SECURITY

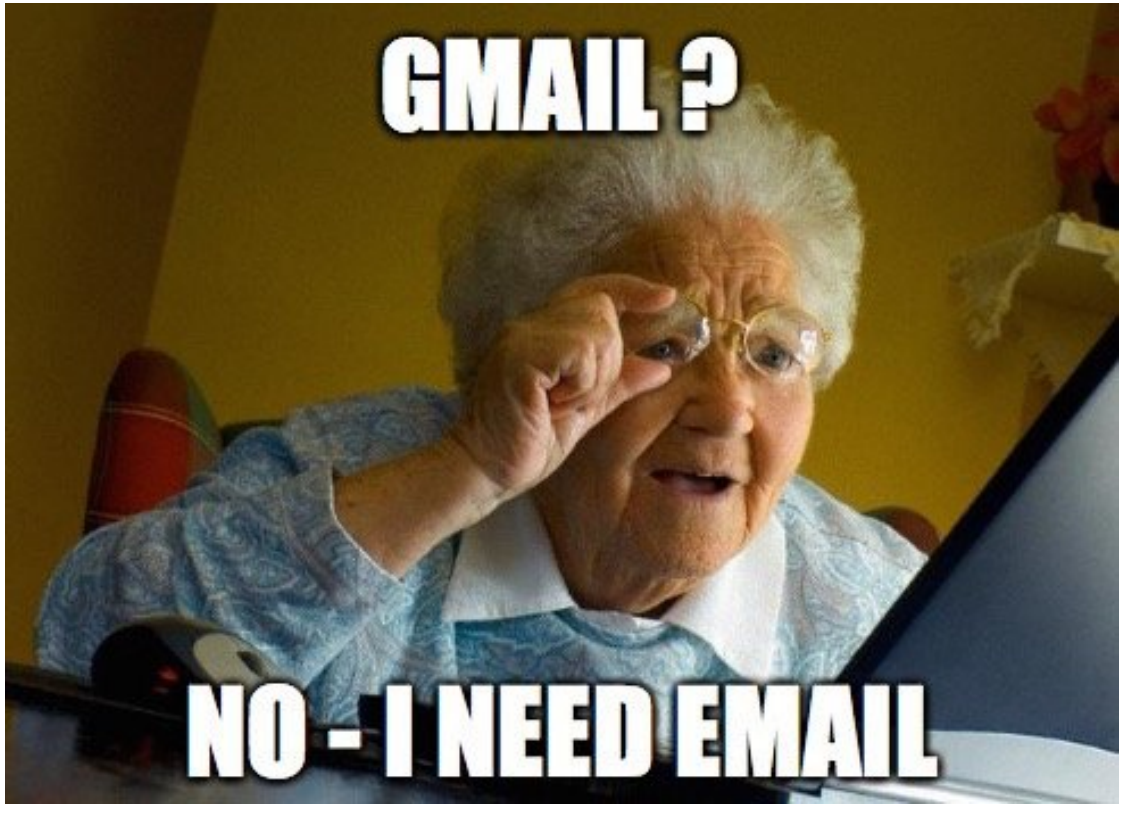
MOTIVATION

UM WAS GEHT ES?

- Datendiebstahl
- Datenveränderung
- Computersabotage
- Computerbetrug

INTERNETNUTZUNG

- (fast) jeder nutzt das Internet



INTERNETNUTZUNG

- das Internet wird für (fast) alles verwendet
- zum Beispiel:
  - Soziale Netzwerke
  - Shopping
  - Banking

INTERNETNUTZUNG

- oft geht es um sicherheitskritische Daten
- zum Beispiel:
  - Privatsphäre & Datenschutz
  - Geld Transaktionen
  - Identitätsdiebstahl

WEBANWENDUNGEN

- sind von überall erreichbar
  - nicht wie ein Bankautomat
- IT-Sicherheit ist keine Einstiegshürde
  - jeder kann mit einem Tutorial ein Webshop schreiben

UNSICHERE WEBANWENDUNGEN

- selbst die größten sind nicht sicher
- man hört immer wieder von Datenleaks oder Einbrüchen
  - Twitch
  - Facebook
  - etc.
- größere Webanwendungen sind attraktivere Ziele

ATTRAKTIVE ZIELE

- einige Ziele sind sehr attraktiv
  - Twitch
  - Facebook
  - Banken
  - Passwortmanager
- dies liegt
  - an der Größe
  - an den verwalteten Daten
- nicht lukrative Ziele müssen trotzdem sicher sein!

NACHTEILE VON SECURITY

- Security bringt Nachteile mit sich
- konkurriert mit der Benutzbarkeit
  - zwei Faktor Authentifizierung (2FA)
- höhere Entwicklungskosten
- komplexere Architektur
- höherer Ressourcenverbrauch

WIE VIEL SICHERHEIT BRAUCHE ICH?

- "it depends"
- es kommt an auf
  - die Anforderungen
  - das Budget
  - die Domäne
  - rechtliche Rahmenbedingungen
- Security ist eine Qualitätsanforderung

MINDESTMASS AN SECURITY

- nicht immer die wichtigste Qualitätsanforderung
- ein Mindestmaß muss vorhanden sein
- dieses Mindestmaß schauen wir uns nun an

ANGRIFFSMETHODEN

ANGRIFFSMETHODEN

- Request-Manipulation
- Directory Traversal
- SQL-Injection
- Session Hijacking
- Cross-Site-Scripting
- Cross-Site-Request-Forgery
- Man-In-The-Browser
- Phishing
- Denail-Of-Service

REQUEST-MANIPULATION

Wer hat davon mitbekommen?

REQUEST-MANIPULATION

- *"www.some-domain.de/users/41"*
- gibt es vielleicht auch einen user 42?
- alle öffentlichen Schnittstellen können aufgerufen werden

REQUEST-MANIPULATION - LÖSUNG

- Datenzugriff nur für berechtigte und authentifizierte Nutzer
- evtl. zusätzlich keine monoton Aufsteigende Id's

DIRECTORY TRAVERSAL

- ähnlich wie Request-Manipulation
- *"http://www.example.com/index.foo?item=datei1.html"*
- *"http://www.example.com/index.foo?item=../../../Config.sys"*

DIRECTORY TRAVERSAL - LÖSUNG

- keine sensitiven Daten an öffentlichen Orten ablegen
- Zugriffsrechte auf Ordner absichern
- Pfade als Eingabe müssen überprüft werden

SQL-INJECTION

SQL-INJECTION

```
1 var username = "foo@mail.com'; --"
2 var password = "lala"
```

```
1 var sql = "SELECT * FROM user " +
2           "WHERE username='" + username + "' " +
3             "AND password= '" + password "';";
```

```
1 SELECT * FROM user
2   WHERE username='foo@mail.com';
3     --' AND password='lala';
```

SQL-INJECTION - VARIANTEN

```
1 var username = "lala'; DROP TABLE user;--"
2 var password = "lala"
```

```
1 var username = "lala'; UPDATE password='password' " +
2                        "WHERE username='foo@mail.com';--"
3 var password = "lala"
```

SQL-INJECTION - VORGEHEN

- ausprobieren der gängigsten Namen für
  - Datenbanken
  - Tabellen
  - Spalten
- Fehlermeldungen liefern wichtige Informationen

SQL-INJECTION - LÖSUNG

- Silver Bullet: Prepared Statements

```
1 var sql = "SELECT * FROM user " +
2           "WHERE username=:username" +
3             "AND password=:password";
4
5 em.createNativeQuery(sql)
6     .setParameter("username", username)
7     .setParameter("password", password)
8     .getSingleResult();
```

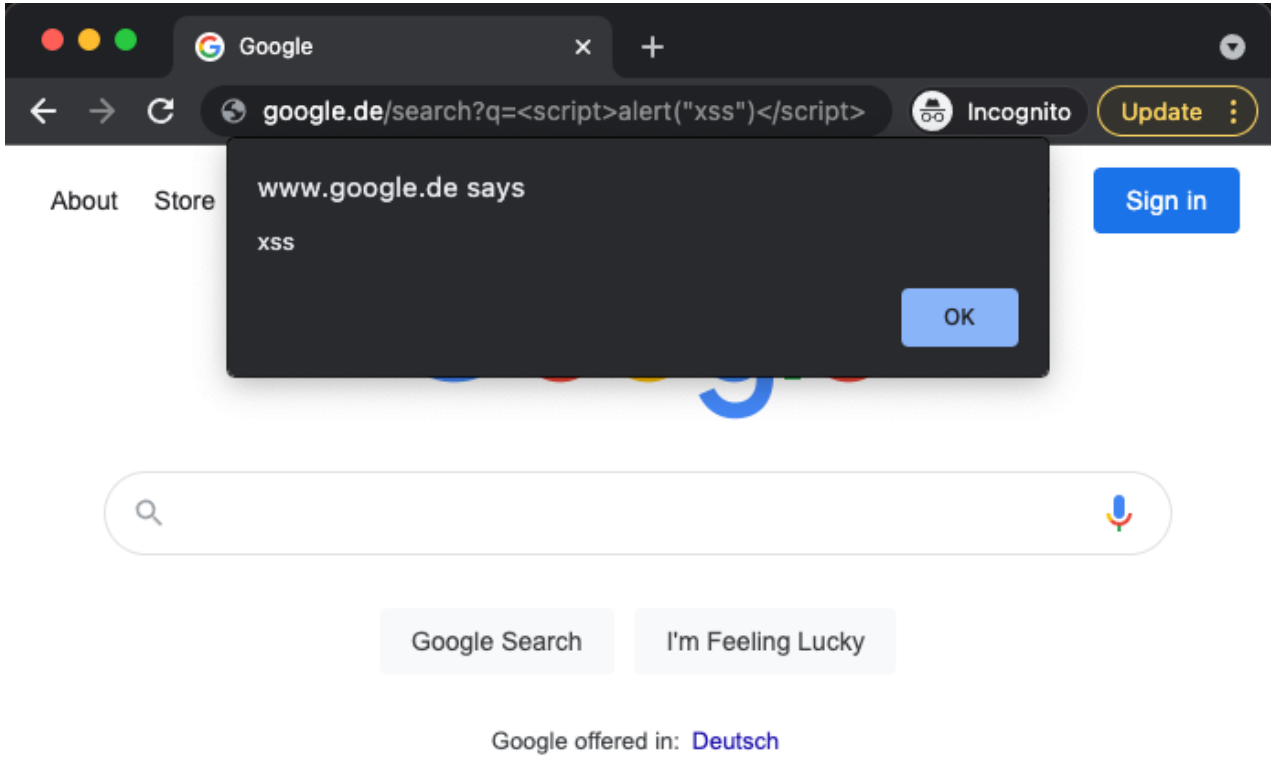
SESSION HIJACKING

- klauen der Session eines Nutzers
- raten der Session ID
- ausspähen der Session ID
- aussperren des Nutzers durch Passwortänderung

SESSION HIJACKING - LÖSUNG

- binden der Session ID an die IP-Adresse oder Browser
- größere Session ID wählen
- Passwort ändern verhindern
  - altes Passwort erneut eingeben

CROSS-SITE-SCRIPTING

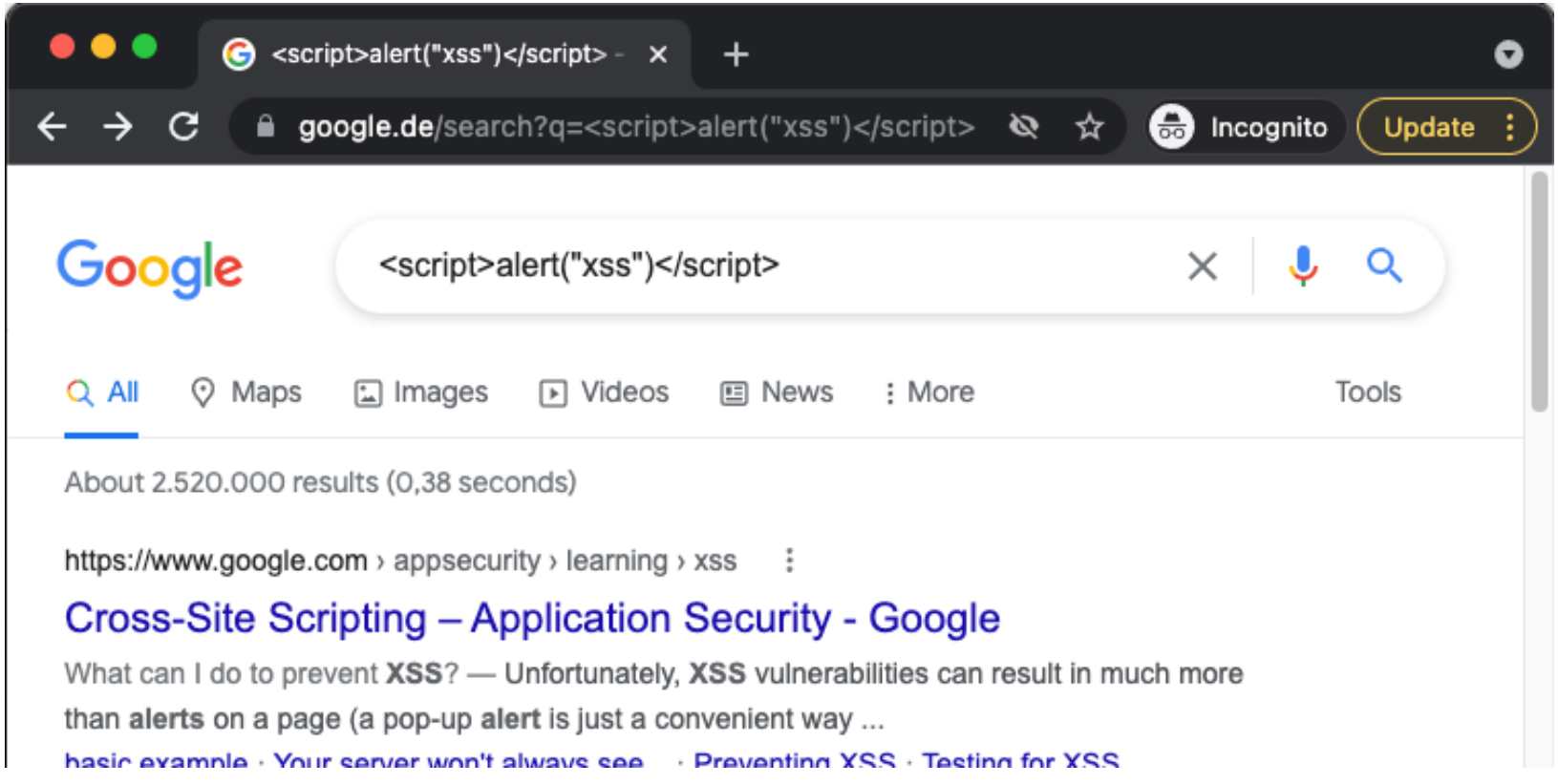


CROSS-SITE-SCRIPTING

- einfügen von JavaScript Code in Webseiten
- Ausführung des Codes durch den Browser
- Möglicher Schaden
  - Verwirrung des Nutzers
  - Weiterleitung auf andere Webseiten
  - auslesen und wegschicken von Daten

CROSS-SITE-SCRIPTING - LÖSUNG

- Encoding von Inhalten die Angezeigt werden
- Angular bringt das von Haus aus mit

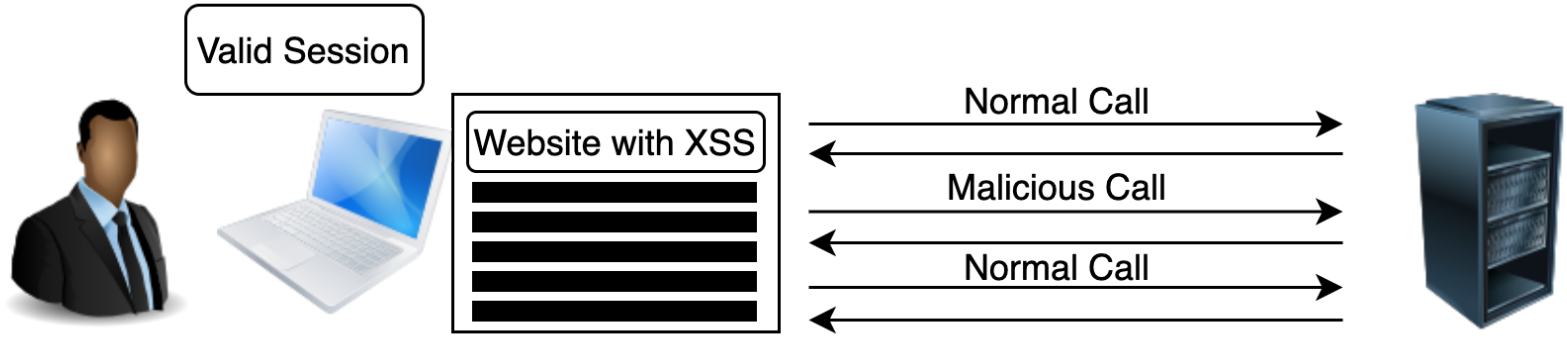


CROSS-SITE-SCRIPTING - LÖSUNG

- Content Security Policy
  - schränkt das Abrufen von Scripts ein
  - schränkt den JavaScript Code ein der ausgeführt wird
    - eval("some-code")

CROSS-SITE-REQUEST-FORGERY

- Mischung aus XSS und Session Hijacking
- Ausnutzung der Session durch XSS
- Script löst im Hintergrund Transaktionen aus



CROSS-SITE-REQUEST-FORGERY - LÖSUNG

- siehe XSS
- Webanwendung und Server teilen ein Secret
  - Secret wird bei jedem Request mitgeschickt
  - Secret kann im Cookie oder im Header liegen

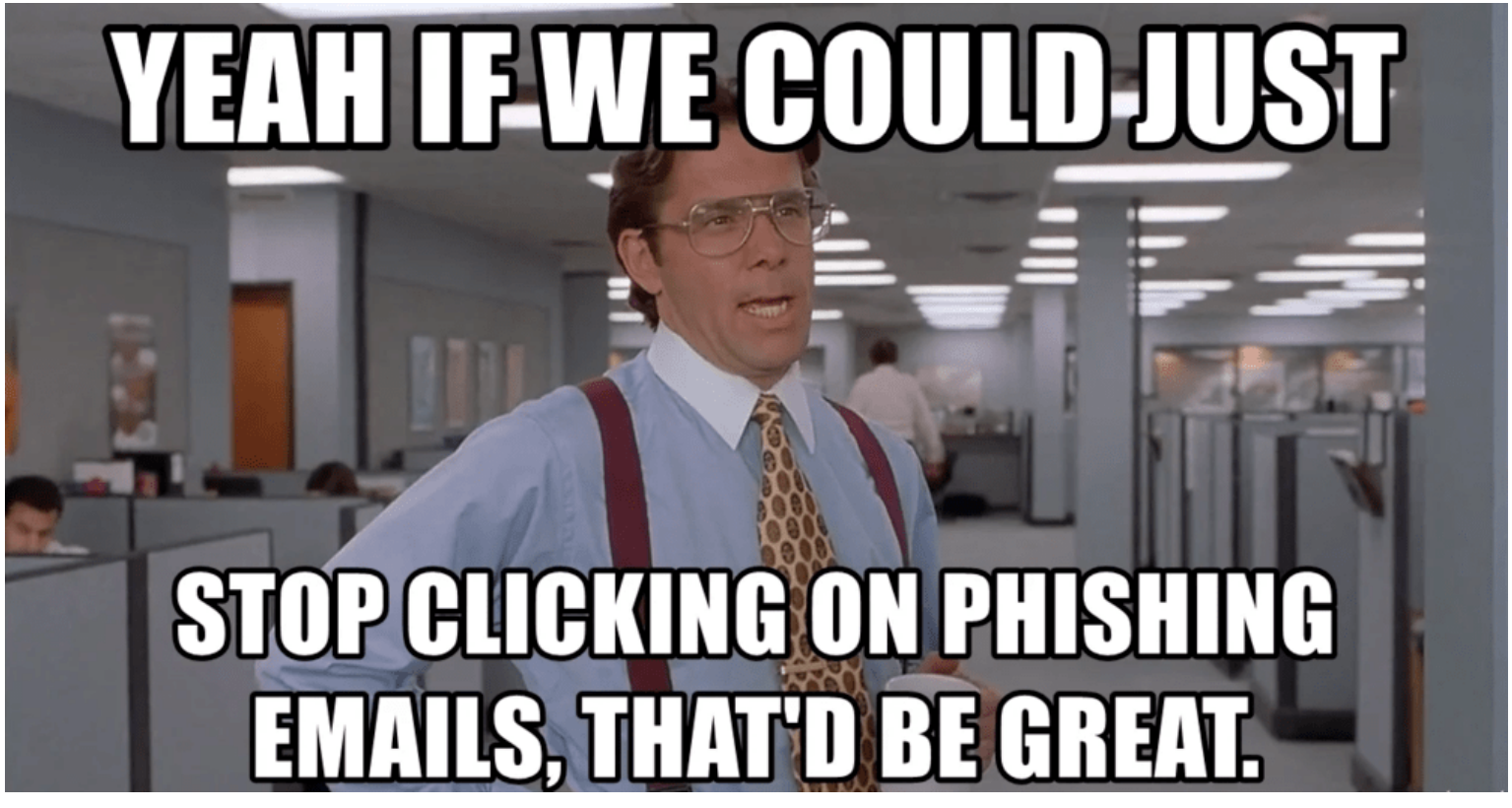
MAN-IN-THE-MIDDLE (MAN-IN-THE-BROWSER)

- Angreifer schaltet sich zwischen Nutzer und Betreiber
- zum Beispiel durch einen Fake Webauftritt
- bei Man-In-The-Browser wird kein Fake Webauftritt gebraucht
  - der Angreifer manipuliert hier direkt den Browser

MAN-IN-THE-MIDDLE - LÖSUNG

- HTTPS verwenden (ordentliche Verschlüsselung)
- zwei Faktor Authentifizierung (2FA)

PHISHING


YEAH IF WE COULD JUST
STOP CLICKING ON PHISHING EMAILS, THAT'D BE GREAT.

PHISHING

- erlangen von persönlichen Daten durchs Vertrauenserschleichung
- Angreifer gibt sich z.B. als Webseiten Betreiber aus
- basiert auf Social Engineering

PHISHING - LÖSUNG

- 2FA hilft gegen Identitätsdiebstahl
- Nutzer müssen leider mitdenken

DENAIL-OF-SERVICE

- Überlastung eines Systems durch viele Anfragen
- im speziellen Fall auch Distributed-Denail-of-Service
  - hierbei werden Anfragen von vielen Rechner gestellt
  - oft durch Botnetze

DENAIL-OF-SERVICE - LÖSUNG

- wird im Idealfall vom Server Provider verhinder
- Muster der Angriffe erkennen und auf diese nicht reagieren

ABWEHRMASSNAHMEN

DATA MINING

- aus Daten auf neue Daten schließen
- so ergeben sich aus wenig Daten viele Informationen
- Empfehlung: Daniel Kriesel "Spiegel Mining"

ALLGEMEINE ABWEHRMASSNAHMEN

- Serverseitig Daten
  - enkodieren
  - validieren
  - nicht interpretieren
- Verschlüsselung verwenden
  - HTTPS
- Wichtige Transaktionen zusätzlich absichern
  - 2FA
- Whitelisting besser als Blacklisting

ALLGEMEINE ABWEHRMASSNAHMEN

- Minimalitätsprinzip
- Sicherheitsmaßnahmen nicht ausschalten/umgehen
  - Content Security Policy
  - Cross-Origin Resource Sharing
- aktualisieren von Abhängigkeiten
  - Sonar - Dependency Check
  - Jenkins - Dependency Upgrade
- Weiterbildung/Fortbildung

PRAXIS

- einbrechen in eine Beispielanwendung
  - http://91.132.146.156:8000/login.php
  - User: bee
  - Password: bug
- wer es lokal aufsetzen möchte
  - dhbw_webengineering_2/security_bWAPP
  - wird mit *"docker-compose up -d"* gestartet
  - docker wird benötigt