Technische Universität Berlin

# Advanced Access Control in GraphQL APIs

Group 4: Wilke Klausing, Huy Viet Nguyen

Technische Universität Berlin

# Structure

1. Recap
2. Plugin Logic
    - 2.1. Purpose Tree
    - 2.2. Rule Creation
3. Showcase
4. Benchmarks
5. Conclusion

Technische Universität Berlin

# 1. Recap: Task

Create a re-usable component for at least one widely used framework/stack for developing GraphQL Web APIs allowing developers to easily add access control to their APIs

Technische Universität Berlin

# 1. Recap: Planning

Access Control Plugin for Apollo Server with following characteristics:

- Configuration:
  - No database, only two configuration files:
  - Purpose tree as configuration file
  - Rules as configuration file

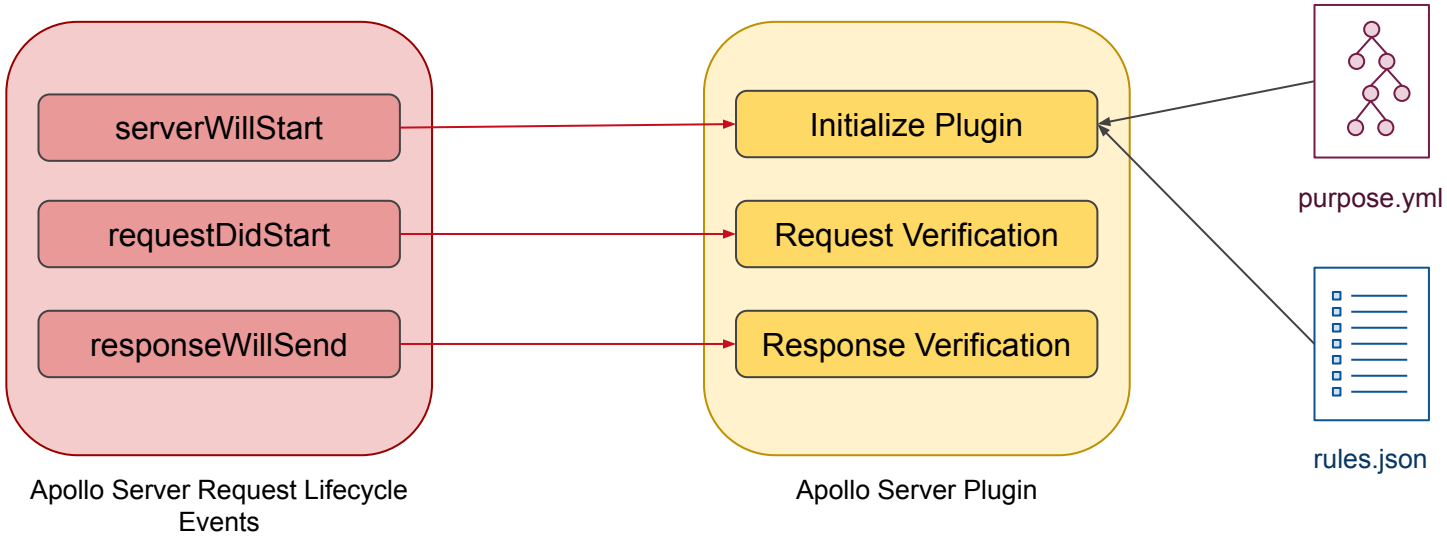
Technische Universität Berlin

# 1. Recap: Planning

- Access Control Models:
  - Purpose-based Access Control
  - Attribute-based Access Control
- Goal:
  - Flexible
  - Reuseable
  - Developer Friendly

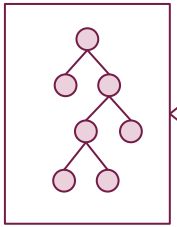
Technische Universität Berlin

# 2. Plugin Logic



Technische Universität Berlin

# 2.1. Purpose Tree



purpose.yml

```
purpose:
  research:
    - sleep research
    - cardiovascular research
  marketing:
    - personalized marketing
    - influencer marketing
  health:
    - descriptive analytics:
      - descriptive sleep analytics
      - descriptive cardiovascular analytics
    - diagnostic analytics:
      - diagnostic sleep analytics
      - diagnostic cardiovascular analytics
    - predictive analytics:
      - predictive sleep analytics
      - predictive cardiovascular analytics
    - prescriptive analytics:
      - prescriptive sleep analytics
      - prescriptive cardiovascular analytics
  fitness:
    - track activity
    - social fitness:
      - group challenge
```

Technische Universität Berlin

# 2.2. Rule Creation (Header Rule)

rules.json

**Header Rule Example**

```
{
    "field": "First_Name",
    "category": "header",
    "compare": "user-agent",
    "operation": "contains",
    "value": "Chrome",
    "policy": "deny",
    "error": "emptystring"
}
```

Technische Universität Berlin

# 2.2. Rule Creation (Purpose Rule)

rules.json

**Purpose Rule Example**

```
{
    "field": "TotalSteps",
    "category": "purpose",
    "purpose": "health",
    "exception": "sleep analytics",
    "policy": "allow",
    "error": "delete"
}
```

Technische Universität Berlin

# Showcase

Technische Universität Berlin

# Benchmark

- Two GCloud VMs used:
  - Location: Germany, US
  - Type: e2-medium (2 vCPUs, 4 GB memory)
- Benchmark test:
  - 1000 Requests to the Apollo Server
  - Response with and without AC Plugin are identical

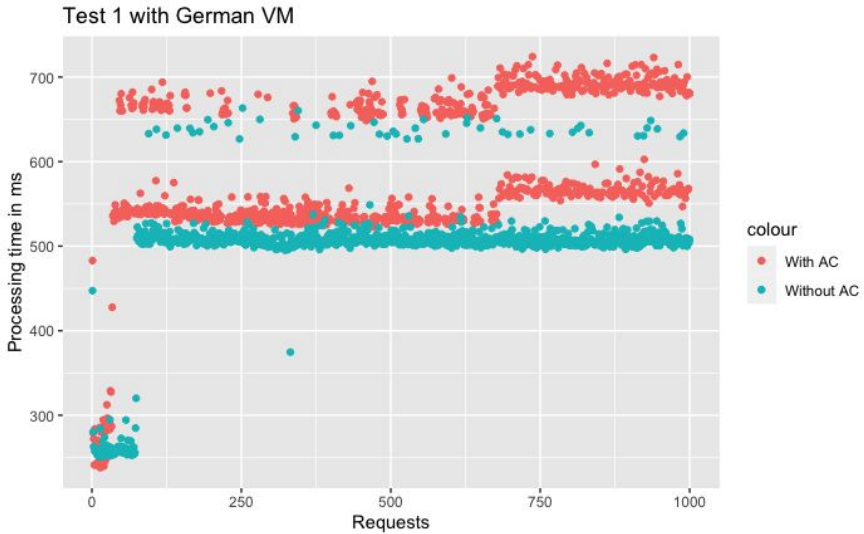
Technische Universität Berlin

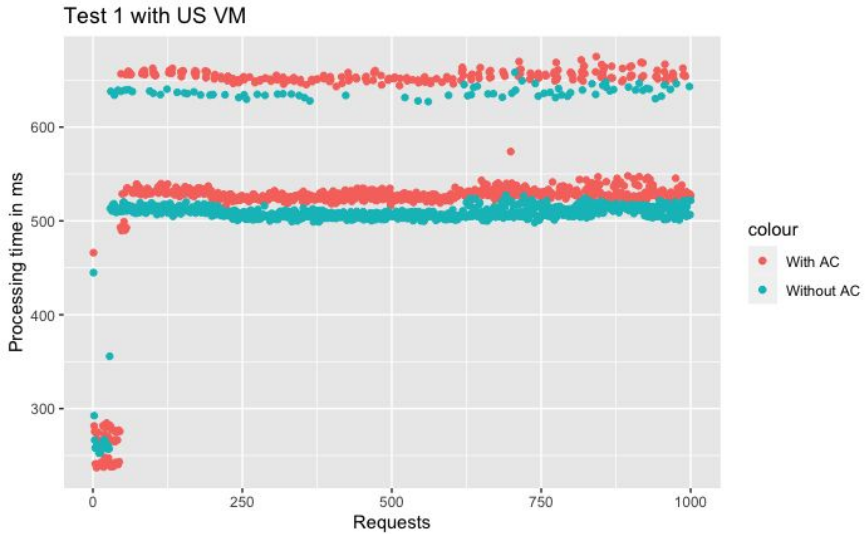
# Benchmark

- Test 1: BURJ (101943 key/values)
- Test 2: GRANDE (833 key/values)
- Test 3: TALL (473 key/values)

Technische Universität Berlin

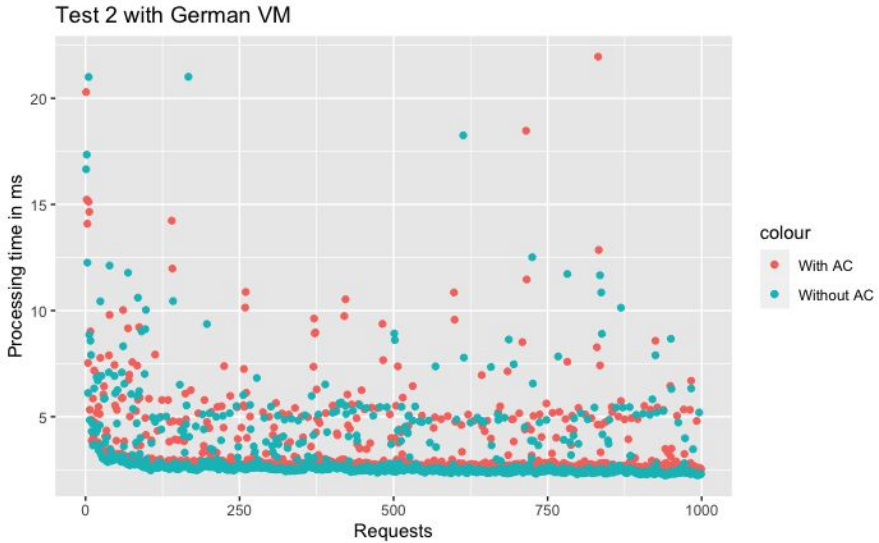
# Benchmark

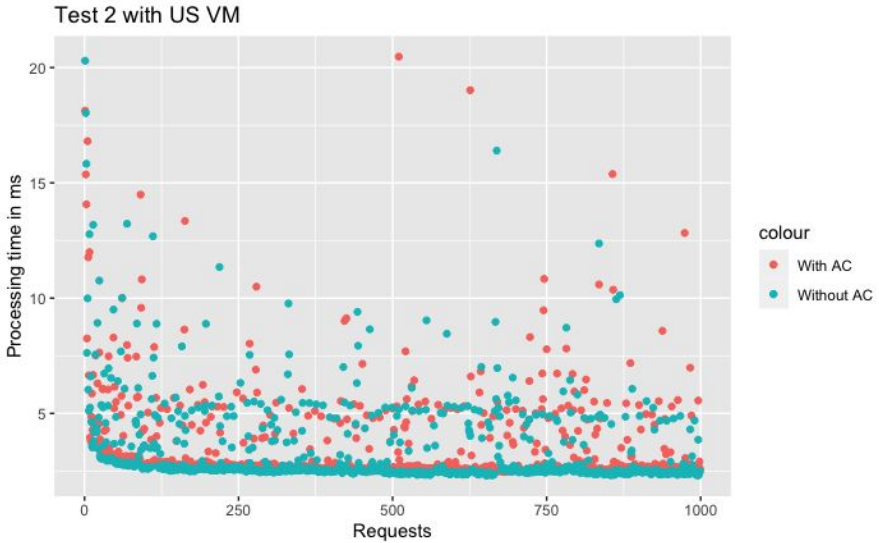


Technische Universität Berlin

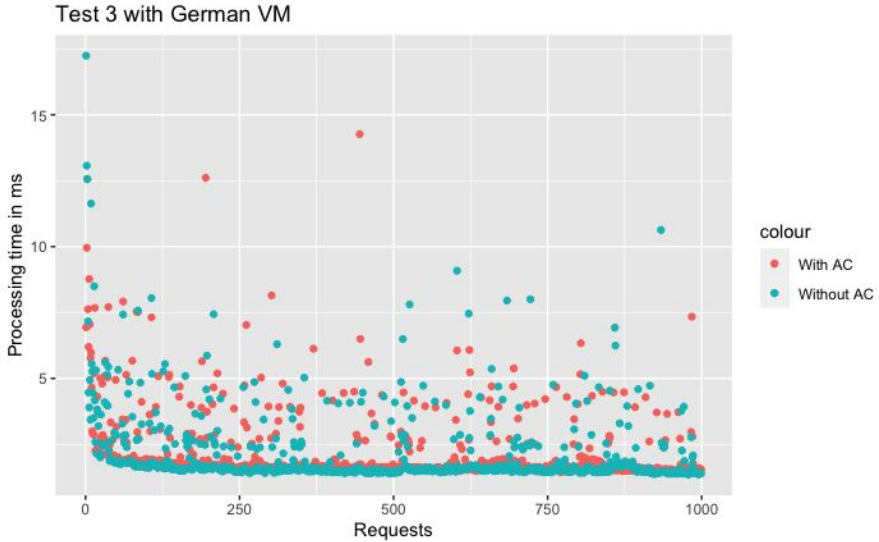
# Benchmark

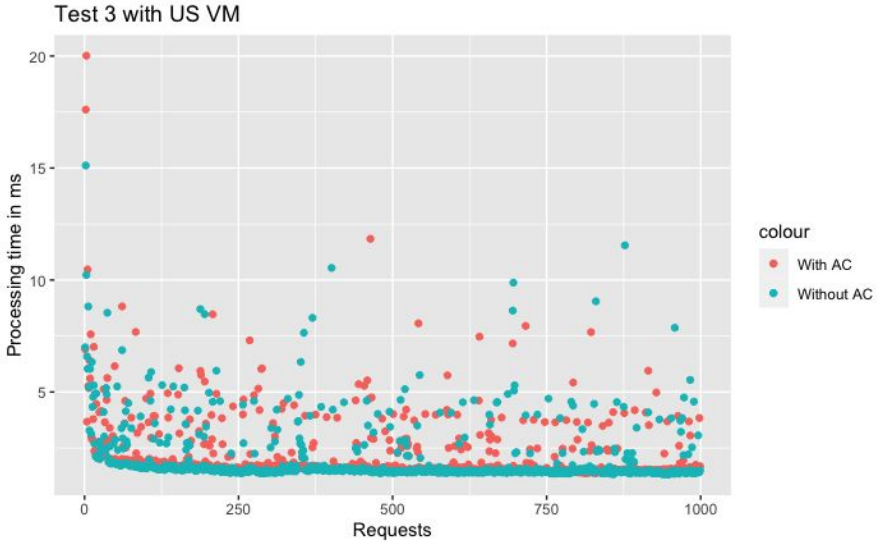


Technische Universität Berlin

# Benchmark



Technische Universität Berlin

# Conclusion

- Results of Germany VMs are more unstable:
  - Test1: 16,78% (86,67ms)
  - Test2: 8,87% (0,2876ms)
  - Test3: 4,74% (0,0973ms)
- Results of US VMs show following overhead:
  - Test1: 6,18% (31,64ms)
  - Test2: 6,34% (14,02ms)
  - Test3: 5,20% (0,1046ms)

Technische Universität Berlin

# Conclusion

- Overhead of 5-10% seems plausible, needs further testing
- Goals accomplished:
    - Flexible: Configuration files
    - Reuseable: Plugin can be added to any Apollo Server
    - Developer Friendly: Configuration written in standard formats

Technische Universität Berlin

# Thank you for your attention.