Service Engineering

Technische Aspekte für (Web-) Services – Security, GraphQL

Die Inhalte der Vorlesung wurden primär auf Basis der angegebenen Literatur erstellt.

Darüber hinaus finden sich vielfältige Beispiele aus dem industriellen Umfeld.

Agenda

- Secure Web-APIs
- GraphQL
- Alternative Kurzübungen

Secure Web-APIs

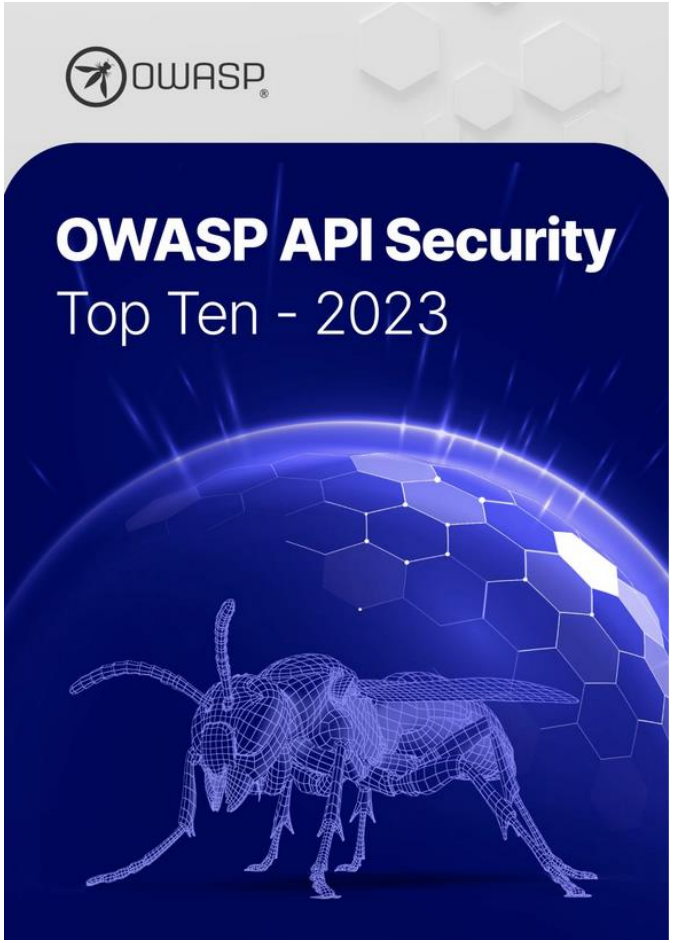
Motivation für Secure Web-APIs

Die Gartner Group geht davon aus, das im Jahr 2021 mehr als 60% aller Anwendungsentwicklungen unter Einsatz von Web-APIs erfolgen. Dies unterstreicht den Bedarf sich mit sicheren, vertrauenswürdigen und gesetzeskonformen Web-APIs auseinanderzusetzen!

Unter Berücksichtigung: Zumerle, D. et al. 2019. API Security. What You Need to Do to Protect Your APIs [online]. Verfügbar unter https://www.gartner.com/en/documents/3956746/api-security-what-you-need-to-do-to-protect-your-apis

OWASP API Security Top 10

- API1:2023 Broken Object Level Authorization
- API2:2023 Broken Authentication
- API3:2023 Broken Object Property Level Authorization
- API4:2023 Unrestricted Resource Consumption
- API5:2023 Broken Function Level Authorization
- API6:2023 Unrestricted Access to Sensitive Business Flows
- …



Quelle der rechten Abbildung: https://owasp.org/www-project-api-security, Abruf: 10. Februar 2025

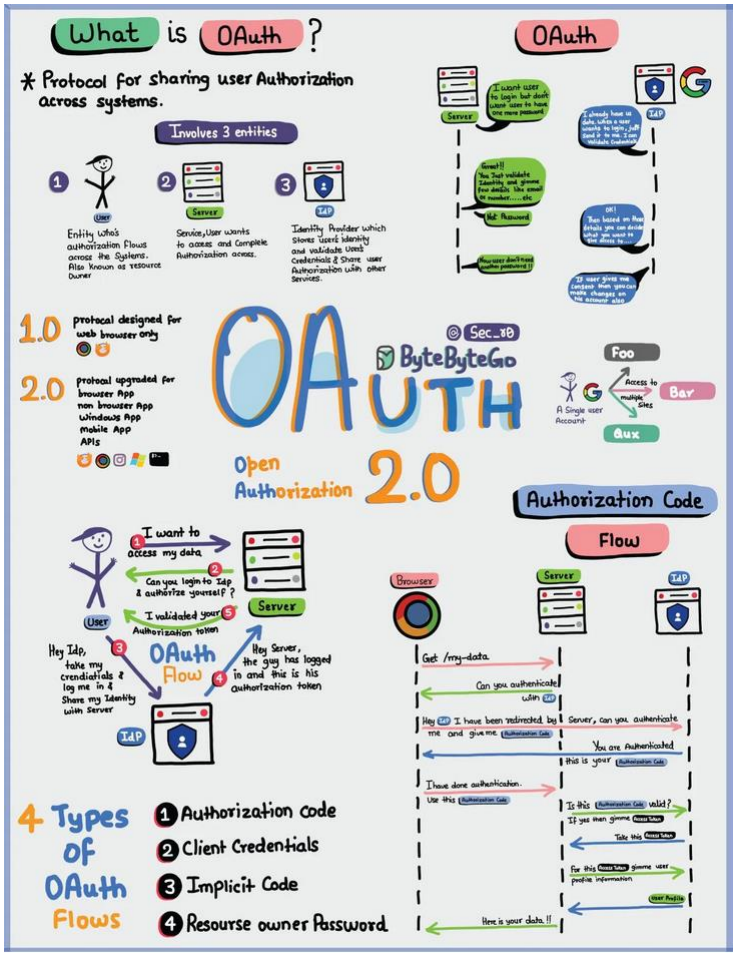
Pragmatische Industriesicht

- Einsatz von Tokens
  - Vertrauenswürdige Identitäten
  - Zugriffskontrolle auf Services und Ressourcen
- Einsatz von Verschlüsselung und Signaturen
  - Verschlüsselung mit Hilfe von TLS (Transport Layer Security)
  - Nur autorisierte Nutzer können Daten entschlüsseln und modifizieren.
- Identifizierung von Schwachstellen
  - Einsatz aktueller Versionen für Hard- und Software
  - Sniffer-Lösungen zur Erkennung von Sicherheitsproblemen
  - Berücksichtigung von „Zero-Day-Exploits“
- Einsatz von Quotas und Throttling
  - Überwachung der Aufrufquantität
  - Regeln zur Erkennung von Spikes und Denial-of-Service-Attacken
- Einsatz eines API Gateways

In Anlehnung an: https://www.redhat.com/de/topics/security/api-security

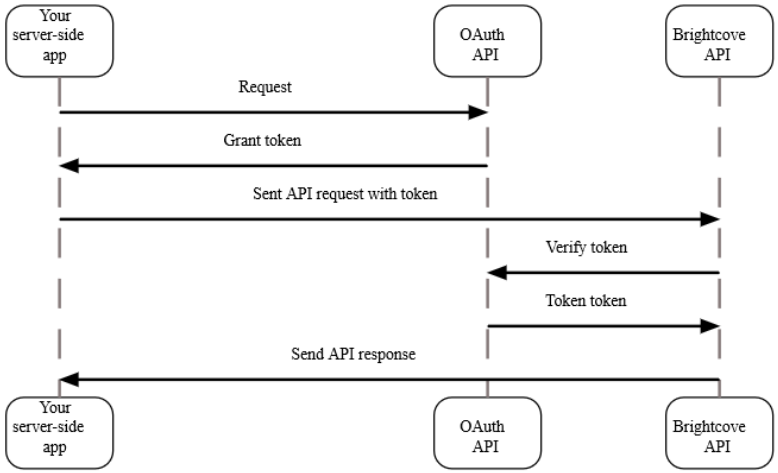
Open Authorization – OAuth 2.0

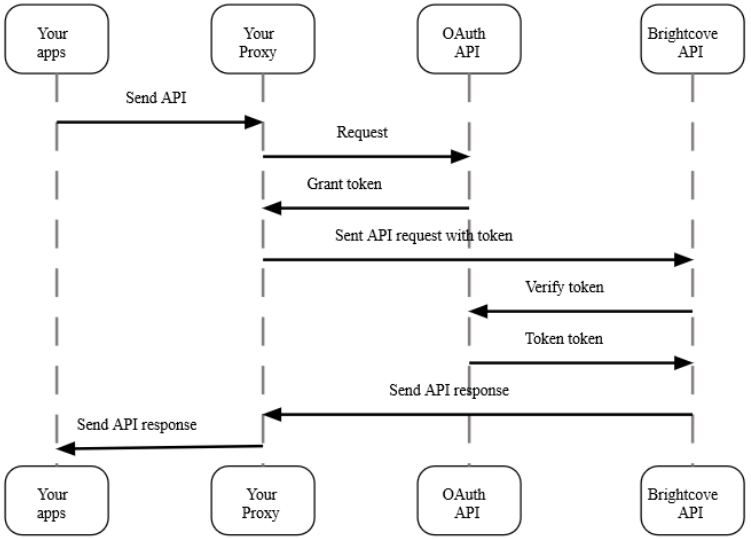
- OAuth 2.0 – Autorisierungsprotokoll
- Ressourcenzugriff – z.B. API
- Einsatz von Zugriffstokens wie z.B.:
  - JSON Web Token (JWT)
  - Bearer Token
  - …
- Verschiedene Rollen
  - Ressourcenbesitzer
  - Client
  - Autorisierungsserver
  - Ressourcenserver
- Grant-Typen (benötigte Schritte für die Autorisierung eines Ressourcenzugriffs)



Quelle der rechten Abbildung: https://blog.bytebytego.com/p/ep72-oauth-20-explained-with-simple, Abruf: 10. Februar 2025

OAuth 2.0 Interkaktionsprinzip





Quelle: https://support.brightcove.com/de/overview-oauth-api-v4#Overview,
Abruf: 30. März 2020

Literaturhinweis

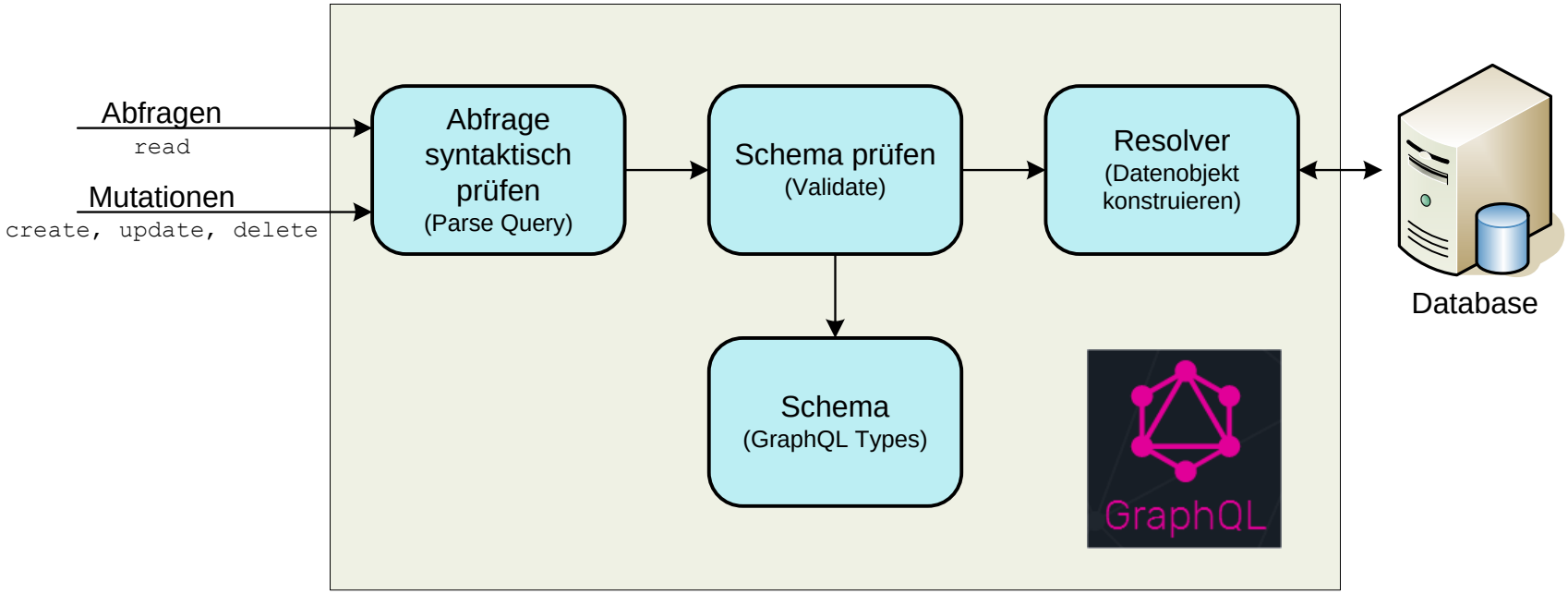
- Grundlagen - Begriffe und Stakeholder.
- Analyse existierender Arbeiten zur API-Sicherheit.
- Sicherheit und Compliance im API-Management (Prozesse/Aufgaben).
- Qualitätssicherung – Secure Web-APIs
  - Konstruktiv
  - Analytisch
  - Betrieblich
- Innovative Lösungsansätze
  - DLT/Blockchain
  - Risiko- und Reifebewertungen
  - KI-basierte Ansätze

GraphQL

Motivation und Ziele GraphQL

- Ursprünglich in 2012 von Facebook veröffentlicht.
- Seit 2015 als Open Source bereitgestellt → https://graphql.org
- Adressierung des Problem des „Overfetching“.
- Adressierung des Problem feingranularer Ressourcenzugriffe.
- Abfragesprache (vgl. SQL) für webbasierte API (meist HTTP).
  - Alternative/Ergänzung zu REST-basierten Web-APIs
  - Alternative/Ergänzung zu XML-basierten Web Services.
- GraphQL Schema als Beschreibung der API

GraphQL Laufzeitarchitektur



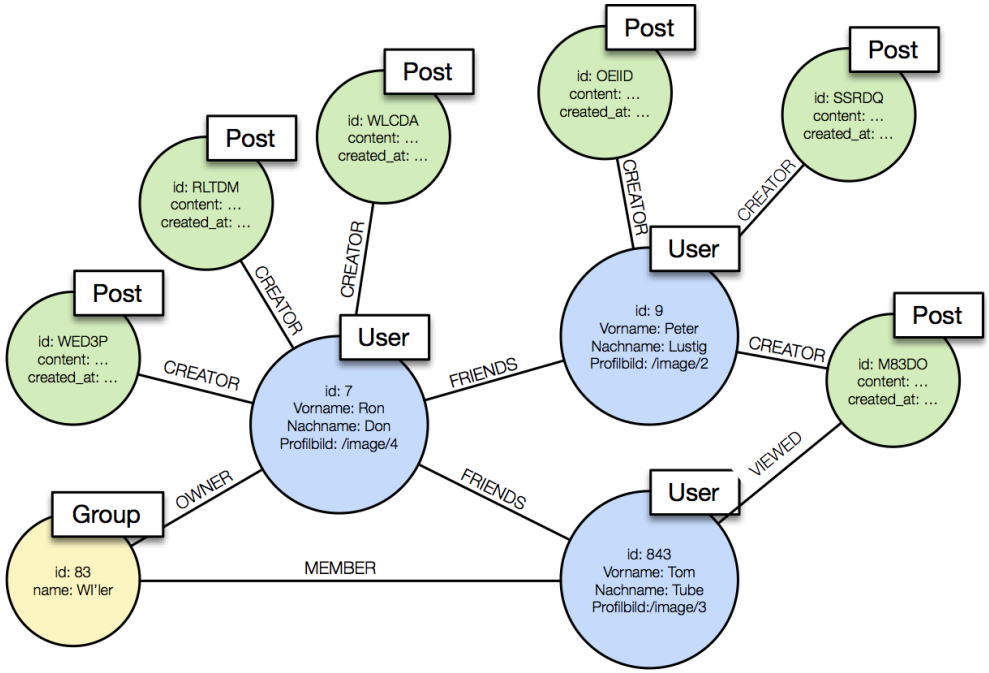
In Anlehnung: redhat: Was ist GraphQL, https://www.redhat.com/de/topics/api/what-is-graphql, Abruf: Mai 2020
Quelle des GraphQL-Symbol: https://graphql.org, Abruf: Mai 2020

GraphQL Type-System (Schema)

```
type User {
  id: Int
  vorname: String
  nachname: String
  friends: [User]
  groups: [Group]
}
```

```
type Post {
  id: Int
  content: String
  creator: User
}
```

```
type Group {
  id: Int
  name: String
  member: [User]
}
```



Quelle: Böhme, L.; Grunert, J.: Einführung und Implementierung von GraphQL als API-Spezifikation, in Proc. Schmietendorf, A.; Nitze, A. (Hrsg.): API-First/API-Management Workshop ECC 2020, Shaker 2020

GraphQL Beispielabfrage

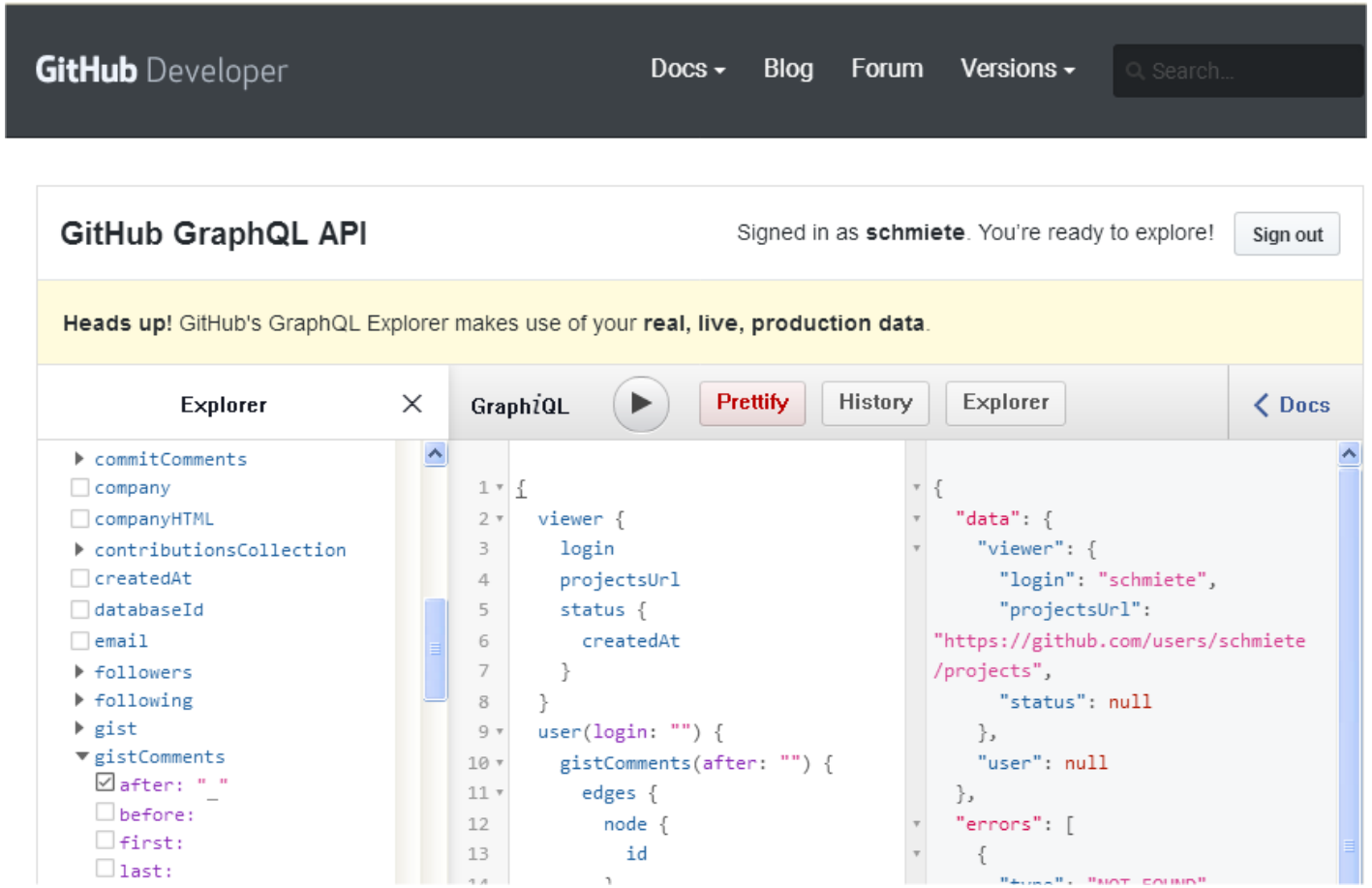
- Die Query wird durch die Verschachtelung von Type Eigenschaften erstellt.
- Das Beispiel zeigt eine Abfrage mit GraphQL, um den letzten Post vom User mit der ID 9 abzurufen.

```
query getPosts {
 user(id: 9){
  id
  vorname
  nachname
  posts(last: 1){
   id
   content
  }
 }
}
```

```
{
  data: {
    "id": 9,
    "vorname": "Peter",
    "nachname": "Lustig",
    "posts": [
      {
        "id": "aW7nD",
        "content": "ECC 2018"
      }
    ]
  }
}
```

Quelle: Böhme, L.; Grunert, J.: Einführung und Implementierung von GraphQL als API-Spezifikation, in Proc. Schmietendorf, A.; Nitze, A. (Hrsg.): API-First/API-Management Workshop ECC 2020, Shaker 2020

GraphQL Explorer unter GitHub



Quelle der Abb.: https://developer.github.com/v4/explorer, letzter Abruf: Mai 2020

Quellen zum Thema

- Kirschner, A.; Pointner, G.: Was der neue API-Ansatz leistet und was nicht - Einführung in GraphQL, https://jaxenter.de/einfuehrung-in-graphql-71048
- Klassen, D.; Augsten, S.: Eine andere Art APIs zu implementieren - GraphQL als Alternative zu REST, https://www.dev-insider.de/graphql-als-alternative-zu-rest-a-752074
- Springer, S.: GraphQL Server auf Basis von React entwickeln, https://entwickler.de/online/development/graphql-server-react-579858102.html

Kurzübung Themenkomplex 3

Kurzübung 3
(Alternative: Secure Web-APIs)

- Was wird unter dem Begtriff “Zero-Day-Exploits” verstanden
- Machen Sie sich mit den “OWASP Top 10 API Security” Risiken vertraut
  - Gehen Sie für 2 Sicherheitslücken auf entwicklerseitige oder betriebliche Möglichkeiten zur Vermeidung bzw. Abmilderung ein.
  - Gehen Sie für 2 Sicherheitslücken auf die Möglichkeiten von Tests zur Aufdeckung potentieller Schwachstellen ein.
- Analysieren Sie die grundlegende Arbeitsweise von OAuth2.
  - Welche grundlegenden Rollen werden im OAuth2 unterschieden?
  - Welche Daten werden mittels Token bereitgestellt, wozu wird dieser genutzt?
  - Welche Zusammenhänge bestehen zwischen OAut2 und OpenID?

Kurzübung 3
(Alternative: GraphQL)

- Machen Sie sich mit den grundlegende Ideen und Prinzipien des GraphQL-Ansatzes vertraut.
- Welche Vor- und Nachteile sehen Sie beim Einsatz von GraphQL im Vergleich zu ausschließlich REST-basierten Web-APIs?
- Wie kann eine Serveranwendung um eine GraphQL-Schnittstelle erweitert werden? Gehen Sie auf die grundlegende Architektur und auf mindestens 2 Implementierungsalternativen (z.B. Framework Apollo) sowie die Schritte zur Entwicklung ein.
- Testen Sie eine fertig angebotene GraphQL-Schnittstelle (z.B. mit Hilfe des GitHub-GraphQL Explorers). → optionale Aufgabe!