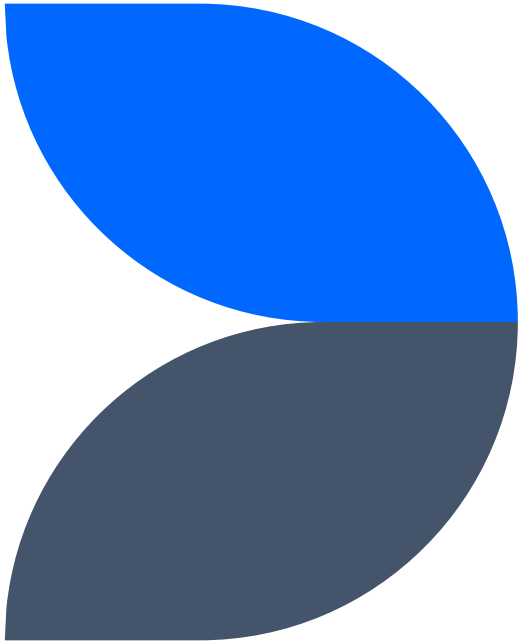
Notiz-App

Kurs: Secure Software Engineering

Aufbau

Frameworks & Entwurf



Frontend - Vue

Eigenschaften

- Javascript Framework
- Clientseitig
- Single Page
- Open Source
- View-orientiert
- Leichtgewichtig
- Nutzt Virtual DOM

Sicherheit

- Filtert Input
- Fulltime Contributor in Bereitschaft
- Escaping content (HTML, Attributebindings, <style>)
- Einstellung des Supports unwahrscheinlich

Backend - Express

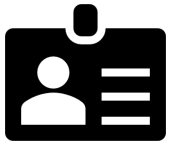
Eigenschaften

- Webframework
- Open Source
- Leichtgewichtig
- Nutzt Node.js
- Einfach zu erlernen / den Programmierern vertraut

Sicherheit

- Einfacherer Code -> sicherere Anwendung
- Viele verträgliche Sicherheitspackages als Middleware
- Einstellung des Supports unwahrscheinlich

Besonders zu schützen

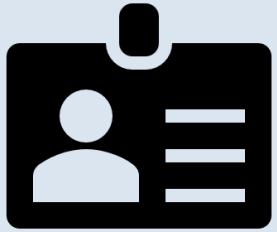


Sessions

Frontend <-> Backend

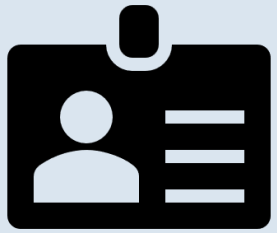
Datenbankinhalte

Uns selbst



Sessions

- Grundlage für Userkonten
  - Privat- / öffentlich-Trennung
  - Notizautoren
  - Rechte zur Veränderung / Löschung von Notizen



Gefahren

CSRF

A03: Injection

Schwaches Passwort

A04: Insecure Design

Unsicherer Ein/Auslog-Mechanismus

A07: Identification and Authorization Failure

Keine / schwache Verschlüsselung

A02: Cryptographic Failures

CSRF

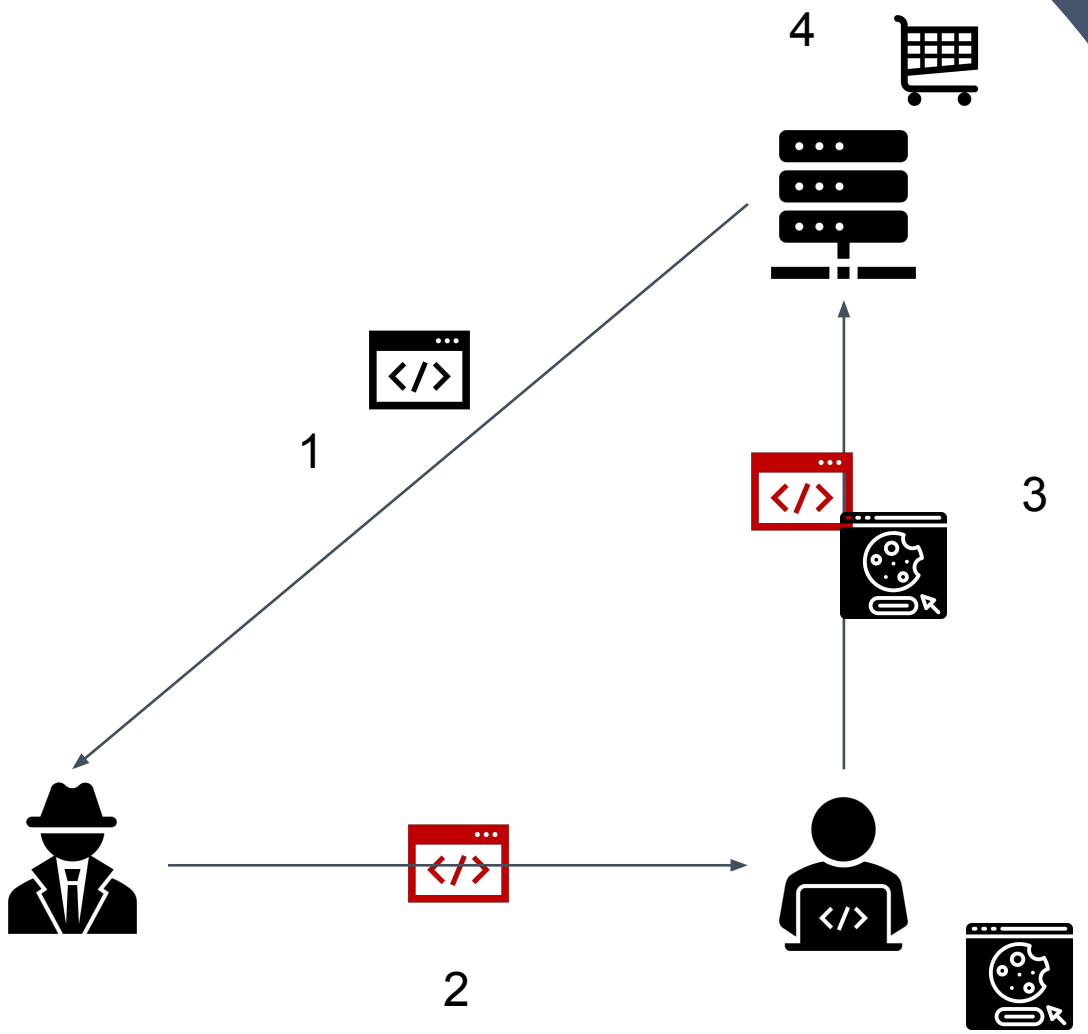
Cross-Site Request Forgery

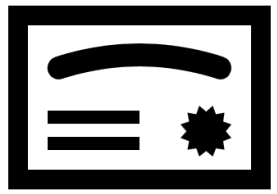
Browser speichert Cookie

Link löst eine zustandsändernde Anfrage aus

Voraussetzungen:

- Nutzer ist bereits eingeloggt
- Nutzer klickt auf manipulierten Link
- Angreifer kennt Nachrichtenstruktur





CSRF-Token

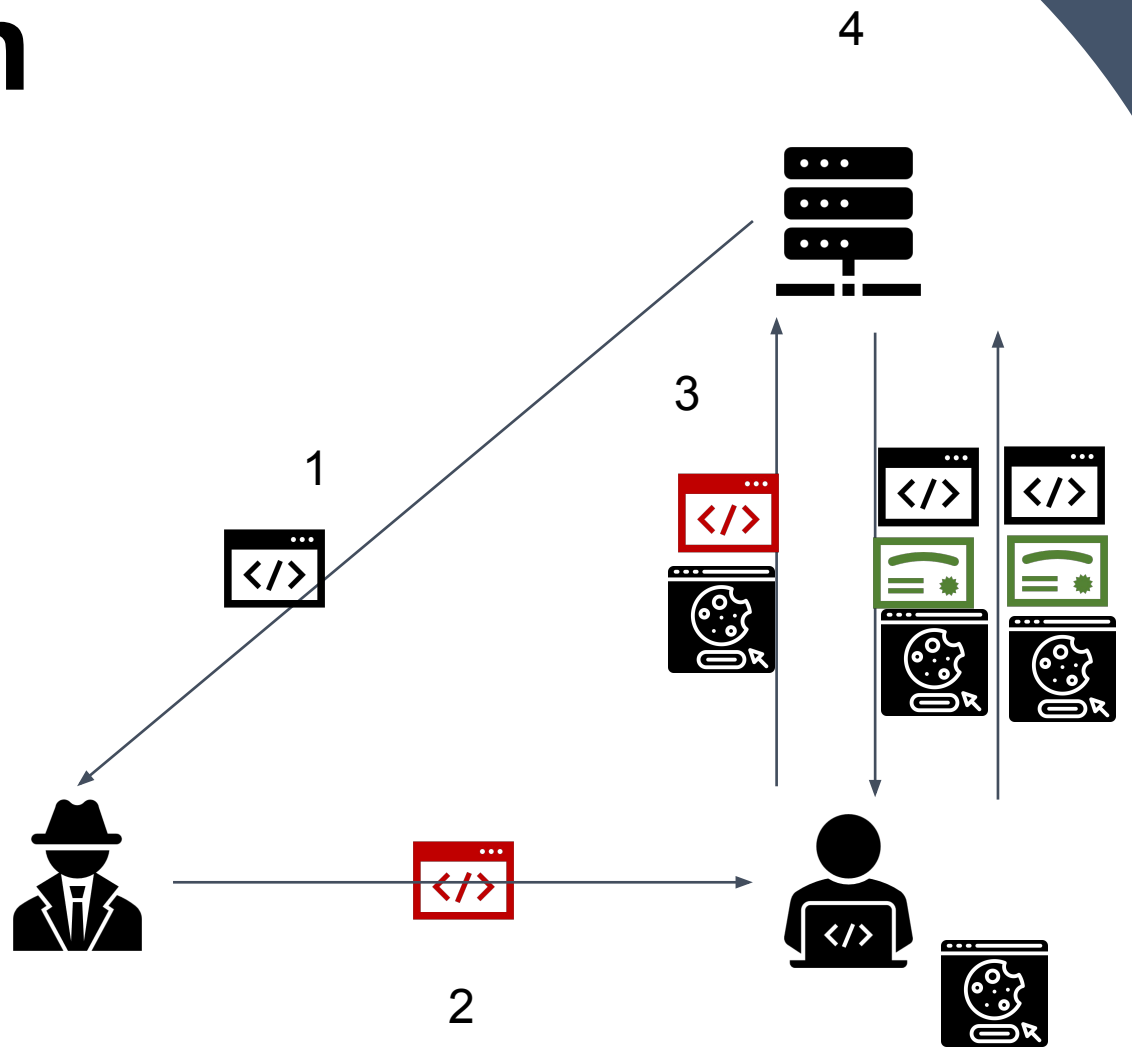
Zusätzlich zu Session

i.d.R. zufällig erstellter String

Nicht permanent im Browser gespeichert

Verschiedene Möglichkeiten: Header, Body …

Package: CSURF



Schwaches Passwort

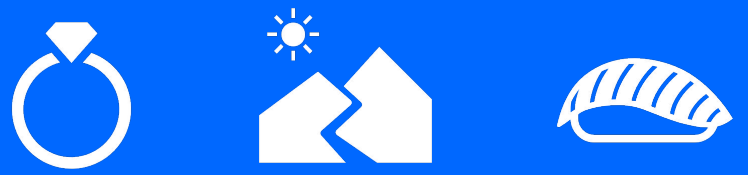
Ein schwaches Passwort bleibt schwach!

Umfangreiche Listen bei Brute Force Attacken

Starkes Passwort erzwingen!
Keine Klebebandsätze, L33t-Speak usw.!
Gut merkbar! (Noch besser, Passwortmanager)

Methode 3:

Soaring thoughts nevermore fall



Verlobung Zugspitze Sashimi

ZXCVBN

Hilfreiche Rückmeldungen

Schränkt Stil nicht ein

Basiert auf Wahrscheinlichkeit des Knackens, nicht auf Regex

Erkennt Muster

-> fester Teil von Registrierungsfunktion im BE

-> + Abfragbare Route

Verschlüsselung / Unsicheres Ein/Ausloggen

HTTPS

Keine Klartextübertragung mehr

Ermöglicht durch NGINX als reverse-proxy zum backend.

Umleitung auf HTTPS

NGINX Server, der das Frontend hostet, leitet immer auf die HTTPS Seite um.

Express-Session

Prüft Session-Cookies serverseitig

Validierungs-schritte nach Aufwendigkeit staffeln

Nachrichten

- Anfälligster Teil
  - Viele Angriffswege
  - Nutzereingaben (Umgehen des Frontend möglich)
  - Viele Möglichkeiten der Fehlkonfiguration
    - Cookies
    - Sessions
    - Proxy
    - Ratelimit
    - …
- Unverzichtbar

Gefahren

XSS

A03: Injection

Keine / fehlerhafte Verschlüsselung

A02: Cryptographic Failures

Misskonfiguration

A05: Security Misconfiguration

Denial of Service

A05: Security Misconfiguration

XSS

Wenn Nutzer ihren eigenen Code einfügen können.

Bei uns: Markdown, HTML (Kann Javascript verstecken)

HTML-sanitization: unerwünschte Tags entfernen/ escapen

Sanitize-html: Schädliche Inhalte wie JavaScript in a-tags werden gefiltert

"Amazon s3 buckets start life completely locked down. […]
So every breach [with them], and it's been billions of records over the last couple years, is because somebody accidentally assigned too permissive a policy […].
When you're trying to build something its like 'well, we'll just give it more permissions. […]'
And they never lock them down again."

Mark Nunnikhoven,

Vice President of Cloud Research at Trend Micro

Rate Limiting

Gegen DOS

Verhindert Überlasten des Servers durch Anfragen **von derselben** IP-Adresse

Nur bedingt bis gar nicht hilfreich gegen DDOS

Gegen Brute Force

Angriffe sollen sich nicht mehr lohnen!

Ressource: Zeit

Datenbank

- Schaden durch Input verhindern!
- Speichert Daten -> Endziel vieler Angreifer
  - Datenverlust : unschön
  - Datenleak : Katastrophe

Datenbank - PostgreSQL

Eigenschaften

- Objektrelational
- Schlanker als z.B. Oracle
- Open Source

Sicherheit

- Integritätsschutz

Ein weiterer Faktor:
Backups

SQL-Injections

Datenbank bereits im Backend schützen!

Art des Client von Bedeutung

Account des BE mit minimalen Berechtigungen

Falls der Angreifer doch Zugriff kriegt…

Passworthashing

Achtung: Übliche Hashalgorithmen nicht unbedingt geeignet!

Ziel: So teuer wie möglich sein!

Argon2id

- Sieger der Password Hashing Competition
- Von Owasp empfohlen
- Einstellbarer CPU und Speicheraufwand

+ automatisch erstellter, zufälliger Salt

-> Nachschauen und Regenbogentabellen sollen sich nicht mehr lohnen!

Rechtliches

Nur essentielle Cookies verwendet

-> keine Zustimmung nötig, aber Aufklärung über die Nutzung

Impressum

-> Deutsch und Englisch

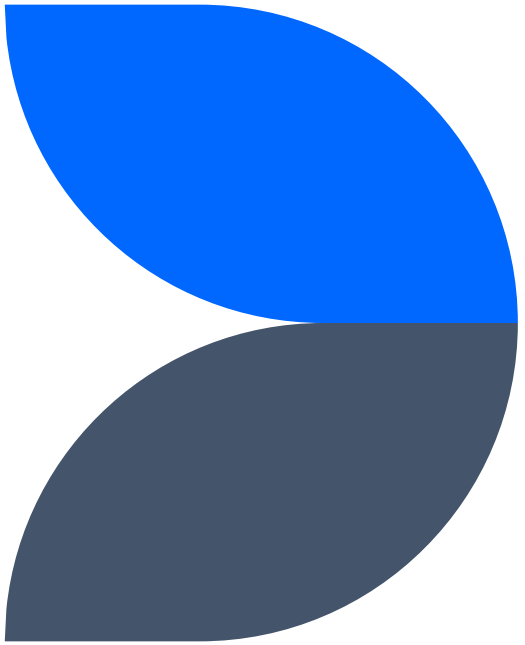
Datenleak

-> Datenschutzaufsichtsbehörde informieren

-> Keine E-Mails -> Benachrichtigung über Landing Page

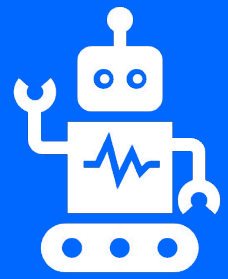
CI & Deployment

Testen, Einstellen, Testen…



Continuous Integration

- GitHub Actions
  - Backendtests für node Versionen 18.x, 16.x
  - Dependency Analysis
  - Snyk
  - CodeQL
  - Hound (noch nicht eingestellt)



Dependabot

Weitere Maßnahmen

Notwendigkeits- und Sicherheitsanalyse für Dependencies in der Dokumentation

Security Policy (ohne echte Mailadresse)

Branch Protection -> Development & Main, Aktualitätscheck, Approval nötig

.env nicht im Repo (nur .testenv) -> kann nicht ausversehen gepusht werden

In einer perfekten Welt

Was sich noch ergänzen ließe,
wenn man Zeit/Geld hätte…

Ideen und Verbesserungen

- Hardware Security Module für Secrets
- Pepper
- Unterstützen von Passwortänderung
- Ersatzseite/Server falls Ausfall
- Pen-Tests, Pen-Tests, Pen-Tests!
- Live-Feedback bei Passworteingabe im Registerfeld
- Loggen: Belastung, Auslastung, zerstörerische Benutzergruppen
- Sperren von Nutzeraccounts ermöglichen

Fragen?

Gleich Live Demo

Passwort-Studie

Yıldırım, M., Mackie, I. Encouraging users to improve password security and memorability. *Int. J. Inf. Secur.* **18,** 741–759 (2019). https://doi.org/10.1007/s10207-019-00429-y

https://rdcu.be/cSuFc

Bildquellen

„Schloss öffnen Icons“ erstellt von Ehtisham Abid – Flaticon
https://www.flaticon.com/de/kostenlose-icons/schloss-offnen

“Cookie icons” erstellt von Smashicons - Flaticon https://www.flaticon.com/free-icons/cookie

„durchlaufzeit Icons“ erstellt von Aranagraphics - Flaticon
https://www.flaticon.com/de/kostenlose-icons/durchlaufzeit

Abgelehnt Icons erstellt von Slidicon - Flaticon