# COMx501: Computer Security and Forensics

**Achim D. Brucker**
a.brucker@sheffield.ac.uk https://www.brucker.ch/

**Software Assurance & Security Research**
Department of Computer Science, The University of Sheffield, Sheffield, UK
https://logicalhacking.com/

March 13, 2018

# COMx501: Computer Security and Forensics

## Part 09: An Introduction to Application Security

**Achim D. Brucker**

**a.brucker@sheffield.ac.uk** **https://www.brucker.ch/**

**Software Assurance & Security Research**
Department of Computer Science, The University of Sheffield, Sheffield, UK
https://logicalhacking.com/

March 13, 2017


Before we start, a few updates:

- Mole Quiz
  - **Start:** Tuesday (March 19th, 2018), 15.00 GMT
  - **Deadline:** Friday (March 23rd, 2018), 15.00 GMT
  - **Duration:** 22.5min (hard time bound: 25min)
  - **Questions:** 15 multiple choice questions
    - one correct answer (out of four possibilities)
    - you might need a calculator
    - question are assigned randomly
- If you have extenuating circumstances that guarantee extra time in assessments, please inform me by email until Monday, March 18th 9.00 am (GMT).
- Update on the Lab:
  - Thanks to all that attended yesterday
  - If you like, you can complete the exercise on your own (feel free to ask questions!)
  - Solutions will be published next week

# Outline

# Vault 7: CIA Hacking Tools Revealed

March 7, 2017: **https://wikileaks.org/ciav7p1/**


**Summary:** It is a leak about the CIA's hacking arsenal used against foreign governments and citizens both domestically and abroad

**Origin** : Allegedly from the CIA's Center for Cyber Intelligence unit in Langley, Virginia USA

**Volume:** 7,818 web pages with 943 attachments

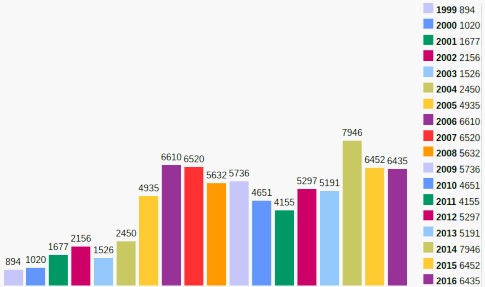
**Period:** Documents are from 2013-2016

- First analysis:
  - No evidence that crypto is broken!
  - Very effective techniques for **exploiting software bugs!**
    - Use of known bugs as well as unknown bugs (zero days)
- First conclusion:
  **We need to improve the quality (security, correctness, …) of the software we develop!**

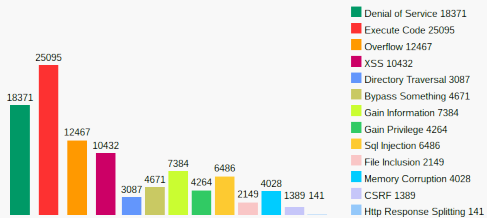
# Not a New Problem: Vulnerability Distribution (Since 1999)

www.cvedetails.com

## Vulnerability by Year



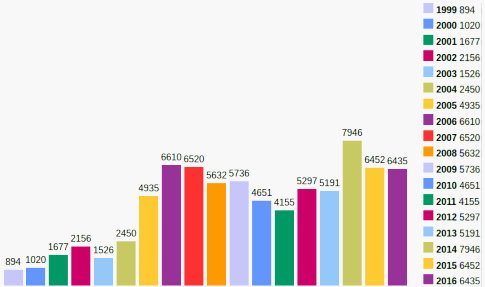

## Vulnerability Types



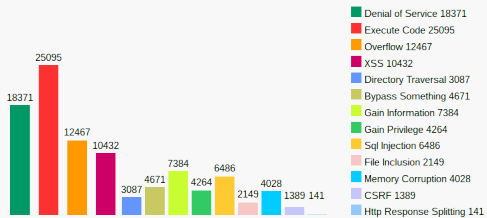
# Not a New Problem: Vulnerability Distribution (Since 1999)

www.cvedetails.com

## Vulnerability by Year




## Vulnerability Types

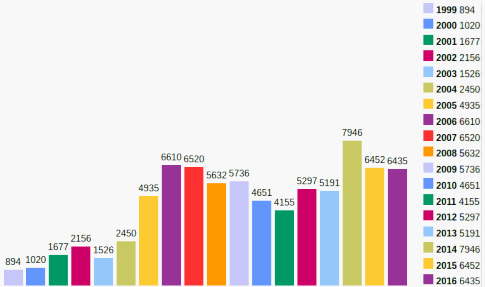



Still, attacks on crypto not a common problem

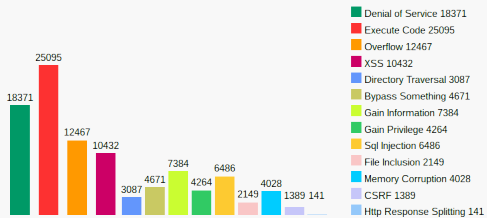
# Not a New Problem: Vulnerability Distribution (Since 1999)

www.cvedetails.com

## Vulnerability by Year



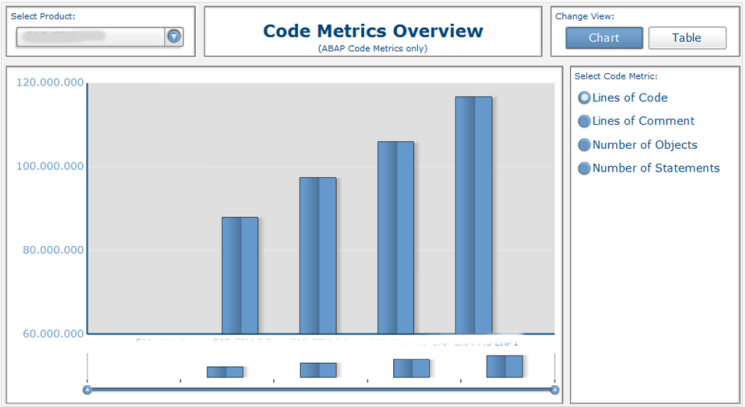

## Vulnerability Types




Still, attacks on crypto not a common problem – **but a lot of "rotten" software …**

# Why is is so Hard to Get Software "Right"?

## Evolution of Source Code



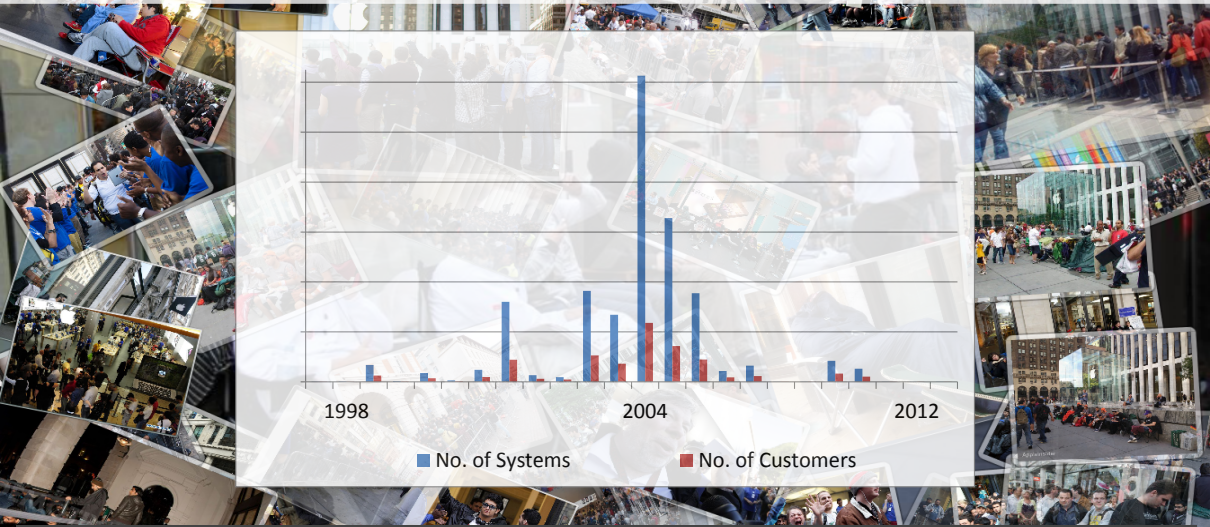

- Increase in
  - code size
  - code complexity
  - number of products
  - product versions
  - used technologies (prog. languages, frameworks)

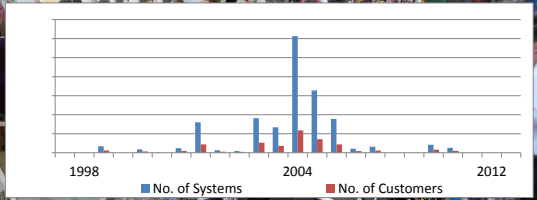
# Why is is so Hard to Get Software "Right"?

## The Software Maintenance Challenge

# Why is is so Hard to Get Software "Right"?

## The Software Maintenance Challenge






## Example

| Product | Release | EOL | ext. EOL |
|---|---|---|---|
| Windows XP | 2001 | 2009 | 2014 |
| Windows 8 | 2012 | 2018 | 2023 |
| Red Hat Ent. Linux | 2012 | 2020 | 2023 |

# Outline

# CWE: Common Weakness Enumeration

https://cwe.mitre.org/index.html


- Catalog (ontology) of software weaknesses and vulnerabilities
  A language for describing software vulnerabilities
- Very fine grained (and not necessarily disjoint)
- Different "entry points" for browsing, e.g.,
  - https://cwe.mitre.org/index.html
  - https://cwe.mitre.org/data/index.html
- Used in the Common Vulnerabilities and Exposures (CVE) entries
- Most common vulnerabilities:
  - CWE-25 (https://cwe.mitre.org/data/definitions/25.html)
  - OWASP Top Ten
    (https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project)

# CVE: Common Vulnerabilities and Exposures

**https://cve.mitre.org/**

CVE is a database of software vulnerabilities

- Each entry has a unique id
- Entries usually contain
  - textual description of the vulnerability
  - description of the affected software and version
  - type of vulnerability (CWE)
  - patch/fix instructions (if available)
  - availability of an exploit
    (a proof of concept that shows how to make use of a vulnerability)
  - standardized risk assessment: Common Vulnerability Scoring System (CVSS)
- No obligation to register CVEs
  - vendors may apply for a CVE id
  - security researchers may apply for a CVE id
  - most FLOSS projects register CVEs

Vendors (or security research firms) may have their system,e g.,

- Microsoft Security Bulletin
  https://technet.microsoft.com/en-us/library/security/ms17-mar

# CVSS: Common Vulnerability Scoring System (1/6)

Industry standard for rating the severity of software vulnerabilities

- Version 3 is the most current one, still version 2 most widely used
- We will focus on the the CVSS base score of version 2:
  - Measures three areas of concern:
    1. **Base Metrics** for qualities intrinsic to a vulnerability
    2. Temporal Metrics for characteristics that evolve over the lifetime of vulnerability
    3. Environmental Metrics for vulnerabilities that depend on a particular implementation or environment

    A numerical score is generated for each of these metric groups
    (often published as vector containing all three values)

**Access Vector (AV):** shows how a vulnerability may be exploited

| Value | Description | Score |
|---|---|---|
| Local (L) | The attacker must have physical access or a local account | 0.395 |
| Adjacent Network (A) | The attacker must have access to a neighboring network | 0.646 |
| Network (N) | The attacker only needs remote access | 1.000 |

**Access Complexity (AC):** how easy/difficult it is to exploit the discovered vulnerability

| Value | Description | Score |
|---|---|---|
| High (H) | Specialized conditions must be fulfilled (e.g., race conditions) | 0.350 |
| Medium (M) | Some additional requirements must be fulfilled (e.g., non-default configuration) | 0.610 |
| Low (L) | No special conditions | 0.710 |

**Authentication (AU):** how often needs an attacker to authenticate to a target to exploit it

| Value | Description | Score |
|---|---|---|
| Multiple (M) | Several authentications required | 0.450 |
| Single (S) | Single authentication required | 0.560 |
| None (N) | Unauthenticated access | 0.704 |

**Confidentiality (C):** impact on the confidentiality of the processed data

| Value | Description | Score |
|---|---|---|
| None (N) | No impact on the confidentiality | 0.000 |
| Partial (P) | Considerable information disclosure (but constrained) | 0.275 |
| Complete (C) | Total information disclosure (all data/information) | 0.660 |

**Integrity (I):** impact on the integrity of the system

| Value | Description | Score |
|---|---|---|
| None (N) | No impact on the integrity | 0.000 |
| Partial (P) | Modification of some data (but limited) | 0.275 |
| Complete (C) | Total loss of integrity | 0.660 |

**Availability (A):** impact on the availability of the system

| Value | Description | Score |
|---|---|---|
| None (N) | No impact on the availability | 0.000 |
| Partial (P) | Reduced performance or some loss of functionality) | 0.275 |
| Complete (C) | Total loss of availability | 0.660 |

- These six metrics are used to calculate the BaseScore:

$$Exploitability = 20 \times AccessVector \times AccessComplexity \times Authentication$$

$$Impact = 10.41 \times (1 - (1 - ConfImpact) \times (1 - IntegImpact) \times (1 - AvailImpact))$$

$$f(Impact) = \begin{cases} 0.000, & \text{if } Impact = 0 \\ 1.176, & \text{otherwise} \end{cases}$$

$$BaseScore = ((0.6 \times Impact) + (0.4 \times Exploitability) - 1.5) \times f(Impact)$$

  The *BaseScore* is rounded to one decimal.
- CVSS 2.0 calculator: https://nvd.nist.gov/CVSS/v2-calculator

## Example

A buffer overflow vulnerability affects web server software that allows a remote user to gain partial control of the system, including the ability to cause it to shut down

## Example

A buffer overflow vulnerability affects web server software that allows a remote user to gain partial control of the system, including the ability to cause it to shut down

| Metric | Value | Description |
|---|---|---|
| Access Vector | Network | Access from any network (Internet) possible |
| Access Complexity | Low | No special requirements for access |
| Authentication | None | No authentication necessary |
| Confidentiality | Partial | Only partial control (e.g., reading files) |
| Integrity | Partial | Only partial access (e.g., modifying files) |
| Availability | Complete | Attacker can cause a shutdown |

## Example

A buffer overflow vulnerability affects web server software that allows a remote user to gain partial control of the system, including the ability to cause it to shut down

| Metric | Value | Description |
|---|---|---|
| Access Vector | Network | Access from any network (Internet) possible |
| Access Complexity | Low | No special requirements for access |
| Authentication | None | No authentication necessary |
| Confidentiality | Partial | Only partial control (e.g., reading files) |
| Integrity | Partial | Only partial access (e.g., modifying files) |
| Availability | Complete | Attacker can cause a shutdown |

- Exploitability: 10 and Impact: 8.5
- **Base Score:** 9.0
- CVSS Vector: (AV:N/AC:L/Au:N/C:P/I:P/A:C)

- While lower CVSS scores represent lower risk, they are not the "absolute truth"!
- Consider Heartbleed (CVE-2014-0160):
  - One of the most well-known vulnerabilities of the last years
  - Full disclosure of server key possible
  - CVSS: 5.0/10
  - Most likely the only CVE with a CVSS warning/explanation:

> CVSS V2 scoring evaluates the impact of the vulnerability on the host where the vulnerability is located. When evaluating the impact of this vulnerability to your organization, take into account the nature of the data that is being protected and act according to your organization's risk acceptance. While CVE-2014-0160 **does not allow unrestricted access to memory** on the targeted host, a **successful exploit does leak** information from memory locations which have the potential to contain particularly sensitive information, e.g., **cryptographic keys and passwords**. Theft of this information could enable other attacks on the information system, the impact of which would depend on the sensitivity of the data and functions of that system.

# Outline

# A Path Towards (More) Secure Software

## SAP's Secure Software Development Lifecycle ($S^2DL$)


## Training

- Security awareness
- Secure programming
- Threat modelling
- Security testing
- Data protection and privacy
- Security expert curriculum ("Masters")


## Risk Identification

- Risk identification ("high-level threat modelling")
- Threat modelling
- Data privacy impact assessment


## Plan Security Measures

- Plan product standard compliance
- Plan security features
- Plan security tests
- Plan security response


## Secure Development

- Secure Programming
- Static code analysis (SAST)
- Code review


## Security Testing

- Dynamic Testing (e.g., IAST, DAST)
- Manual testing
- External security assessment


## Security Validation ("First Customer")

- Check for "flaws" in the implementation of the $S^2DL$
- Ideally, security validation finds:
- No issues that can be fixed/detected earlier
- Only issues that cannot be detect earlier
  (e.g., insecure default configurations, missing security documentation)

Penetration tests in productive environments are different:

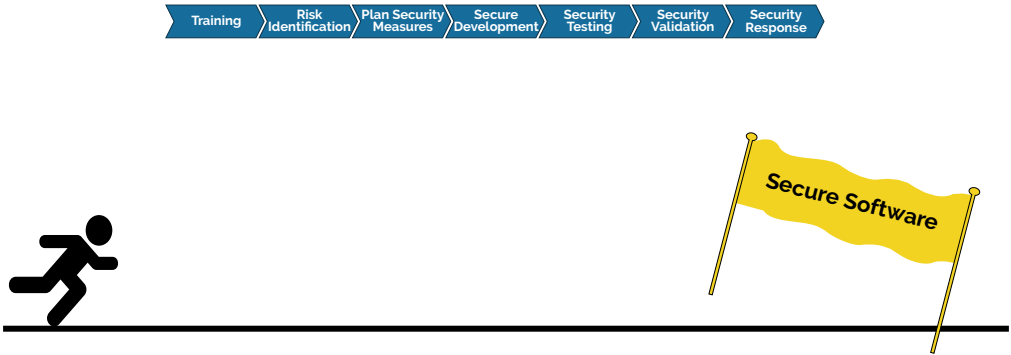
- They test the actual configuration
- They test the productive environment (e.g., cloud/hosting)


## Security Response

- Execute the security response plan
- Security related external communication
- Incident handling
- Security patches
- Monitoring of third party components

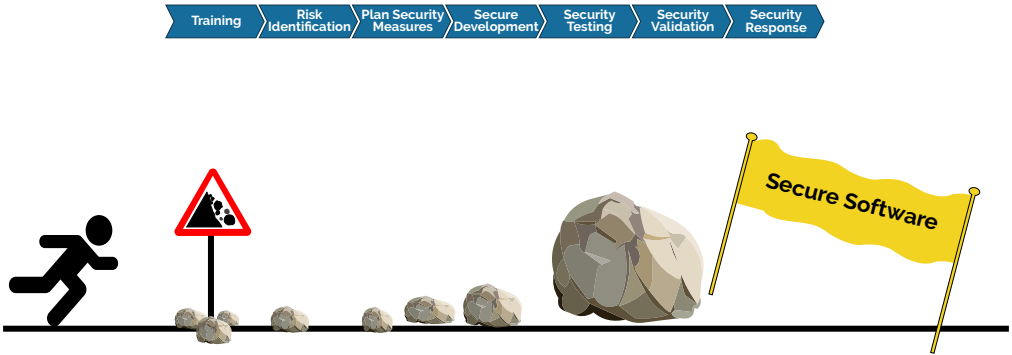
# A Path Towards (More) Secure Software

## SAP's Secure Software Development Lifecycle ($S^2DL$)



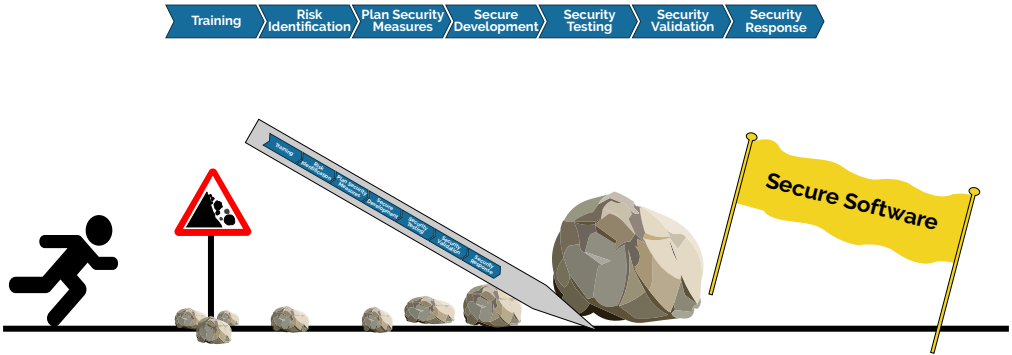
# A Path Towards (More) Secure Software

## SAP's Secure Software Development Lifecycle ($S^2DL$)



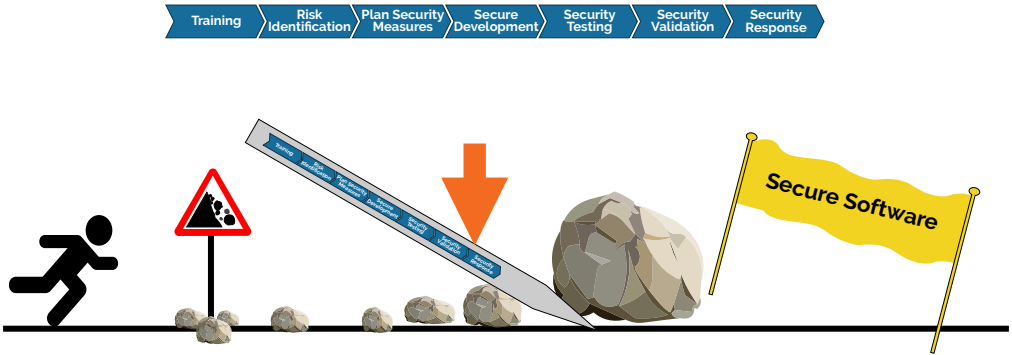
# A Path Towards (More) Secure Software

## SAP's Secure Software Development Lifecycle ($S^2DL$)



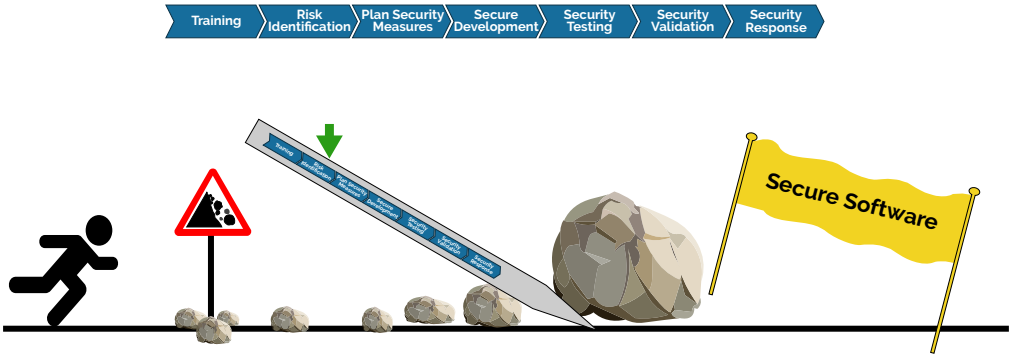
# A Path Towards (More) Secure Software

SAP's Secure Software Development Lifecycle ($S^2DL$)



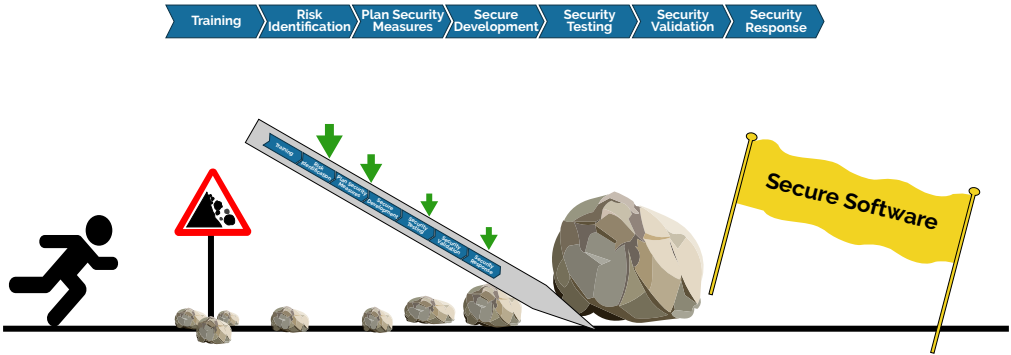
# A Path Towards (More) Secure Software

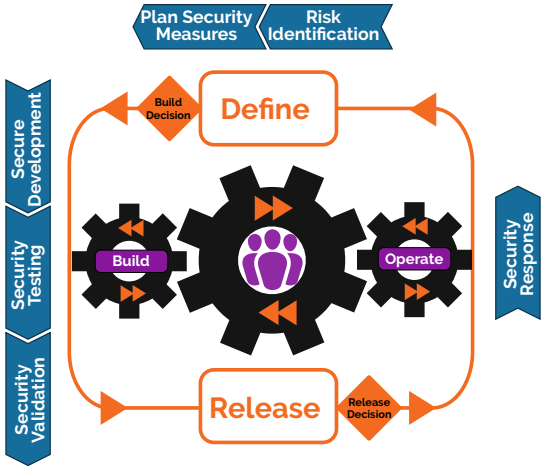
SAP's Secure Software Development Lifecycle ($S^2DL$)

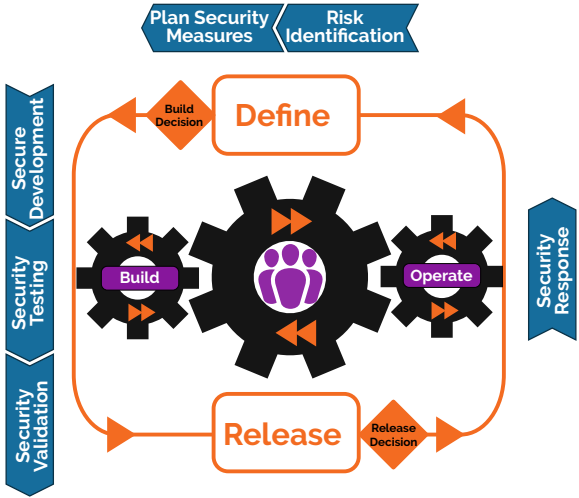


# A Path Towards (More) Secure Software

## SAP's Secure Software Development Lifecycle ($S^2DL$)




Plan Security Measures
Risk Identification
Secure Development
Security Testing
Security Validation
Build Decision
Define
Build
Operate
Security Response
Release
Release Decision


Plan Security Measures
Risk Identification
Secure Development
Security Testing
Security Validation
Build Decision
Define
Build
Operate
Security Response
Release
Release Decision

# Secure Software Lifecycle: My Vision

# Secure Software Lifecycle: My Vision

# Secure Software Lifecycle: My Vision

# Outline

# Conclusion

- Application/software security is important
- Attackers (and governments) exploit software vulnerabilities
  - recall the famous iPhone case
    ("FBI paid more than $1 Million to access iPHone)
  - attacks on actual crypto (i.e., the math) very seldom
- Application security is mostly a software engineering/programming problem!

What we will do during the next weeks:

- Have a closer look at several steps of the SDLC:
  - Threat modelling
  - Static analysis
  - Security testing
- Lab on application security

Thank you for your attention!
Any questions or remarks?

**Contact:**

Dr. Achim D. Brucker
Department of Computer Science
University of Sheffield
Regent Court
211 Portobello St.
Sheffield S1 4DP, UK

a.brucker@sheffield.ac.uk
@adbrucker
https://de.linkedin.com/in/adbrucker/
https://www.brucker.ch/
https://logicalhacking.com/blog/

# Bibliography I

Ross J. Anderson.

*Security Engineering: A Guide to Building Dependable Distributed Systems*.
John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 2001.
The complete book is available at: http://www.cl.cam.ac.uk/~rja14/book.html.

# Document Classification and License Information