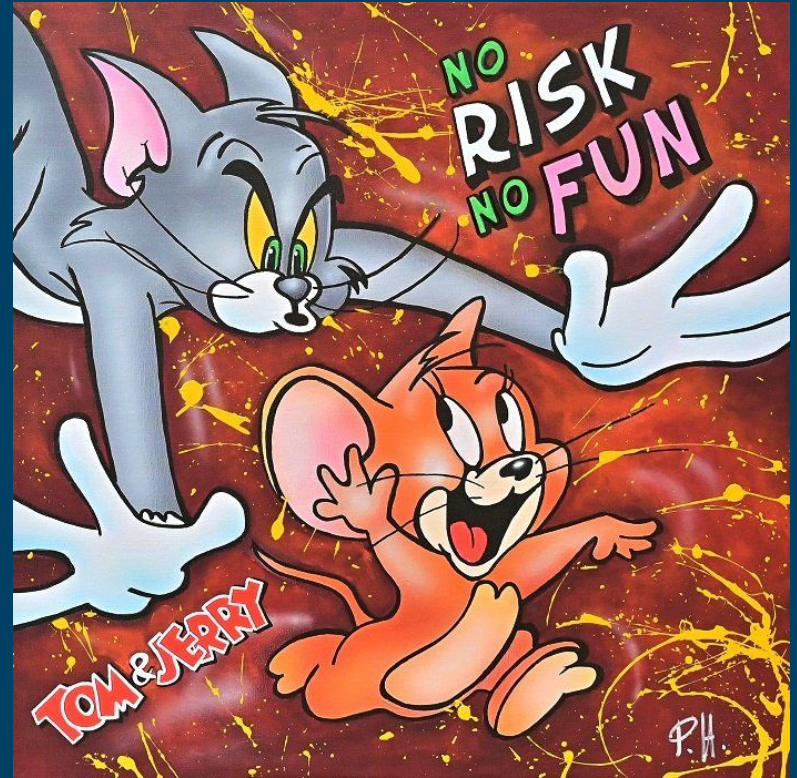


Sichere Software- entwicklung

No risk no fun?

jasie.de/secure-dev
[@jasie@machteburch.social](https://twitter.com/ja_sie)
twitter.com/ja_sie



Quelle: peterheylands.com

Über mich



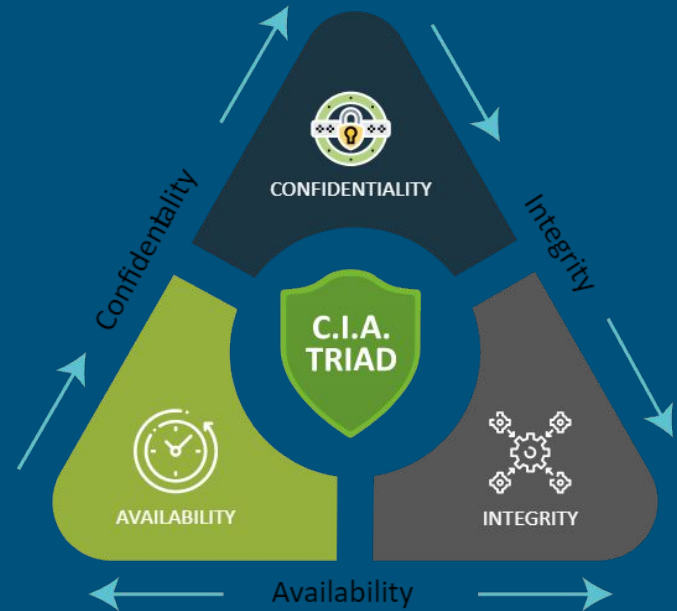
Gliederung

- I. Einführung
- II. Grenzen & Nutzen
- III. Basis-Anforderungen
- IV. In der Praxis

I) Einführung

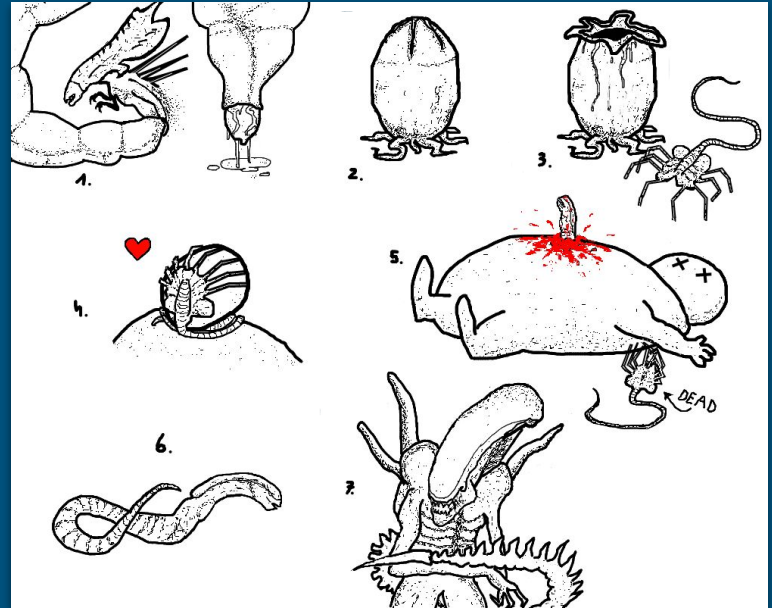
Begriffe

- **Softwaresicherheit**
 - Vertraulichkeit - Integrität - Verfügbarkeit (CIA Schutzziele)
 - Böswilligkeit und Unabsichtlichkeit



Begriffe

- **Sichere Softwareentwicklung**
 - CIA in jeder Phase des Entwicklungszyklus'
 - Planung & Spezifikation
 - **eigentliche Entwicklung** ♥
 - Tests & Freigabe
 - Auslieferung
 - Inbetriebnahme
 - Anpassungen



Meine Motivation

[ISO 27001]

*Informationstechnik – Sicherheitsverfahren –
Informationssicherheitsmanagementsysteme – Anforderungen*

→ **A.14.2.1 Richtlinie für sichere Entwicklung:**

*“Regeln für die Entwicklung von Software [...] sind **festgelegt**
und bei Entwicklungen in der Organisation **angewendet**.”*



© iso.org

Quelle: IT Crowd



Eure Motivation

[ISO 25010]

System und Software-Engineering –

Qualitätskriterien und Bewertung von System und Softwareprodukten (SQuaRE)



© iso.org

Functional Suitability	Performance Efficiency	Compatibility	Usability	Reliability	Security	Maintainability	Portability
<ul style="list-style-type: none">• Functional Completeness• Functional Correctness• Functional Appropriateness	<ul style="list-style-type: none">• Time Behaviour• Resource Utilization• Capacity	<ul style="list-style-type: none">• Co-existence• Interoperability	<ul style="list-style-type: none">• Appropriateness• Recognizability• Learnability• Operability• User Error Protection• User Interface Aesthetics	<ul style="list-style-type: none">• Maturity• Availability• Fault Tolerance• Recoverability	<ul style="list-style-type: none">• Confidentiality• Integrity• Non-repudiation• Authenticity• Accountability	<ul style="list-style-type: none">• Modularity• Reusability• Analysability• Modifiability• Testability	<ul style="list-style-type: none">• Adaptability• Installability• Replaceability

BSI IT-Grundschutz

CON: Konzeption und Vorgehensweise

- ↓ CON.1 Kryptokonzept
- ↓ CON.2 Datenschutz
- ↓ CON.3 Datensicherungskonzept
- ↓ CON.6 Löschen und Vernichten
- ↓ CON.7 Informationssicherheit auf Auslandsreisen
- ↓ CON.8 Software-Entwicklung ←
- ↓ CON.9 Informationsaustausch
- ↓ CON.10 Entwicklung von Webanwendungen
- ↓ CON.11.1 Geheimschutz VS-NUR FÜR DEN DIENSTGEBRAUCH (VS-NfD)



© LaundryFactory | redbubble.com

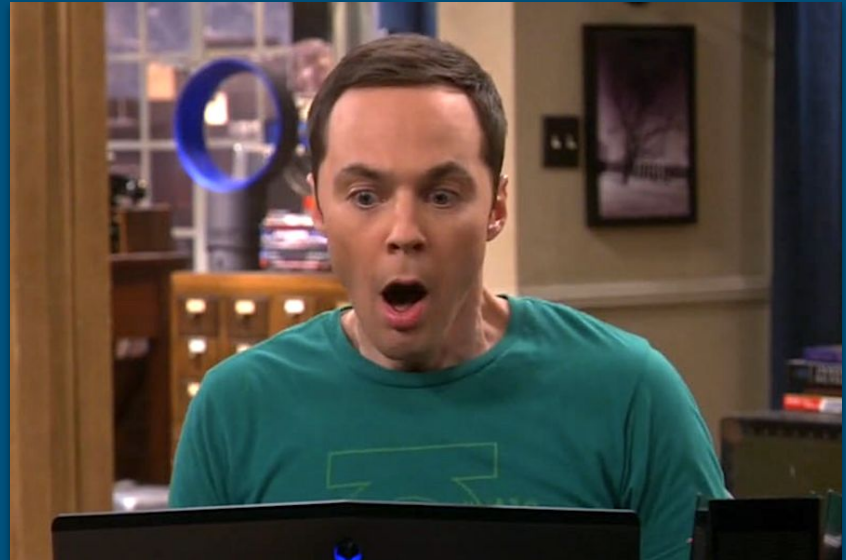
IT-Grundschutz-
Kompendium

[BSI CON.8]

II) Nutzen & Grenzen

Grenzen 🙄

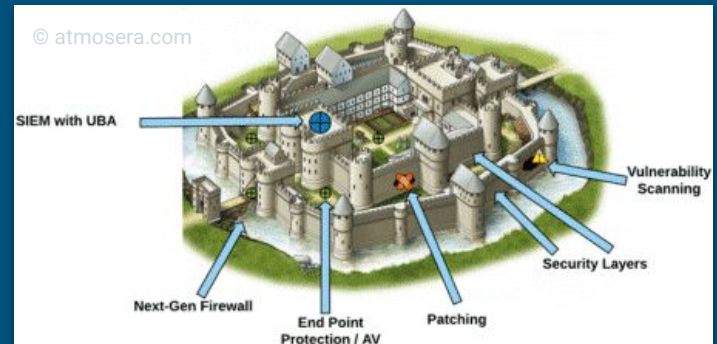
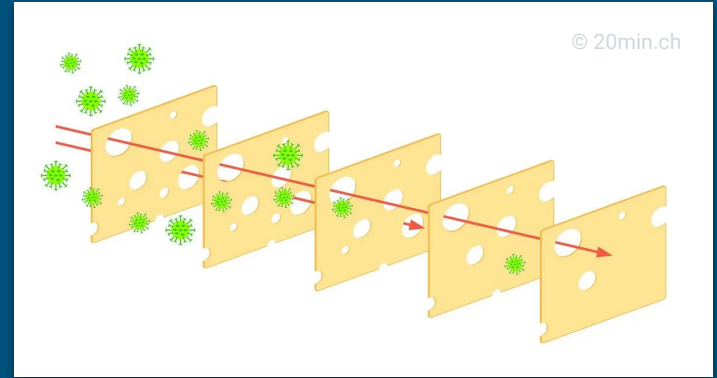
- Usability
- keine absolute Sicherheit
- niemals fertig
- Zeit



Quelle: Big Bang Theory

Nutzen 😊

- reduziertes Risiko bzgl. Schutzziele CIA
- Nachweis gegenüber Kunde
- Zertifizierungen / Wettbewerbsvorteil



Defense in Depth

III) Basis-Anforderungen

Auswahl eines **Vorgehens-** **modells** zur Software- Entwicklung

Basis-Anforderung **1**
(CON.8.A2)

1) Vorgehensmodell

- *“Sicherheitsanforderungen des Auftraggebers an **Vorgehensweise** im **Vorgehensmodell** integriert”*
- *“Personal in Methodik des **Vorgehensmodells** geschult”*

[CON.8.A2]



“Didn’t you get the memo?!”

1) Vorgehensmodell

Beispiele:

- Security Acceptance Criteria
[github.com | security AC]
- "AbUser"-Stories
aka Evil User Stories
- Security Stories

*"As (**malicious user**), I want (**malicious activity**), in order to (**business impact**)".*

"As an attacker, I want to try to guess a user's password by sending a large number of authentication requests in parallel to log in to his session".

*"As (**squad role**), I want to (**activity**), in order to (**prevent an evil user story from happening**)".*

"As a **developer**, I want to **set up a mechanism to block user accounts** after 5 attempts **to avoid brute force attacks**".

Auswahl einer **Entwicklungs- umgebung**

Basis-Anforderung **2**
(CON.8.A3)

2) Entwicklungsumgebung

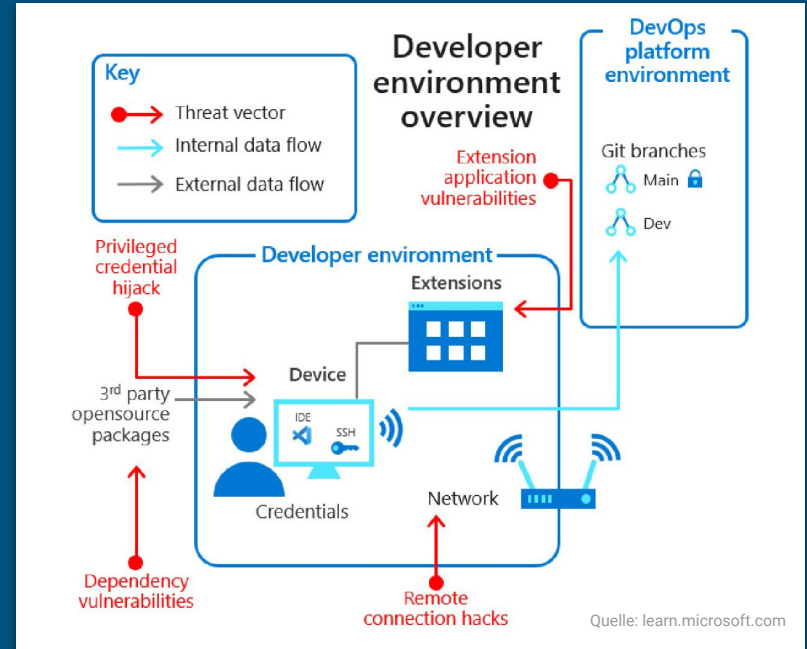
- *“Liste erforderlicher und optionaler Auswahlkriterien für eine **Entwicklungsumgebung** vom Verantwortlichen für die Software-Entwicklung erstellt”*
- *“**Entwicklungsumgebung** anhand der vorgegebenen Kriterien ausgewählt”*



2) Entwicklungsumgebung

Beispiele:

- erforderlich:
sicheres OS,
Auth via SSH
- optional:
welche IDE,
welches Git Tool



Sicheres
Systemdesign
in der zu
entwickelnden
Software

Basis-Anforderung **3**
(CON.8.A5)

3) Sicheres Systemdesign /1

- *“alle **Eingabedaten** vor der Weiterverarbeitung geprüft und **validiert**”*
[securecoding.com | input validation]
- *“bei Client-Server-Anwendungen Daten grundsätzlich auf **Server** validiert”*

[CON.8.A5]



3) Sicheres Systemdesign /1

Beispiele:

- Pflichtangaben, Min/Max Length, Dateiformat, Rechte, ...
- Whitelists statt Blacklists



3) Sicheres Systemdesign /2

- *“bei **Fehlern** oder **Ausfällen** von Komponenten des Systems keine schützenswerten Informationen preisgegeben”*
- *“Schützenswerte Daten **verschlüsselt** übertragen und gespeichert”*

[CON.8.A5]

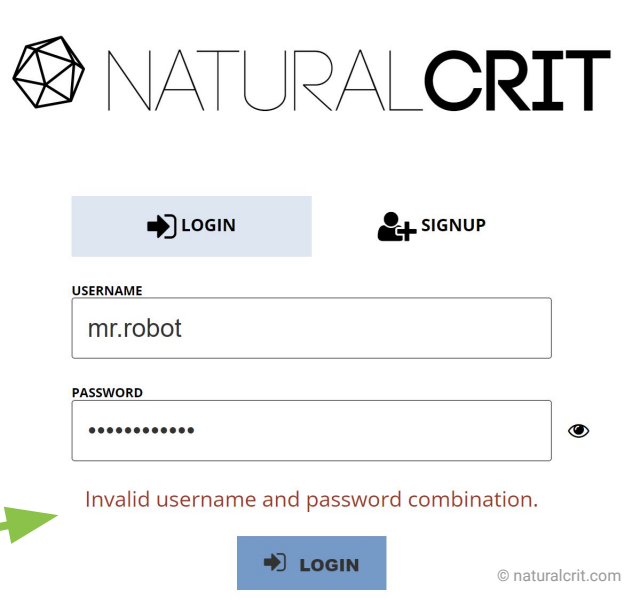


Quelle: James Bond

3) Sicheres Systemdesign /2

Beispiele:

- Security by Obscurity
- Development mode & Production mode

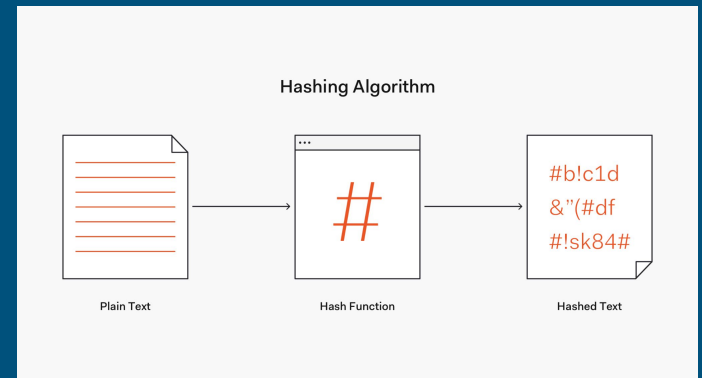
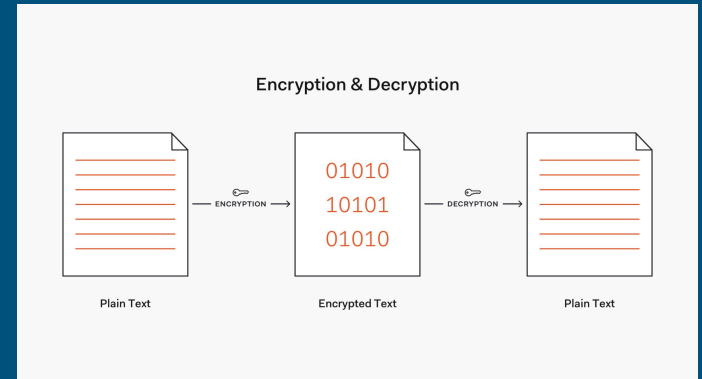


The screenshot shows the login interface for 'NATURAL CRIT'. At the top left is a logo consisting of a wireframe cube. To its right, the text 'NATURAL CRIT' is displayed in a large, thin, sans-serif font. Below the logo, there are two buttons: a light blue button with a right-pointing arrow and the text 'LOGIN', and a dark blue button with a person icon and a plus sign, labeled 'SIGNUP'. Underneath these buttons are two input fields. The first is labeled 'USERNAME' and contains the text 'mr.robot'. The second is labeled 'PASSWORD' and contains ten dots, indicating a masked password. To the right of the password field is an eye icon for toggling visibility. Below the password field, a red error message reads 'Invalid username and password combination.' A green arrow points from the left side of the slide towards this error message. At the bottom of the form, there is a dark blue button with a right-pointing arrow and the text 'LOGIN'. In the bottom right corner, the copyright notice '© naturalcrit.com' is visible.

3) Sicheres Systemdesign /3

- *“Benutzer-Authentisierung und **Authentifizierung** entsprechend **Sicherheitsanforderungen** der **Anwendung**”*
- *“Passwörter mit **sicherem Hashverfahren** gespeichert”*

[CON.8.A5]



3) Sicheres Systemdesign /3

Beispiele:

- Auth-Anforderung:
2FA,
Sperrung bei x Fehlversuchen
- PW-Hashverfahren:
~~MD5~~ ~~SHA-1~~ Argon2, Salted Hashes
[password-hashing.net]

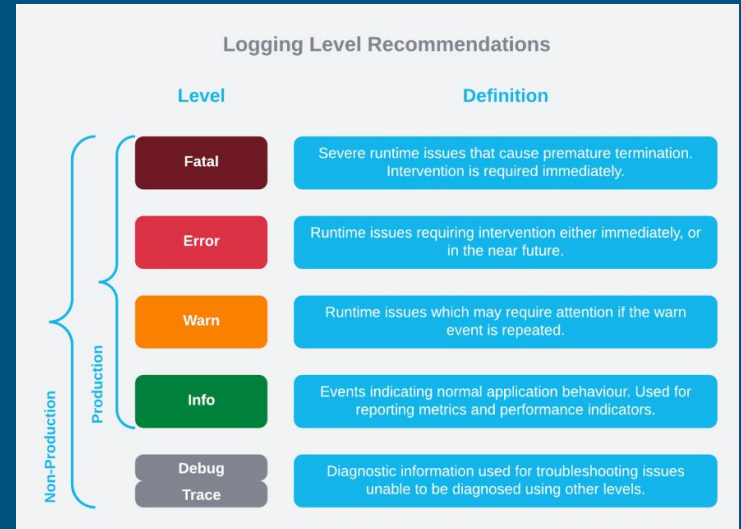


Quelle: kaspersky.de

3) Sicheres Systemdesign /4

- *“sicherheitsrelevante Ereignisse protokolliert”*
[medium.com | intro to logging]
- *“Programmcode und Konfigdateien bereinigt ausgeliefert”*

[CON.8.A5]



3) Sicheres Systemdesign /4

Beispiele:

- Protokolle:
DB-Fehler, Rechteänderungen, ...
- bereinigter Code:
keine Secrets, keine Kommentare, ...



Quelle: Der Tatortreiniger

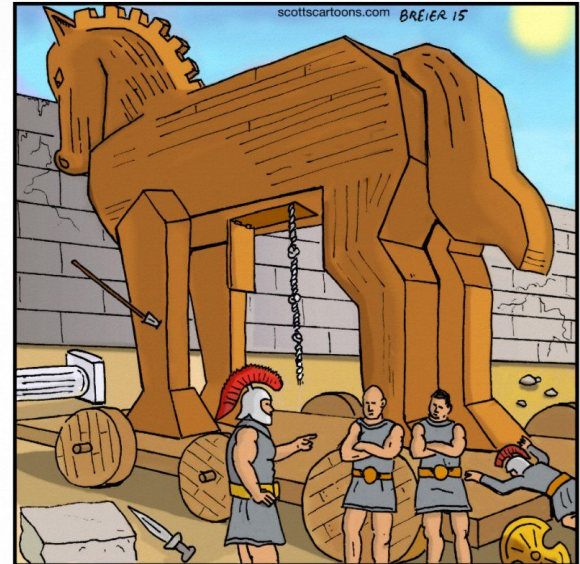
Verwendung
von **externen**
Bibliotheken
aus vertrauens-
würdigen
Quellen

Basis-Anforderung **4**
(CON.8.A6)

4) Bibos aus vertrauenswürdigen Quellen

- “externe **Bibliotheken** aus **vertrauenswürdigen** Quellen bezogen”
[choosing-a-third-party-library]
- “**Integrität** der externen Bibliotheken vor Verwendung sichergestellt”

[CON.8.A6]



“Norton! McAfee! How did you NOT detect this as a threat? You’re both useless, you know that?”

4) Bibos aus vertrauenswürdigen Quellen

Beispiele:

- Vertrauenswürdigkeit:
Anzahl Entwickler, Reputation,
Sponsoring, Contributors, ...
- Integrität:
Update Frequency, Anzahl Issues,
Alter, Versionen/Tags, ...

The screenshot shows the npmjs.org page for the package 'is-ten-thousand'. The package is version 0.2.0, published a year ago, and is public. It has 28 dependencies and 1 dependent. The page includes a 'Readme' tab, an 'Explore' button, and a '2 Versions' link. The main content area displays the package name 'is-ten-thousand' and a description: 'If you need to know if a number is ten thousand.' Below this, there are keywords: 'original is-ten-thousand is-thousand is-hundred is-ten beauty grace'. On the right side, there is an 'Install' section with a command: 'npm i is-ten-thousand'. Below that, there is a 'Weekly Downloads' chart showing 20 downloads. The 'Version' section shows '0.2.0' and the 'License' is 'WTFPL'. The 'Unpacked Size' is '7.08 kB' and the 'Total Files' is '5'. The source is listed as 'Quelle: npmjs.com'.

The screenshot shows a GitHub issue titled '#4 npmjs.org tells me that left-pad is not available (404 page)'. The issue is for the repository 'left-pad/left-pad' and has 193 comments. It was opened by 'silkenentrance' on March 22, 2016. To the right of the issue is a cartoon by xkcd. The cartoon is titled 'ALL MODERN DIGITAL INFRASTRUCTURE' and shows a complex, multi-tiered structure of servers and towers. A speech bubble from the cartoon says: 'A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003'. The cartoon is signed '© xkcd'.

Durchführung von entwicklungs- begleitenden **Software-Tests**

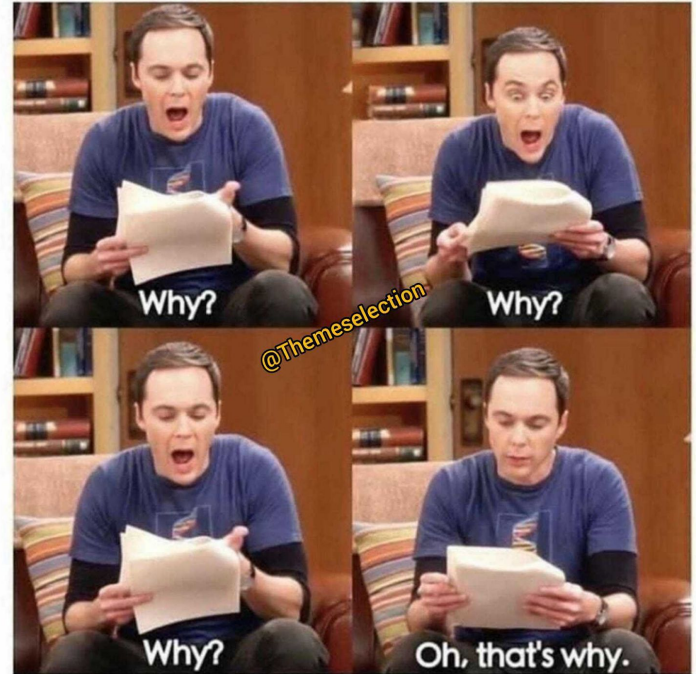
Basis-Anforderung **5**
(CON.8.A7)

5) Software-Tests /1

- “vor Freigabe entwicklungsbegleitende **Software-Tests** durchgeführt”
- “vor Freigabe Quellcode auf **Fehler** gesichtet”

[CON.8.A7]

Programmers while reviewing the codes



Quelle: Big Bang Theory

5) Software-Tests /1

Beispiele

- Komponenten- & Integrationstests, *NUnit*, *JUnit*, *JEST*, ...
- toolgestütztes Code Reviewing, *Github PR*, *Gitlab MR*, *Bitbucket PR*, ...

The screenshot shows a Bitbucket pull request page. At the top, it indicates the repository is 'Atlassian / Mobile / Alpha-team-app' and the pull request is 'MOB-22/Add packages to pipelines config'. The pull request is from branch 'MOB-123/add packages to pi...' to 'master' and is in an 'OPEN' state. It was created 11 hours ago and last updated one hour ago. There are 2 approvals and a 'Merge' button is visible.

A notification banner states: 'This pull request is ready to merge. All merge checks have passed. Merge pull request'.

The description of the pull request reads: 'This work focuses on fixing the issue raised by support about the layout issues. It should work on desktop browser and mobile browsers. I've run the tests and they pass locally. I'll check to make sure the pipelines tests pass as well.'

Summary statistics: 0 attachments, 2 comments, 2 commits (10 commits behind "staging", Sync now).

The code diff shows changes to 'buildtools / pipelines.yml':

```
@@ -17,22 +17,17 @@
19 19 # Only use spaces to indent your .yaml configuration.
20 20 # -----
21 21 # You can specify a custom docker image from Docker Hub at your build environment.
22 22 - image: openjdk:8
23 23 + image: java:8
24 24 pipelines:
25 25   default:
```

On the right side, there are three sections of checks:

- 3 of 3 checks passed: All tasks resolved, At least two approvals, No failed builds on last commit.
- 3 of 3 builds passed.
- 3 of 3 tasks resolved: add tests to address the edge cases, create a new class, update screenshots.

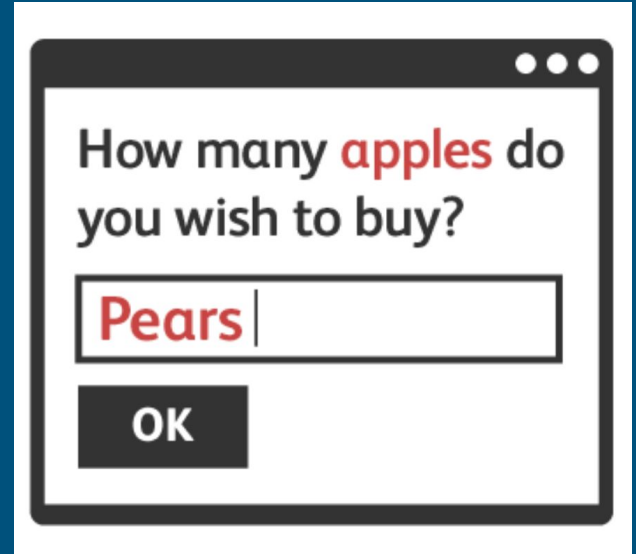
At the bottom right, a file list shows 'pipelines.yml' with 10 additions and 7 deletions, and 'build.gradle.kts' with 5 additions and 1 deletion.

Quelle: Atlassian Bitbucket

5) Software-Tests /2

- *“Tests umfassen die **funktionalen** und **nicht-funktionalen** Anforderungen der Software”*
- *“**Negativtests** abgedeckt & **kritische Grenzwerte** der Eingabe sowie der Datentypen überprüft”*
[\[smartsheet.com | negative-test-cases\]](https://smartsheet.com/negative-test-cases)

[CON.8.A7]

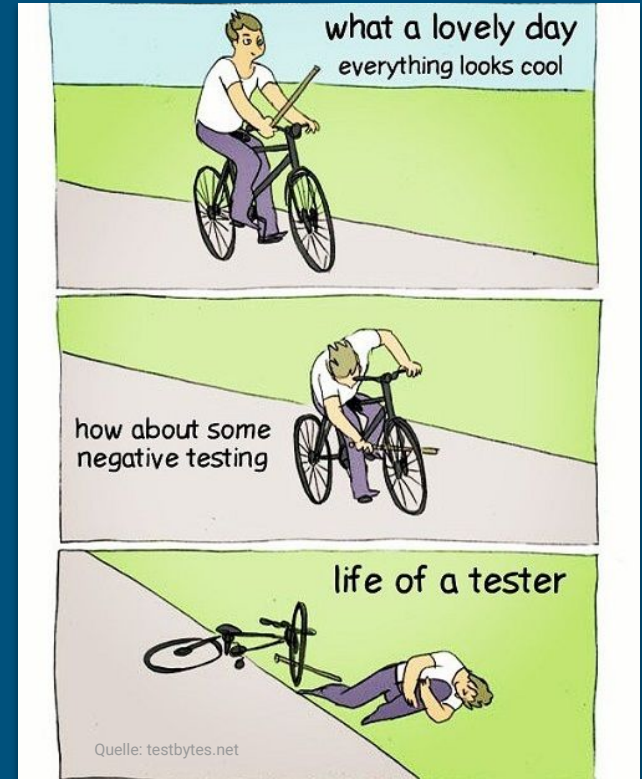


Quelle: soapui.org

5) Software-Tests /2

Beispiele

- funktional:
Komponententest
- nicht-funktional:
Performanztest
- Negativtest:
neg. Wert, null, n/a



5) Software-Tests /3

- “Software getestet in **Test- und Entwicklungsumgebung** getrennt von **Produktionsumgebung**”
- “automatische **statische Code-Analyse** durchgeführt”

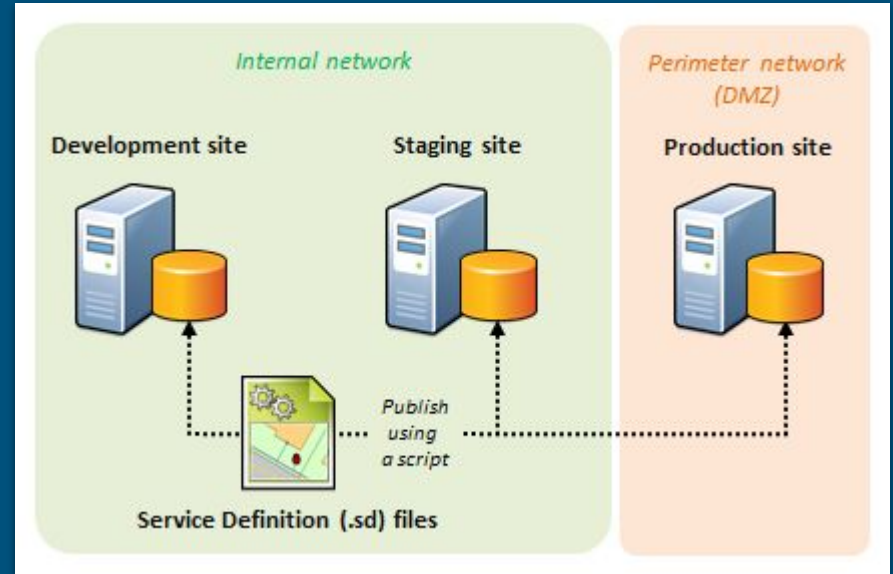
[CON.8.A7]



5) Software-Tests /3

Beispiele

- getrennte Umgebungen:
Staging Server in production mode, realitätsnah
- statische Analyse:
in IDE u/o in der CD/CI Pipeline,
z.B. *ESLint*, *SonarQube*



Bereitstellung
von **Patches**,
Updates &
Änderungen für
die entwickelte
Software

Basis-Anforderung **6**
(CON.8.A8)

6) Patches & Updates

- *“sicherheitskritische Patches und Updates für entwickelte Software zeitnah durch Entwickler bereitgestellt”*

[CON.8.A8]



Versions- verwaltung des Quellcodes

Basis-Anforderung **7**
(CON.8.A10)

7) Versionsverwaltung

- “Quellcode des Entwicklungsprojekts über **geeignete Versionsverwaltung** verwaltet”

[\[medium.com | securing-your-version-control\]](https://medium.com/_securing-your-version-control)

[CON.8.A10]



IN CASE OF FIRE 



1. git commit



2. git push



3. git out!

7) Versionsverwaltung

Beispiele:

- Force Pushing auf geteilten Branches verbieten
- Branching-Modell festlegen, z.B. *Git Flow*, *Trunk Based*, ...
- Pull/Merge Requests
- Tagging erfolgreicher Builds

	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Überprüfung von **externen** **Komponenten**

Basis-Anforderung **8**
(CON.8.A20)

8) Überprüfung externer Komponenten

- *“unbekannte externe Komponenten, deren Sicherheit nicht durch etablierte und anerkannte Peer-Reviews oder vergleichbares sicherstellbar, auf **Schwachstellen** überprüft”*

[CON.8.A20]



Quelle: Star Trek

8) Überprüfung externer Komponenten

Beispiel:

- Vulnerability Scanner Tools:
npm audit, SonarCube, BlackDuck, ...
[\[Vulnerability-Scanner-Tools\]](#)
- CVE Verzeichnisse:
cve.mitre.org, cvedetails.com, ...

```
PS C:\Users\james\Documents\nodeProjects\nodetest2\nodetest2> npm audit

=== npm audit security report ===

Manual Review
Some vulnerabilities require your attention to resolve
Visit https://go.npm.me/audit-guide for additional guidance

Low          Incorrect Handling of Non-Boolean Comparisons During
              Minification
Package      uglify-js
Patched in   >= 2.4.24
Dependency of jade
Path         jade > transformers > uglify-js
More info    https://nodesecurity.io/advisories/39

Low          Regular Expression Denial of Service
Package      uglify-js
Patched in   >=2.6.0
Dependency of jade
Path         jade > transformers > uglify-js
More info    https://nodesecurity.io/advisories/48

found 2 low severity vulnerabilities in 242 scanned packages
  2 vulnerabilities require manual review. See the full report for details.
PS C:\Users\james\Documents\nodeProjects\nodetest2\nodetest2>
```

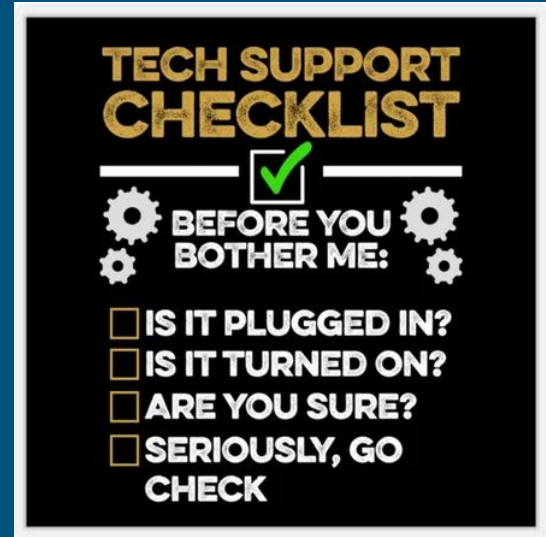
Zusammenfassung CON.8

- 1) Auswahl eines **Vorgehensmodells**
- 2) Auswahl einer **Entwicklungsumgebung**
- 3) Sicheres **Systemdesign**
- 4) **Bibliotheken** aus vertrauenswürdigen Quellen
- 5) Entwicklungsbegleitende **Software-Tests**
- 6) **Patches**, Updates und Änderungen
- 7) **Versionsverwaltung** des Quellcodes
- 8) Überprüfung von **externen Komponenten**

IV) In der Praxis

In der Praxis

- **Richtlinie** für Sichere Softwareentwicklung
- **Security Checklist** je Projekt
- 1 Verantwortlicher je Projekt
- DoD / Security Acceptance Criteria
- Capture the Flag 🚩



Quelle: KJ / etsy.com

In der Praxis

Checkliste_CON.8.xlsx | © BSI

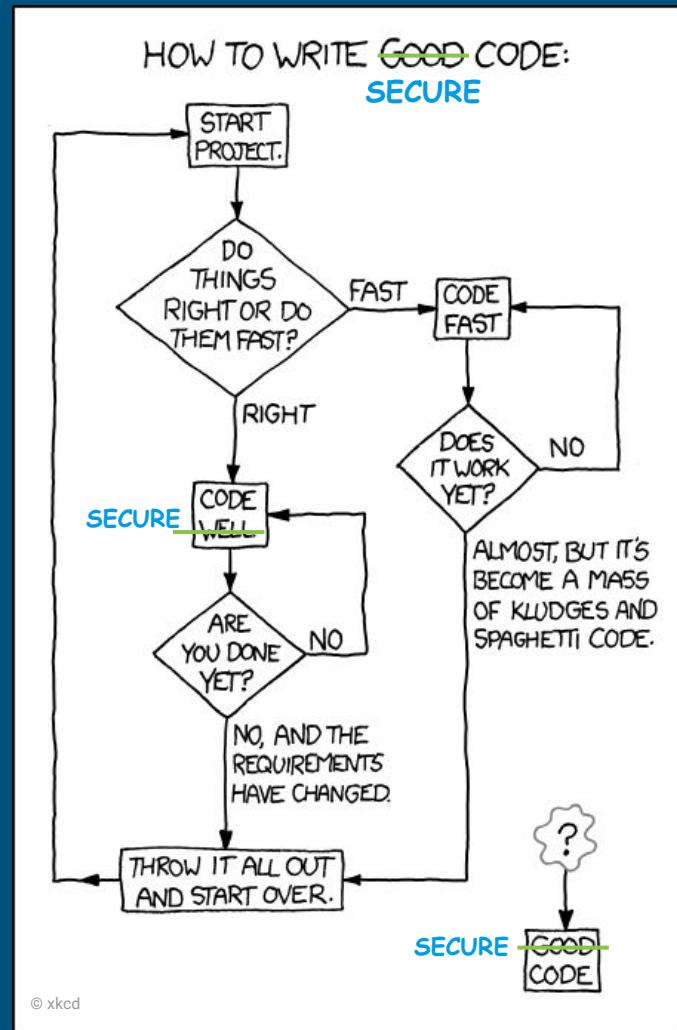
	A	B	C	D	E	F	G	H	I	J
1										
2		Baustein: CON.8 Software-Entwicklung								
3		Kompendium: 2023								
4										
5		ID-Anforderung	Titel	Inhalt	Typ	Entbehrlich	Begründung für Entbehrlichkeit	Umsetzung	Umsetzung bis	Verantwortlich
6		CON.8.A2	Auswahl eines Vorgehensmodells	Ein geeignetes Vorgehensmodell zur Software-Entwicklung MUSS festgelegt werden.	Basis					
7		CON.8.A2	Auswahl eines Vorgehensmodells	Anhand des gewählten Vorgehensmodells MUSS ein Ablaufplan für die Software-Entwicklung erstellt werden.	Basis					
8		CON.8.A2	Auswahl eines Vorgehensmodells	Die Sicherheitsanforderungen der Auftraggebenden an die Vorgehensweise MÜSSEN im Vorgehensmodell integriert werden.	Basis					
9		CON.8.A2	Auswahl eines Vorgehensmodells	Das ausgewählte Vorgehensmodell, einschließlich der festgelegten Sicherheitsanforderungen, MUSS eingehalten werden.	Basis					
10		CON.8.A2	Auswahl eines Vorgehensmodells	Das Personal SOLLTE in der Methodik des gewählten Vorgehensmodells geschult sein.	Basis					
11		CON.8.A3	Auswahl einer Entwicklungsumgebung	Eine Liste der erforderlichen und optionalen Auswahlkriterien für eine Entwicklungsumgebung MUSS von Fachverantwortlichen für die Software-Entwicklung erstellt werden.	Basis					

Fazit

Fazit



© Artistry Team / RedBubble



Weiterführend /1

1. IT-Grundschutz-Kompendium: **CON.8 Software-Entwicklung** | BSI 2023
bsi.bund.de/.../Grundschutz/IT-GS-Kompendium_Einzel_PDFs_2023/03_CON_Konzepte_und_Vorgehensweisen/CON_8_Software_Entwicklung_Edition_2023.html
2. IT-Grundschutz-Kompendium: **Checklisten** | BSI 2023
bsi.bund.de/.../Grundschutz/Kompendium/checklisten_2023.html?nn=128568
3. iso25000.com: **ISO/IEC 25010**
iso25000.com/index.php/en/iso-25000-standards/iso-25010/
4. ScienceSoft: **Sichere Softwareentwicklung: Schritt-für-Schritt-Anleitung** | D. Dmitry Nikolaenya, 2019
scnsoft.de/blog/sichere-softwareentwicklung

Weiterführend /2

- [5] zu CON 8.A.2: **Ein Vorgehensmodell für die Entwicklung sicherer Software** | A Lunkeit, W. Zimmer 2016
syssec.at/de/veranstaltungen/.../dachsecurity2016/papers/DACH_Security_2016_Paper_22B3.pdf
- [6] zu CON 8.A.2: OWASP: **Security Acceptance Criteria** | E. Johnson 2018
github.com/OWASP/user-security-stories/blob/master/security-acceptance-criteria.md
- [7] zu CON 8.A.3: National Cyber Security Center: **Secure development and deployment guidance – Secure your development environment**
ncsc.gov.uk/collection/developers-collection/principles/secure-your-development-environment
- [8] zu CON 8.A.5: Medium: **Password Hashing: Scrypt, Bcrypt and ARGON2** | M. Prezioso 2019
medium.com/analytics-vidhya/password-hashing-pbkdf2-scrypt-bcrypt-and-argon2-e25aaf41598e
- [9] zu CON 8.A.5: SecureCoding: **Input Validation: Client-Side & Server-Side Cybersecurity Deterrent** | 2021
<https://www.securecoding.com/blog/input-validation/>

Weiterführend /3

- [9] zu CON 8.A.6: Medium: **Choosing a Third-Party Library** | D. Scheider 2021
dana-scheider.medium.com/choosing-a-third-party-library-e8b0f7aa9497
- [10] zu CON 8.A.7: smartsheet: **Everything You Need to Know About Negative Test Cases** | K. Eby 2021
<https://www.smartsheet.com/negative-test-cases>
- [11] zu CON 8.A.8: **Wie man mit 3rd Party Libraries umgeht: Strategien für den erfolgreichen Einsatz von Fremdbibliotheken** | 2015
entwickler.de/iot/wie-man-mit-3rd-party-libraries-umgeht-strategien-fur-den-erfolgreichen-einsatz...
- [12] zu CON 8.A.10: Medium: **Code Security – The Importance of Securing Your Version Control** | L. Oliff 2018
medium.com/@lukeocodes/the-importance-of-securing-your-version-control-131841429994
- [13] zu CON 8.A.20: DNSstuff: **Top 15 der kostenpflichtigen und kostenfreien Vulnerability-Scanner-Tools** | 2020
dnsstuff.com/de/network-vulnerability-schwachstellen-scanner