ABB Technikerschule

Technik. Informatik. Wirtschaft. Management →

Dominik Meyer
Software Engineering

Application Security *angewandt*

Puffer

Kleiner Puffer 10min für Präsentation «Security» Task Lektion 7

Ziel

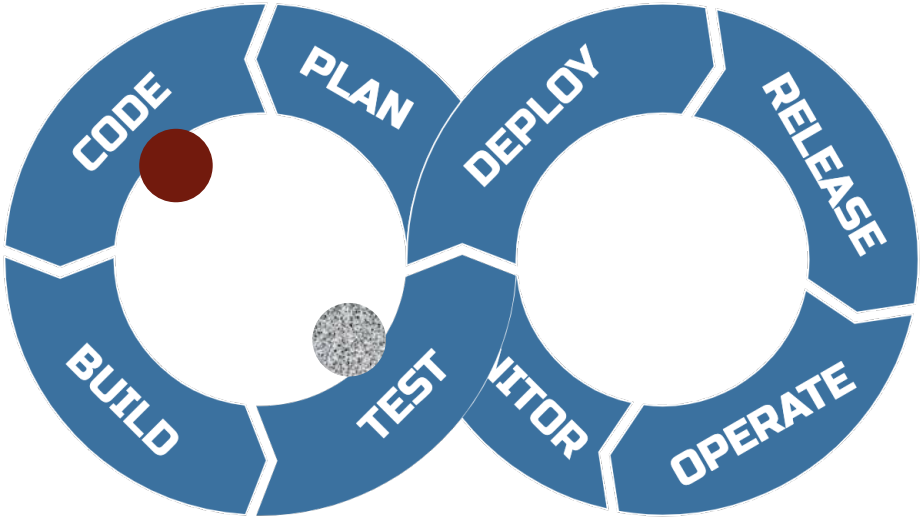
Nach der Lektion können die Studierenden einen sicheren Endpunkt mit Basic Authentication in ihre Applikation einbauen und haben dies am Dozenten-Starter getestet.

Docker

– Lektion 2 – HTTP(S) Headers und Client-Server Prinzip
– Lektion 6 – sicheren Docker Container erstellt
– Lektion 7 – Grundprinzipien Sicherheit erarbeitet
– Lektion 8 – eigene App, Dependencies, DRY

Disclaimer: Der Inhalt heute macht unsere App nicht bombensicher. Wir lernen wieder einen «Ansatz».

Output: Starter App mit einem gesicherten Endpunkt



Merke

Dozent reitet auf Themen rum

- Wer es noch nicht gemerkt hat, ich reite immer auf den gleichen Themen rum
  - Git
  - Container und Docker
  - Continuous Integration
  - Dependencies

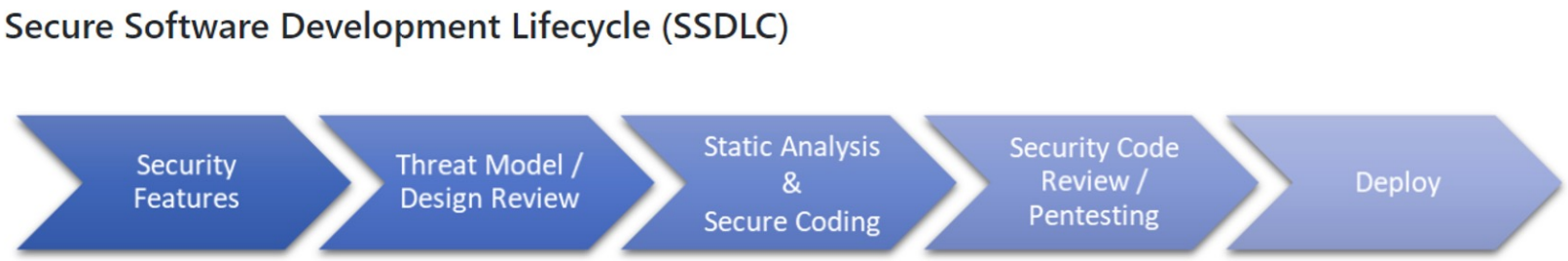
- Das ist nicht **ohne Grund**
  - Analogie: Diese Themen sind wie beim Autofahren Benzin und Lenkrad
  - Euer Urteil jetzt: Zu kompliziert, zu viel Mehraufwand, kenne es schon ohne
  - DevOps Mentalität: `Do it once, twice, now automate`
  - DRY – MA nutzt bereits Wordpress → Auth, DB, etc. gelöst!

- Wir haben in späteren Lektionen die Möglichkeit etwas zu «wiederholen»
  - aber wir müssen uns mit diesen Themen **beschäftigen**!
  - on Demand (ihr fragt nach Hilfe)

Agenda

- Threat Modeling
  - Authentication
- Spring Web Security (Dependency)
- Einfaches Beispiel
  - Unterricht geklont
  - Transfer selber erstellt
- Analyse
- Zielkontrolle

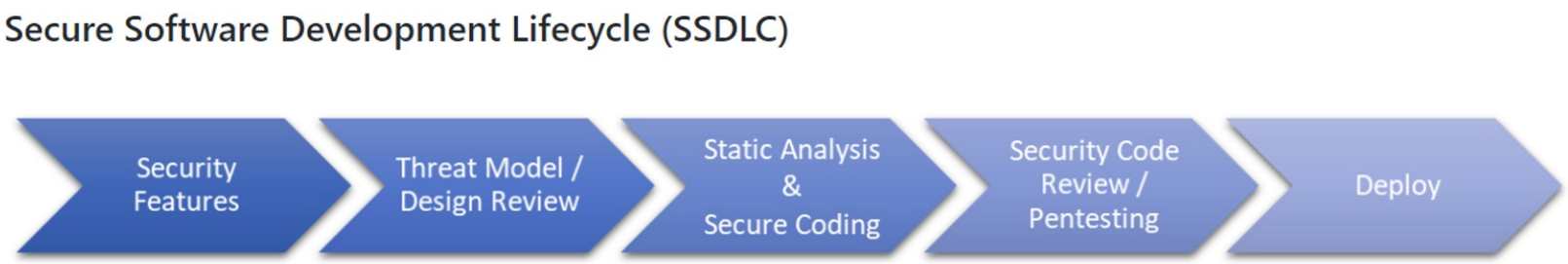
SSDLC



- Simple Threat Model for
  - Spring Starter App
  - Durch Dozent hinzugefügter unsicherer Endpunkt (/employees)
    - Holt private Daten von allen Mitarbeitenden der Firma «Science Expeditions» (ExMan)
  - Gedanklich in die Cloud deployt (z.B. exman.abbts.ch)

- Welche Angriffsvektoren können wir feststellen?

SSDLC



- Threat Model
  - Spring Starter App
  - Durch Dozent hinzugefügter unsicherer Endpunkt (/employees)

- Threat #1
  - Diesen Endpunkt, in Fakt aktuell alles, kann JEDER nutzen
  - Für später absolut katastrophal
  - Sehr wenige Funktionalitäten sollten anonym nutzbar sein
    - schon nur wegen Rate Limit, Audit, (D)DOS
    - Gesund ist, jeden Aufruf zu Authentifizieren (NICHT unbedingt Autorisieren)
    - kleines Ausnahme Beispiel: Version-Info Endpoint (aber auch das wäre Rate Limit schlau)

- weitere Angriffspunkte «verdrängen» wir in die Modularbeit 😊
  - z.B. SQL Injection?

Herleitung

Secure Software Development Lifecycle (SSDLC)

spring-starter

GET /employees

Etwas fehlt für Authentifizierung…

?

Identity Providers

Wir überlassen die Identität (Authentication) jemand anderem!

- jemandem, der es viel besser(sicherer) kann
- jemandem, dessen Beruf «Identitiy Management» ist

- Wordpress
- Azure AD, Amazon IAM, Google IAM
- Social Logins (Facebook, Apple, Google, etc.)
- SaaS Spezialisten
  - https://auth0.com
  - https://www.onelogin.com

- Mischlösung
  - https://www.keycloak.org
    - Jemand anderer entwickelt, wir hosten (Open Source)
    - Kosten umgewälzt, statt Lizenz → Wartungsaufwand

Spring Web Security

– DRY, wir entwickeln keine Authentication selber
  – Keine selber gewerkelte 🌭 Authentication ist production-ready
  – ausser man heisst: Google, Netflix, Spotify etc.
    – nicht mal die, die nutzen z.T. Social Logins
    – Swiss eID wäre z.B. ein trustworthy-er IDP geworden
  – Ausnahmen überall, auch bei MA, aber dann mit sauberer Risikoanalyse und Konzept wie das in Produktion «betrieben» werden wird
  – Kleine Analogie zum Verständnis: Wenn wir ein strassen-zugelassenes Auto bauen, kaufen wir die Gurten und Airbags im zertifizierten Fachgeschäft oder basteln wir selber welche?

https://spring.io/projects/spring-security

– z.B Azure AD

```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-security'
    implementation 'org.springframework.security:spring-security-test'
}
```

Spring Web Security

Spring Security is a powerful and highly customizable authentication and access-control framework. It is the de-facto standard for securing Spring-based applications.

Spring Security is a framework that focuses on providing both authentication and authorization to Java applications. Like all Spring projects, the real power of Spring Security is found in how easily it can be extended to meet custom requirements

Features

- Comprehensive and extensible support for both Authentication and Authorization
- Protection against attacks like session fixation, clickjacking, cross site request forgery, etc
- Servlet API integration
- Optional integration with Spring Web MVC
- Much more...

Task

Security konfigurieren

AUFTRAG

- Check Sie die Lösung aus
  - `git fetch --all`
  - `git checkout 0.1.1` (auf dem Dozenten-Starter)

- Inspizieren Sie für 5 Minuten
  src/main/java/ch/abbts/nds/swe/swdt/starter/CustomWebSecurityConfigurerAdapter.java

- Wir wollen (u.a. im Transfer) einen sicheren* Endpunkt verbauen, was brauchen wir dazu?

* sicher im Bezug auf Authentication

Zielkontrolle

Die Studierenden haben einen sicheren Endpunkt mit Basic Authentication an dem Dozenten-Starter getestet.

Was ist Authorization: `Basic dXNlcjpwYXNzd29yZA==`?

Welcher Teil unserer App, zum Zeitpunkt jetzt, ist nicht bereit für die Produktion und was fehlt dazu?

Hinweis: Es ist keine MA Vorgabe einen Identity Provider zu nutzen. Ebenso ist es erlaubt selber ne Authentication zu bauen (oder im Pflichtenheft ist gar keine). Gefordert ist aber, dass Sie sich intensiv Gedanken machen wie die Lösung in der Produktion läuft / laufen kann!

Unser Beispiel heute erhielt keinen Test! Schwach, aber für heute OK.

Exkurs: Base64

Replit.com kann man Online zusammen coden… 🎉

https://replit.com/join/rmxhzfuj-mambax

base64encode.org

Text zu Binär
Das ist kein Security Feature, auch wenn es «hashed erscheint»
Primär für Transportzwecke, respektive dessen Vereinfachung

WEITER WISSEN.

Wir begleiten Sie!