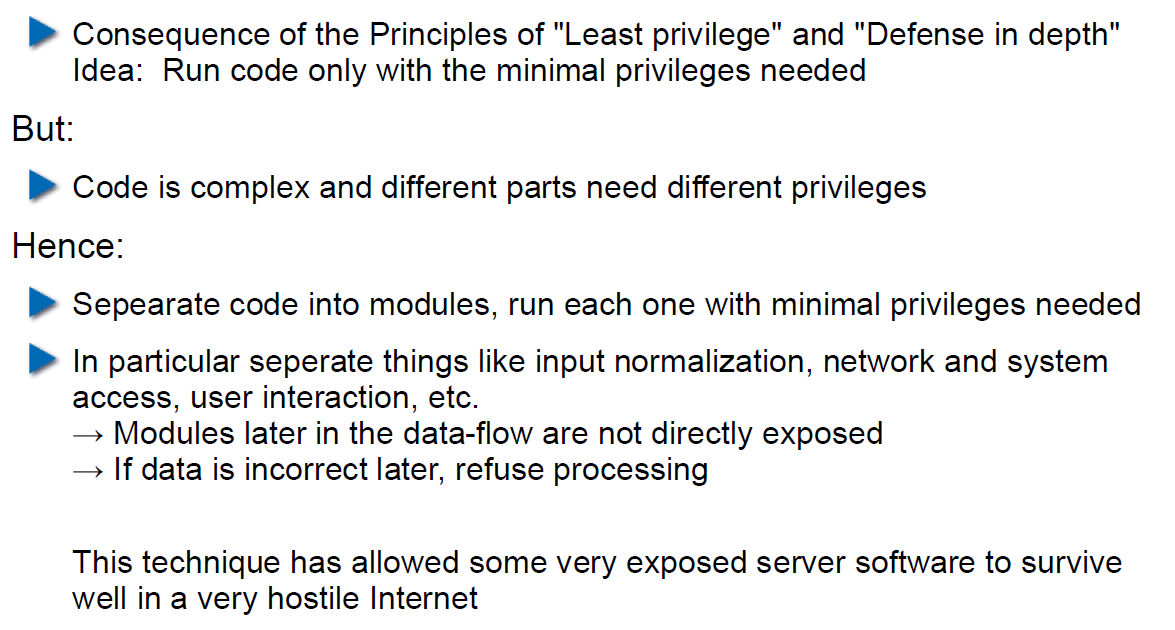
## MAC Example: AppArmor



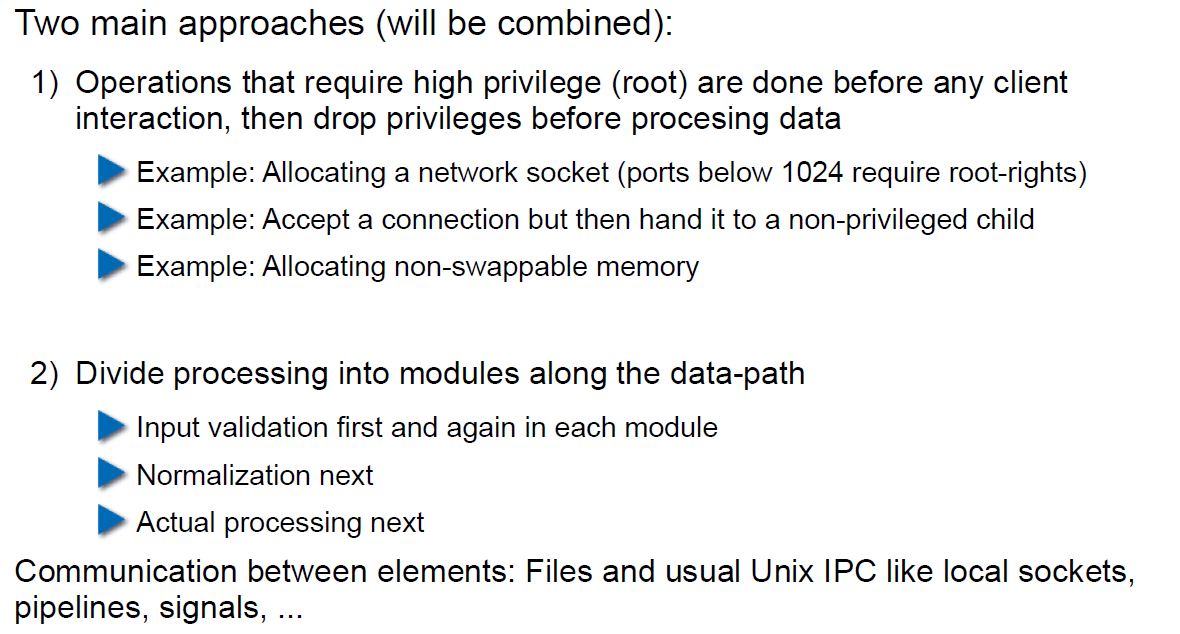
## Secure Coding Principle: Defense in Depth



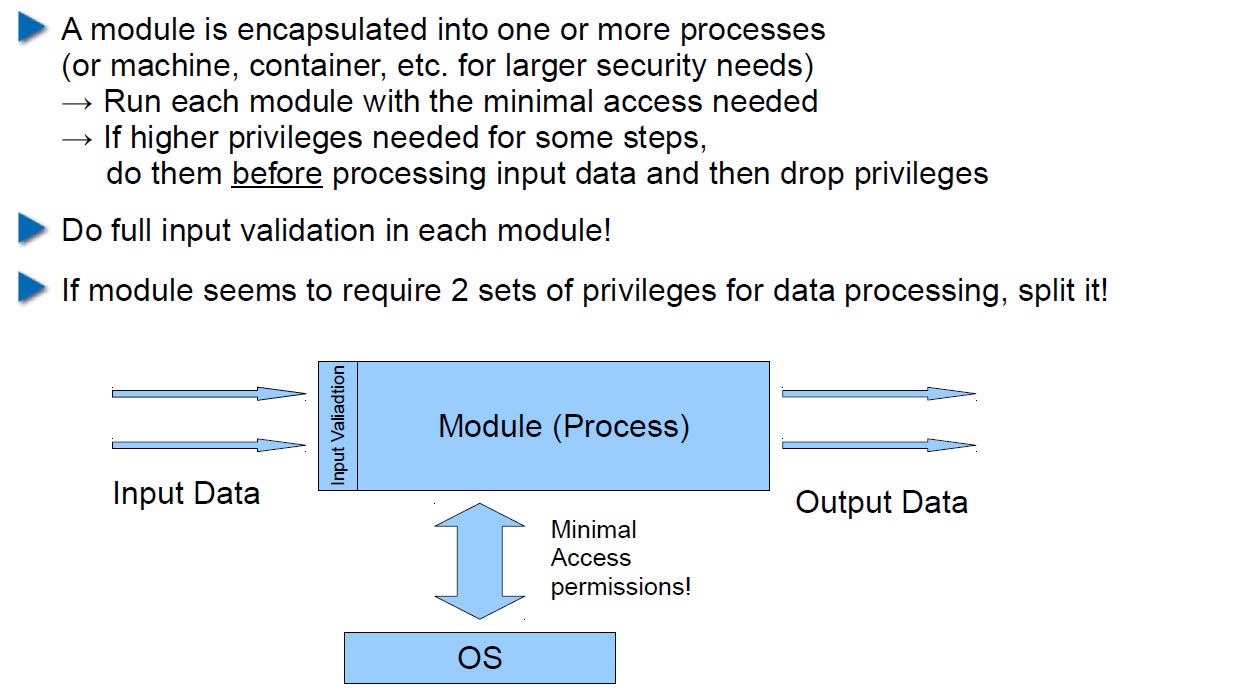
## Privilege Separation



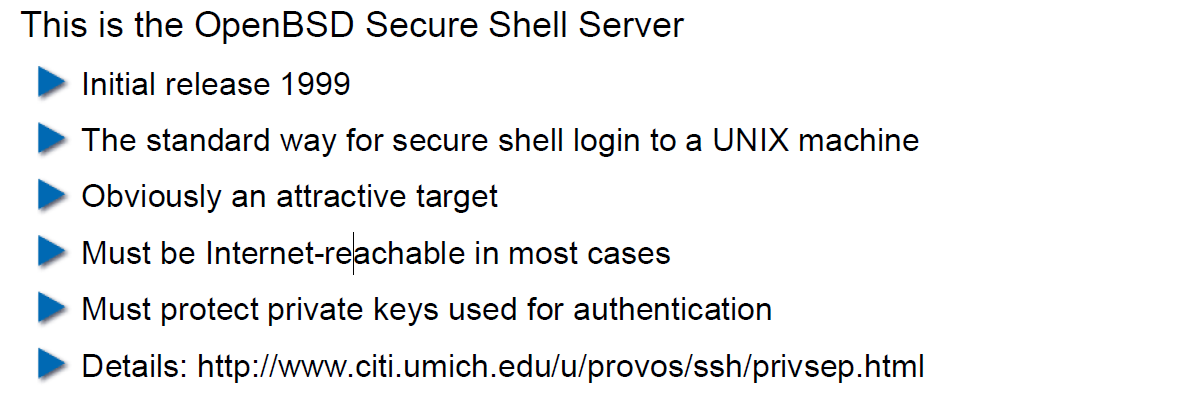
## Overall Idea



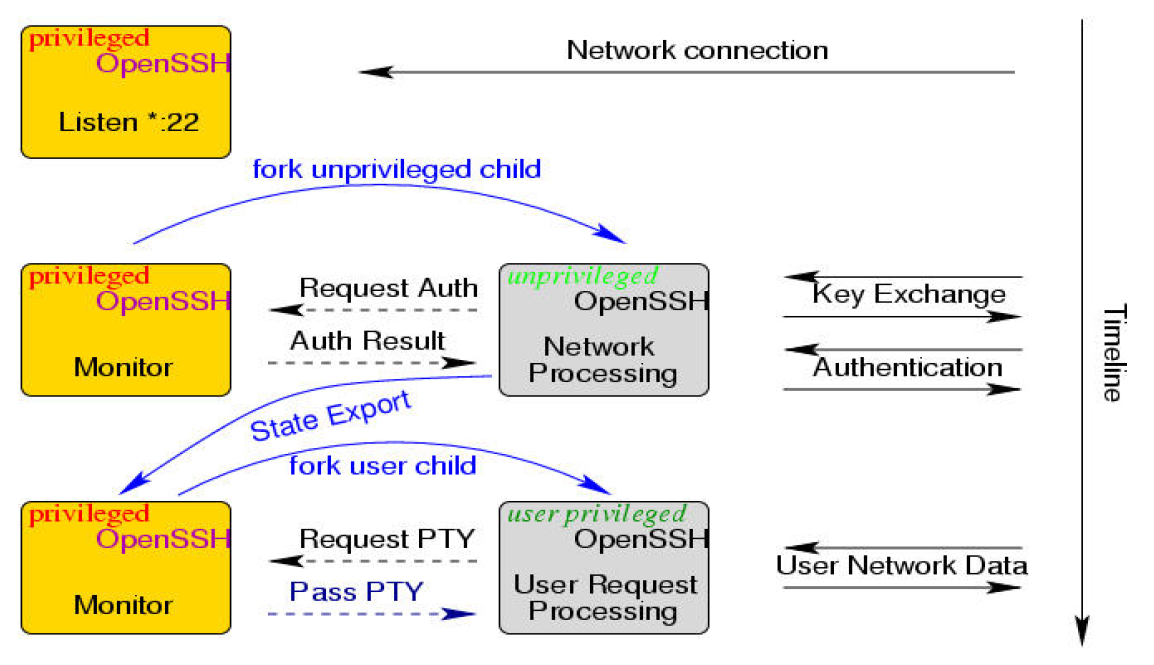
## Form of the Modules



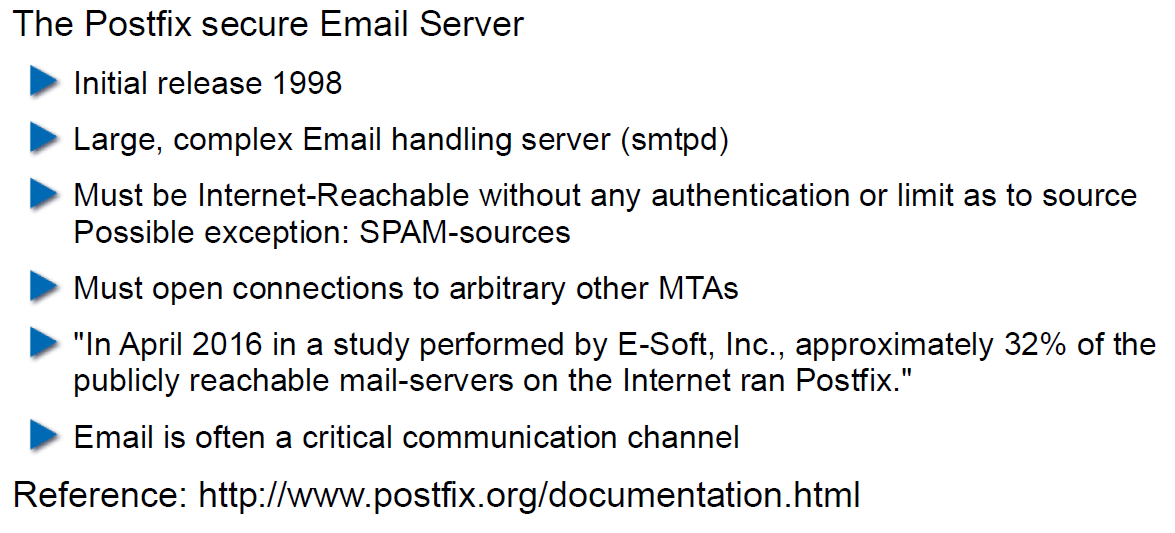
## Example: OpenSSH



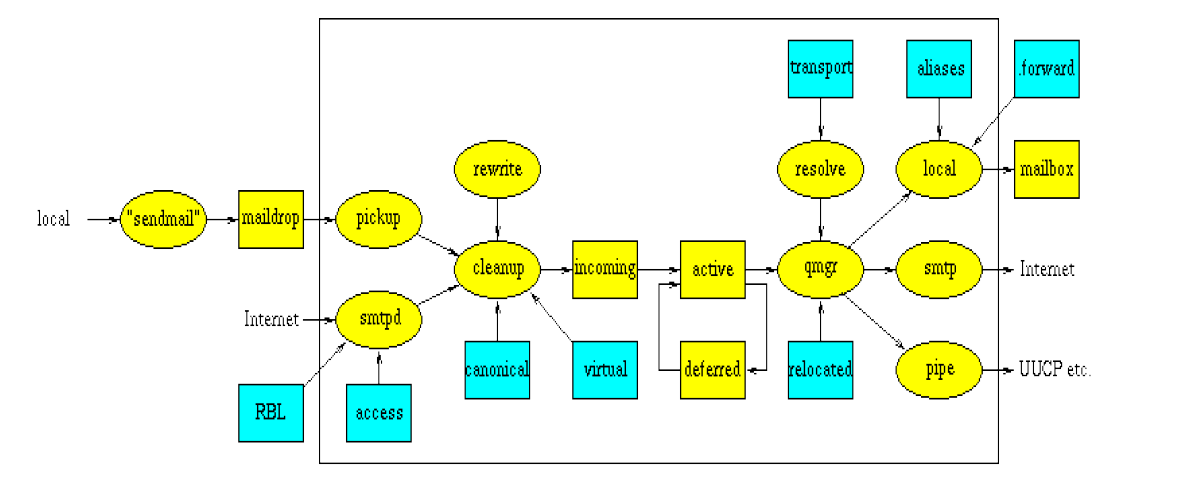
## OpenSSH: Privilege Separation Approach



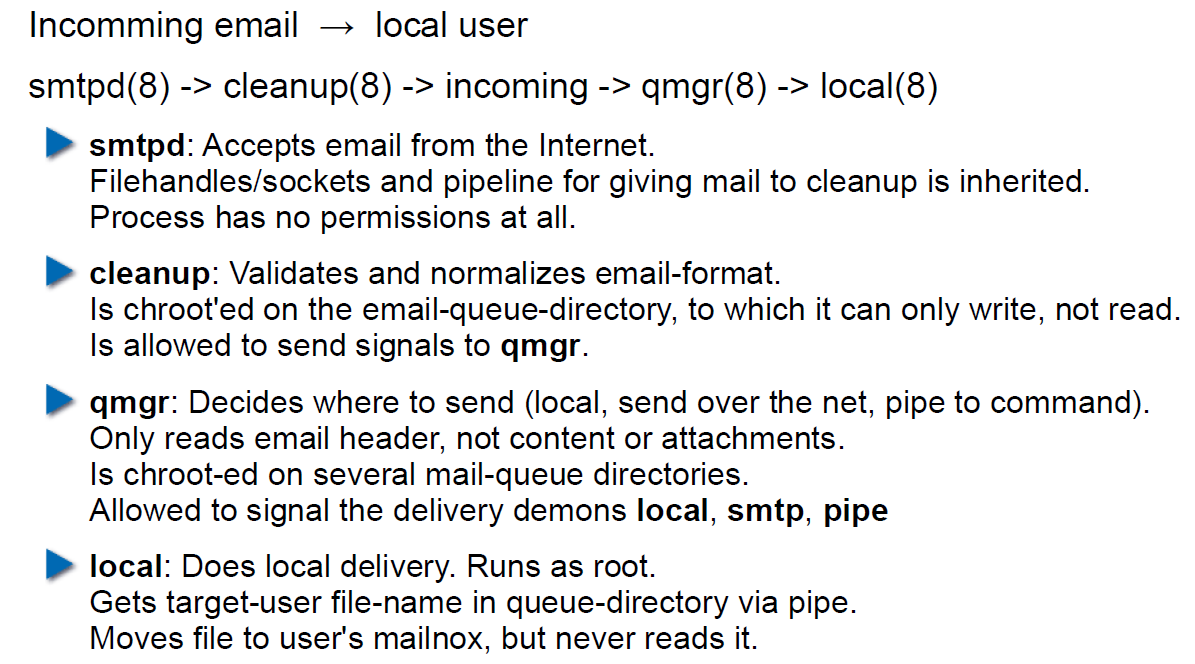
## Example: Postfix MTA (Email Server)



## Postfix System View



## Postfix: Example Data-Flow



# Woche 4