oneconsult®

OWASP API Security Top 10

How APIs are Hacked and How to Develop Securely

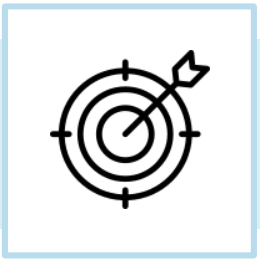
Frank Ully, Senior Penetration Tester & Security Consultant

23 August 2020

AGENDA

Introduction to OWASP API Security Top 10

Risks



Summary

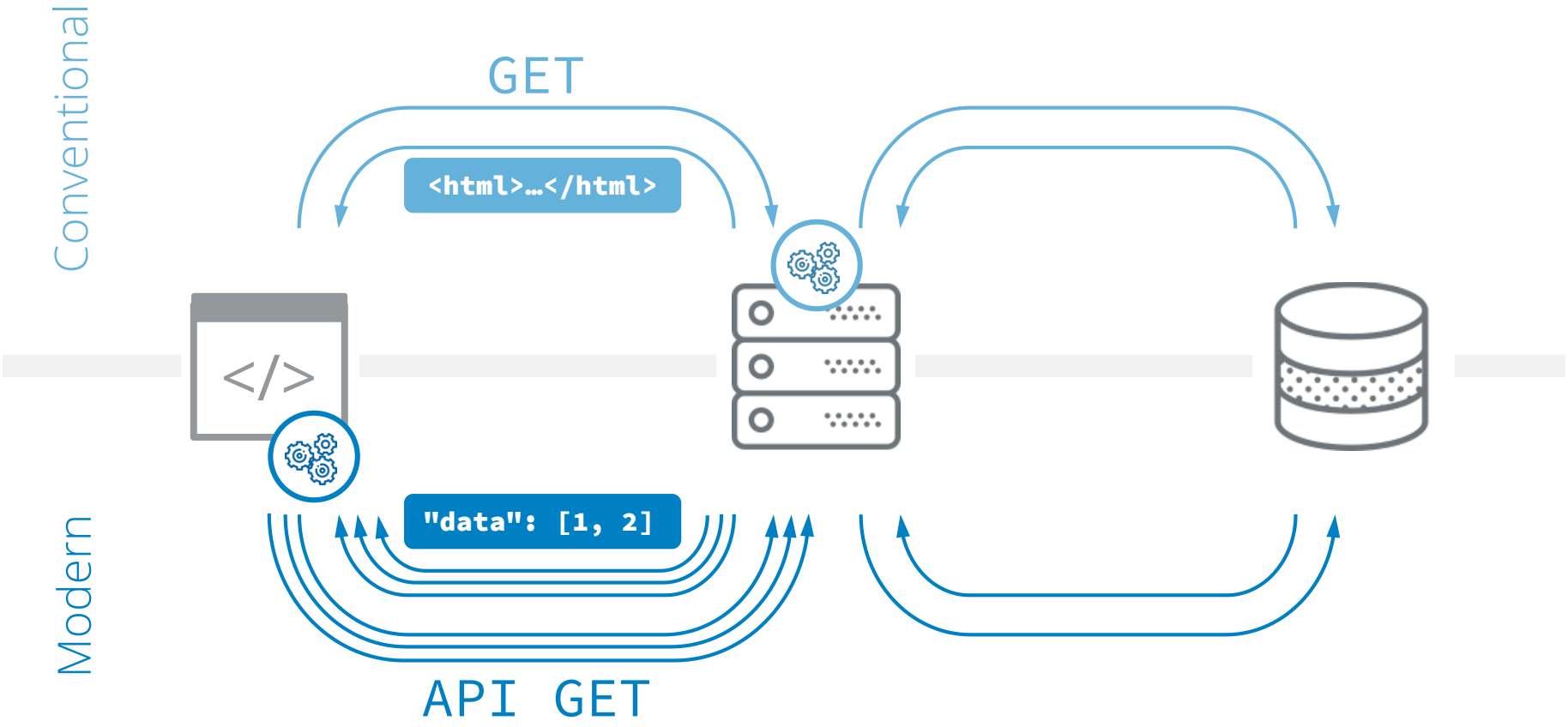
Resources

oneconsult®

Introduction to OWASP API Security Top 10

OVERVIEW OF THE OWASP TOP 10 RISKS

CONVENTIONAL VS. MODERN



oneconsult®

API1:2019 Broken Object Level Authorization

OVERVIEW

Description

- Privileges of authenticated users are insufficiently checked at object level
- Attackers change the ID sent in a request to that of another user (or one of his records)
- Attackers gain unauthorized access to data of other customers

Risk Assessment

Exploitability **3**

Prevalence **3**

Detectability **2**

Technical impact **3**

CAUSE

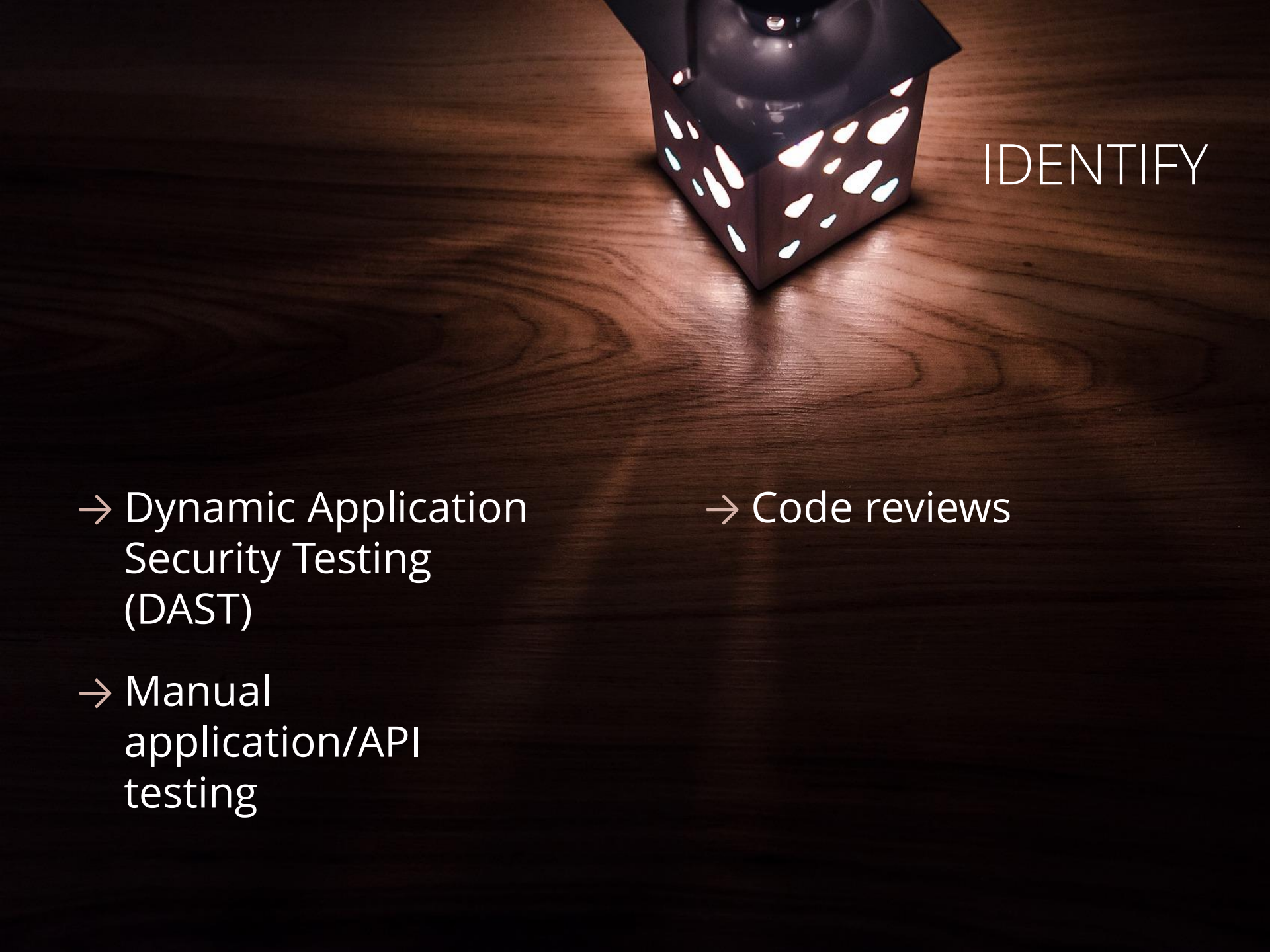
```
POST api/talk/rate
{ "talk_id":1234,
 "rate": 5 }
```

```
Talk.find_by_id
(1234).update
```

```
UPDATE talks
WHERE id=1234
```

RESULT

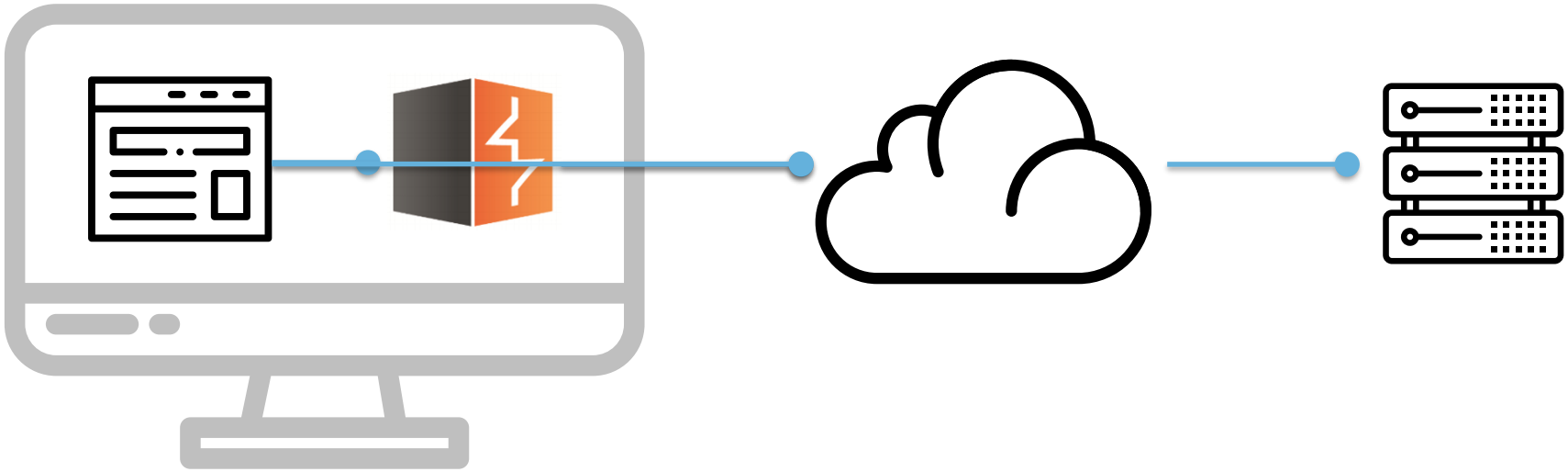
- Data of other users is exposed
- Data of other tenants is exposed
- Horizontal privilege escalation
- Possible vertical privilege escalation



IDENTIFY

- → Dynamic Application Security Testing (DAST)
- → Manual application/API testing
- → Code reviews

INTERCEPTION PROXY



MEASURES

#123

```
GET
/talk/444
```

object controller

#123 is allowed to see #444?

#456

```
PUT
/user_profile
{"uid": 555,
 "name": "Hugo" }
```

profile controller

#456 is allowed to edit #555?

#789

```
DELETE
/admin/user_profile?
uid=777
```

admin controller

#789 is allowed to delete #777?

Decision Engine

Source: https://www.youtube.com/watch?v=wY6q583JWLc&t=420s

oneconsult®

API2:2019
Broken User Authentication

OVERVIEW

Description

- Authentication and session management is not properly implemented
- Attackers can log on unauthenticated, impersonate another user, or steal passwords or session information

Risk Assessment

Exploitability **3**

Prevalence **2**

Detectability **2**

Technical impact **3**

CAUSE

- ForgotPassword
- Login
- MobileLogin
- EditPhoto
- UpdateStatus

special protection required – missing

- lockout mechanism
- CAPTCHA
- protection against credential stuffing

misimplementation

- service does not check OAuth provider
- passwords are saved without salt
- *JSON Web Token (JWT) used improperly*

JSON WEB TOKEN: BACKGROUND INFORMATION

Header

```
eyJhbGciOiJIUzI1NiIs
InR5cCI6IkpXVCJ9.
```

```
{ "alg": "HS256",
  "typ": "JWT" }
```

Payload

```
eyJzdWIiOiIxMjM0NTY3
ODkwIiwibmFtZSI6Ikpv
aG4gRG9lIiwiYWRtaW4i
OmZhbHNlLCJpYXQiOjE1
NjE3MzIzMDcsImV4cCI6
MTU5MzM1NDcwN30.
```

```
{ "sub": "1234567890",
  "name": "John Doe",
  "admin": false,
  "iat": 1561732307,
  "exp": 1593354707 }
```

Signature

```
K6fJEWLLeMFnYwYcNv-
ZUWUoytgB38GWyCVonV
6w-A8
```

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  "secret")
```

JSON WEB TOKEN: CAUSE

Header

```
eyJhbGciOiAiTm9uZSIs
InR5cCI6ICJKV1QifQo.
```

```
{ "alg": "None",
  "typ": "JWT" }
```

Payload

```
eyJzdWIiOiIxMjM0NTY3
ODkwIiwibmFtZSI6Ikpv
aG4gRG9lIiwiYWRtaW4i
OnRydWUsImlhdCI6MTU2
MTczMjMwNywiZXhwIjox
NTkzMzU0NzA3fQ.
```

```
{ "sub": "1234567890",
  "name": "John Doe",
  "admin": true,
  "iat": 1561732307,
  "exp": 1593354707 }
```

Signature

RESULT

- JWT: privilege escalation
- JWT: counterfeit tokens
- Unauthorized access to data
- Unauthorized execution of actions

IDENTIFY

- → How are users authenticated?
- → How are passwords stored and transferred?
- → How is the user identification developed and managed?

- → JWT: dynamic
  - Manipulate signature
  - Manipulate header
- → JWT: static
  - Validate instead of decode
  - Strong secret

MEASURES

- Protect credentials and session ID
- Use standard mechanisms
- Implement password policy as per NIST 800-63
- JWT: IETF (Draft) JSON Web Token Best Current Practices
- Don‘t forget registration, recovery, etc.

oneconsult®

API3:2019
Excessive Data Exposure

OVERVIEW

Description

- API-based applications provide a lot of data that is first filtered by the client
- This data is visible when a specific request is made and/or API responses are viewed directly
- Attackers gain access to confidential information, e.g. addresses or account details

Risk Assessment

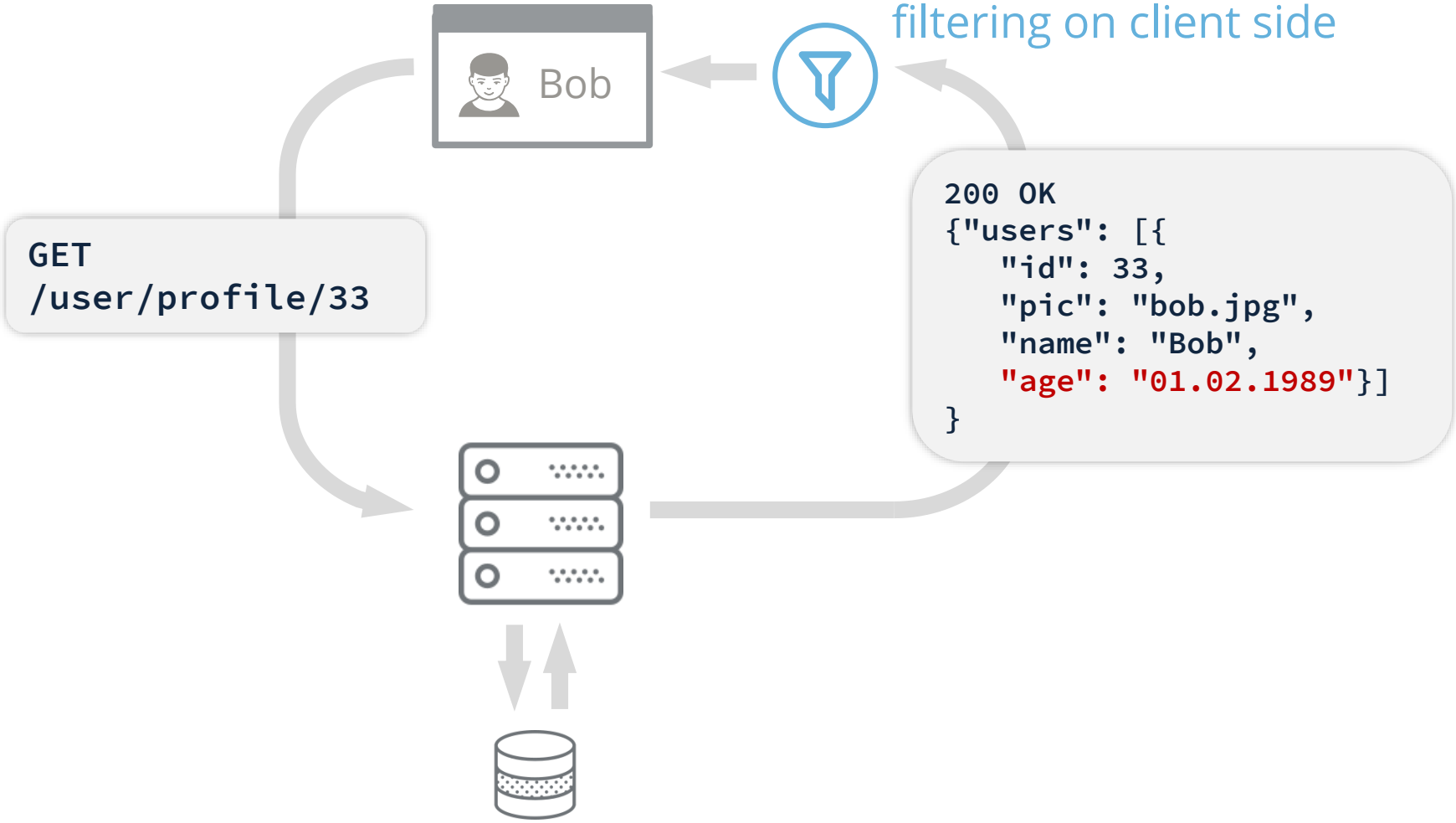
Exploitability **3**

Prevalence **2**

Detectability **2**

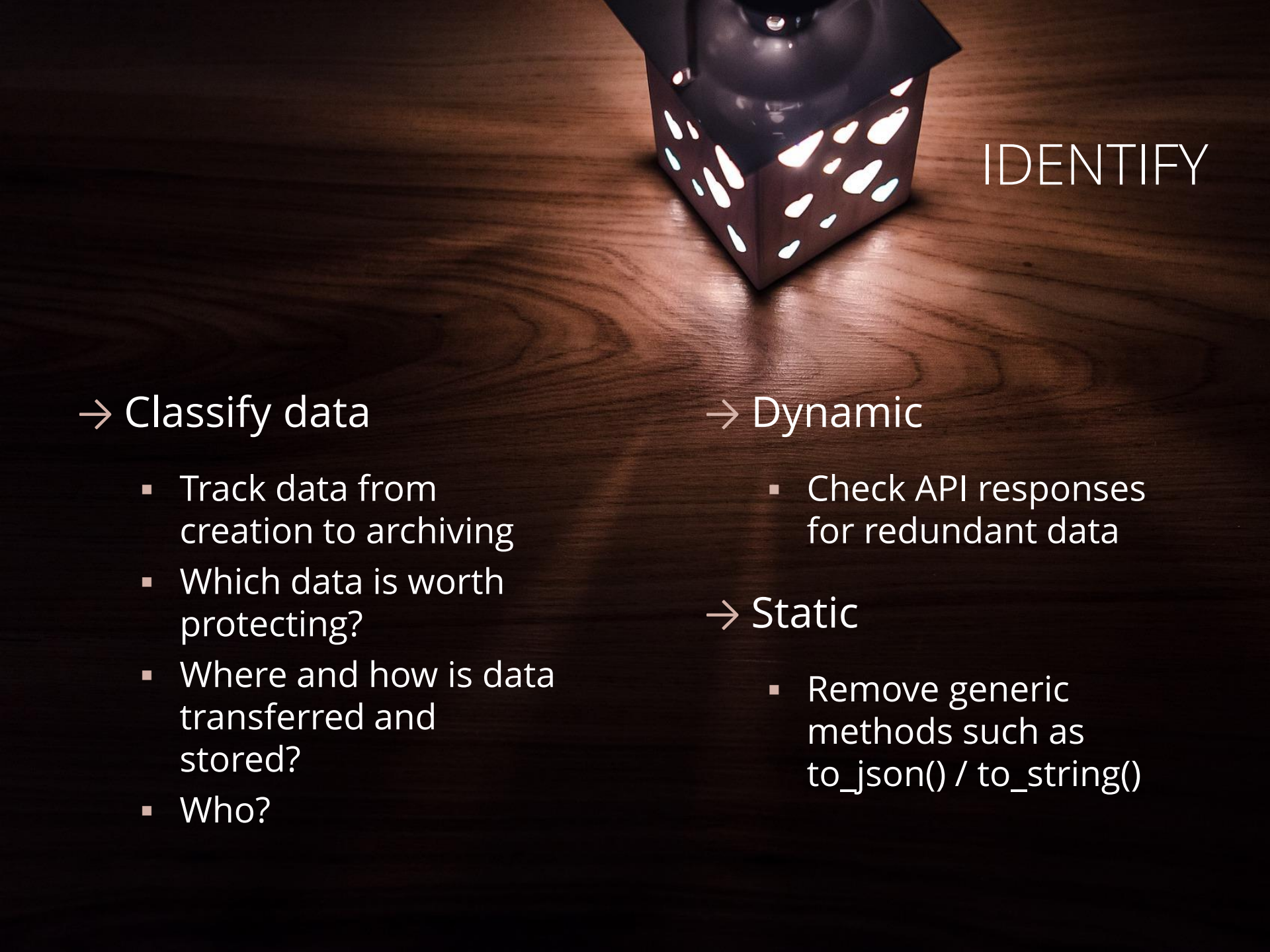
Technical impact **2**

CAUSE



RESULT

- Disclosure of sensitive data
- Financial loss
- Damage to reputation



IDENTIFY

→ Classify data

- Track data from creation to archiving
- Which data is worth protecting?
- Where and how is data transferred and stored?
- Who?

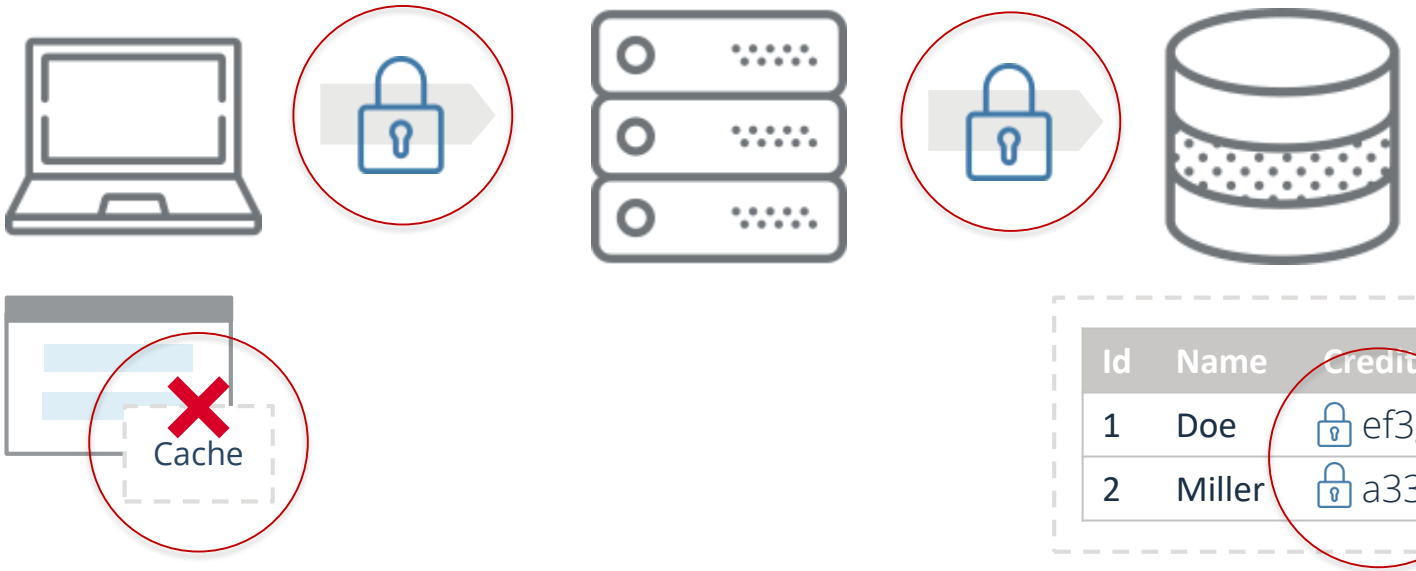
→ Dynamic

- Check API responses for redundant data

→ Static

- Remove generic methods such as to_json() / to_string()

MEASURES



oneconsult®

API4:2019 Lack of Resources and Rate Limiting

OVERVIEW

Description

- API is not protected against great number of requests or large payloads
- Through missing or only partially configured rate limiting – or easily circumventable rate limiting – attackers can execute brute-force attacks, e.g. on user accounts, up to denial-of-service attacks

Risk Assessment

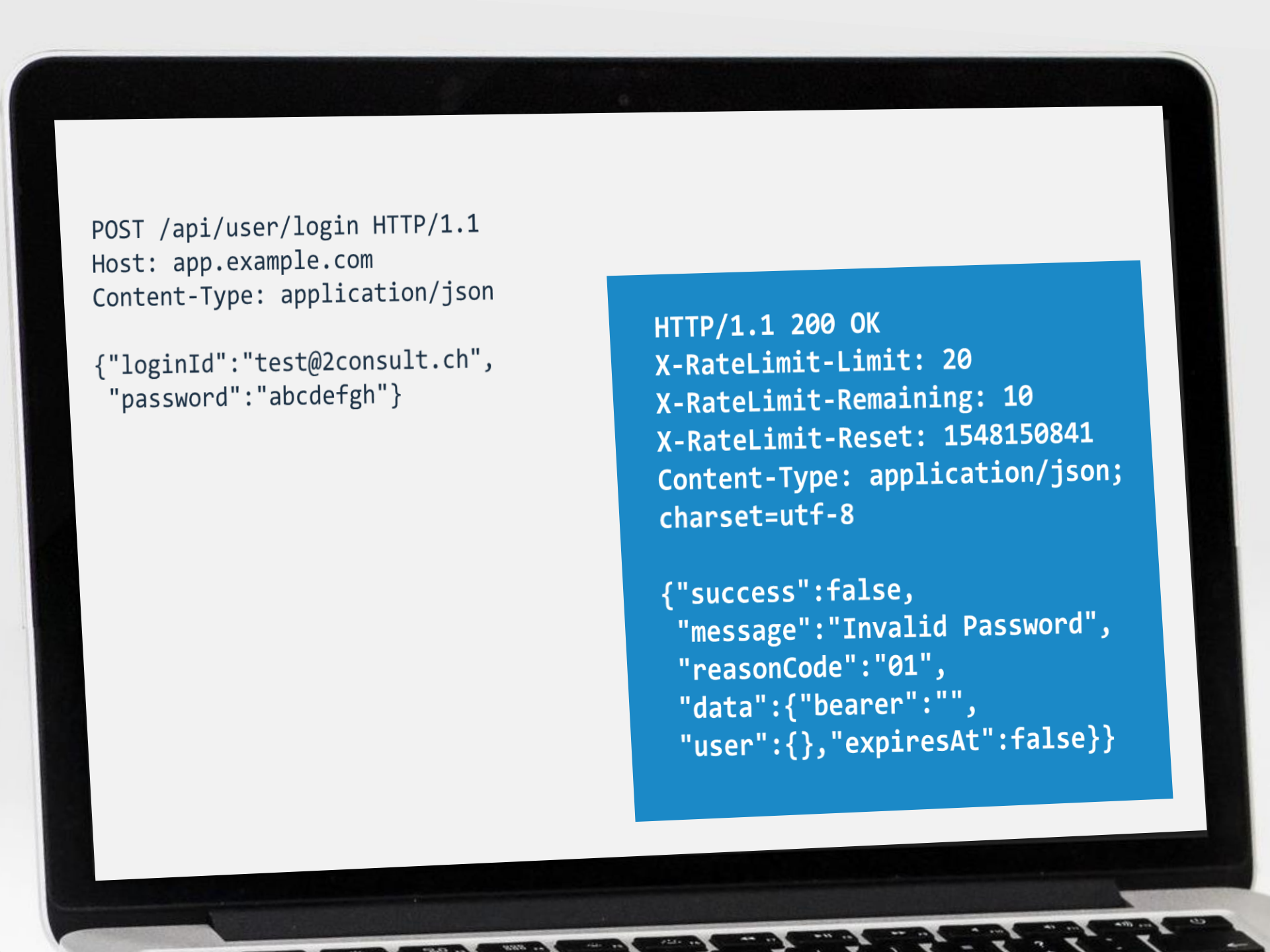
Exploitability 2

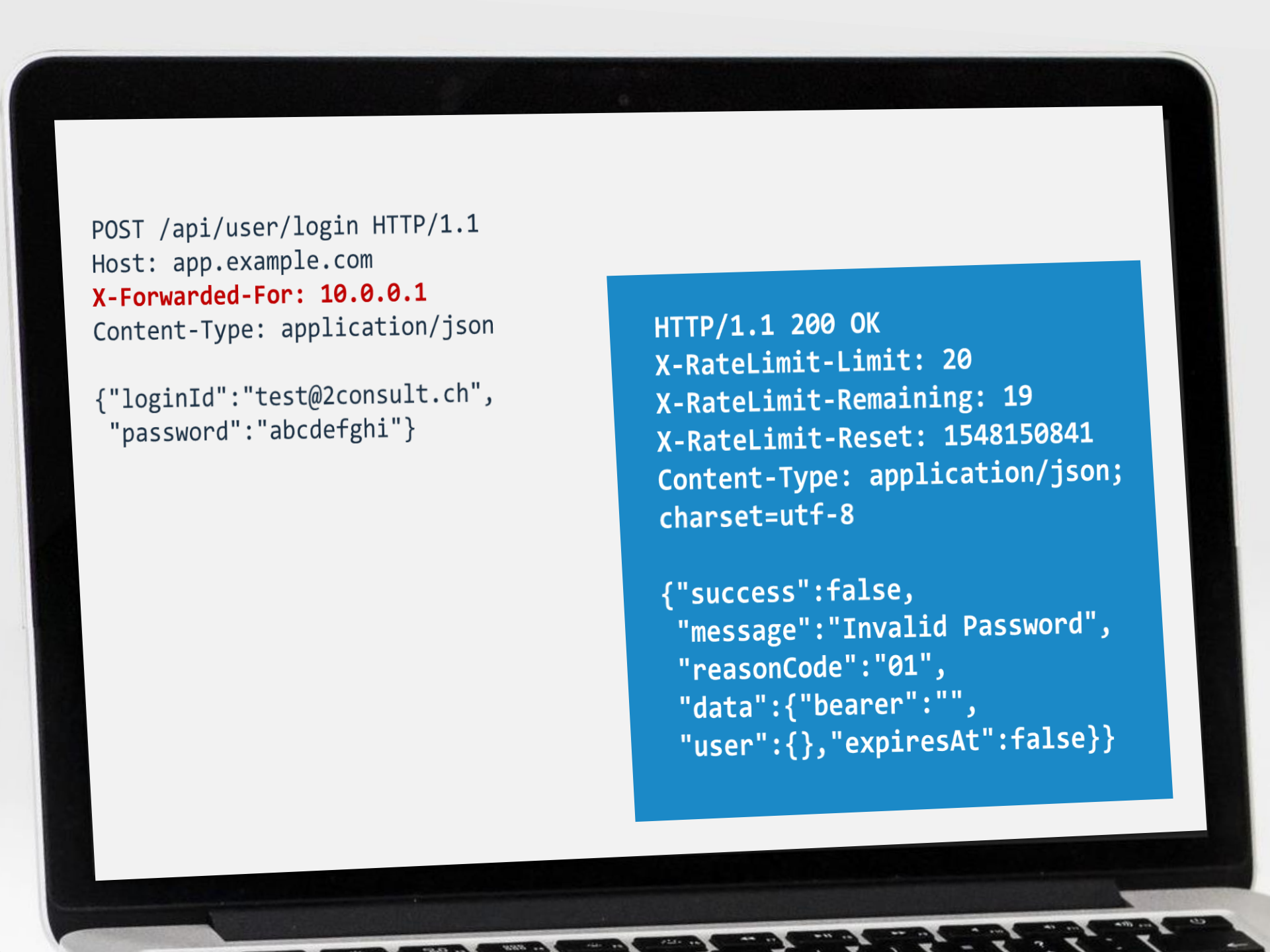
Prevalence 3

Detectability 3

Technical impact 2

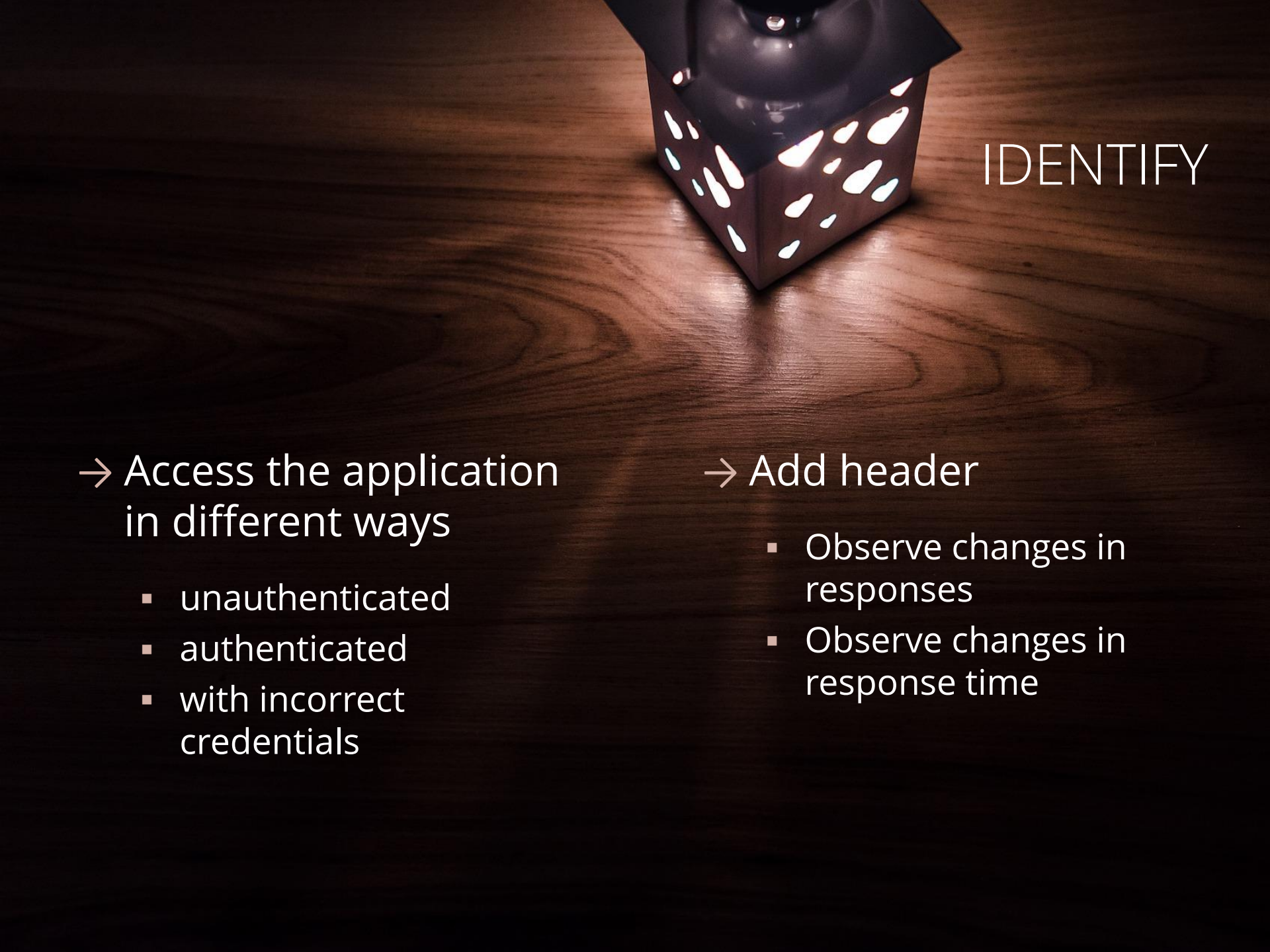

POST /api/user/login HTTP/1.1
Host: app.example.com
Content-Type: application/json
{"loginId":"test@2consult.ch",
"password":"abcdefgh"}
CAUSE


POST /api/user/login HTTP/1.1
Host: app.example.com
Content-Type: application/json
{"loginId":"test@2consult.ch",
"password":"abcdefgh"}
HTTP/1.1 200 OK
X-RateLimit-Limit: 20
X-RateLimit-Remaining: 10
X-RateLimit-Reset: 1548150841
Content-Type: application/json;
charset=utf-8
{"success":false,
"message":"Invalid Password",
"reasonCode":"01",
"data":{"bearer":"",
"user":{},"expiresAt":false}}


POST /api/user/login HTTP/1.1
Host: app.example.com
X-Forwarded-For: 10.0.0.1
Content-Type: application/json
{"loginId":"test@2consult.ch",
"password":"abcdefghi"}
HTTP/1.1 200 OK
X-RateLimit-Limit: 20
X-RateLimit-Remaining: 19
X-RateLimit-Reset: 1548150841
Content-Type: application/json;
charset=utf-8
{"success":false,
"message":"Invalid Password",
"reasonCode":"01",
"data":{"bearer":"",
"user":{},"expiresAt":false}}

RESULT

- Brute-force attacks are not prevented
- Data leakage
- Unauthorized access to data
- Denial of service



IDENTIFY

→ Access the application in different ways

- unauthenticated
- authenticated
- with incorrect credentials

→ Add header

- Observe changes in responses
- Observe changes in response time

MEASURES

- Implement rate limiting (429 Too Many Requests)
- Implement payload limit
- Never trust user input
- ~~X-Forwarded-For~~, ~~True-Client-IP~~, ~~X-Real-IP~~, ~~Referer~~, ...
- ~~X-Host~~, ~~X-Forwarded-Host~~, ~~X-Original-URL~~, ~~X-Rewrite-URL~~

oneconsult®

API5:2019 Broken Function Level Authorization

OVERVIEW

Description

- Privileges of authenticated users are insufficiently checked at function level
- Attackers modify the path or method of a request to reach a privileged API endpoint
- Attackers gain access to administrative functions

Risk Assessment

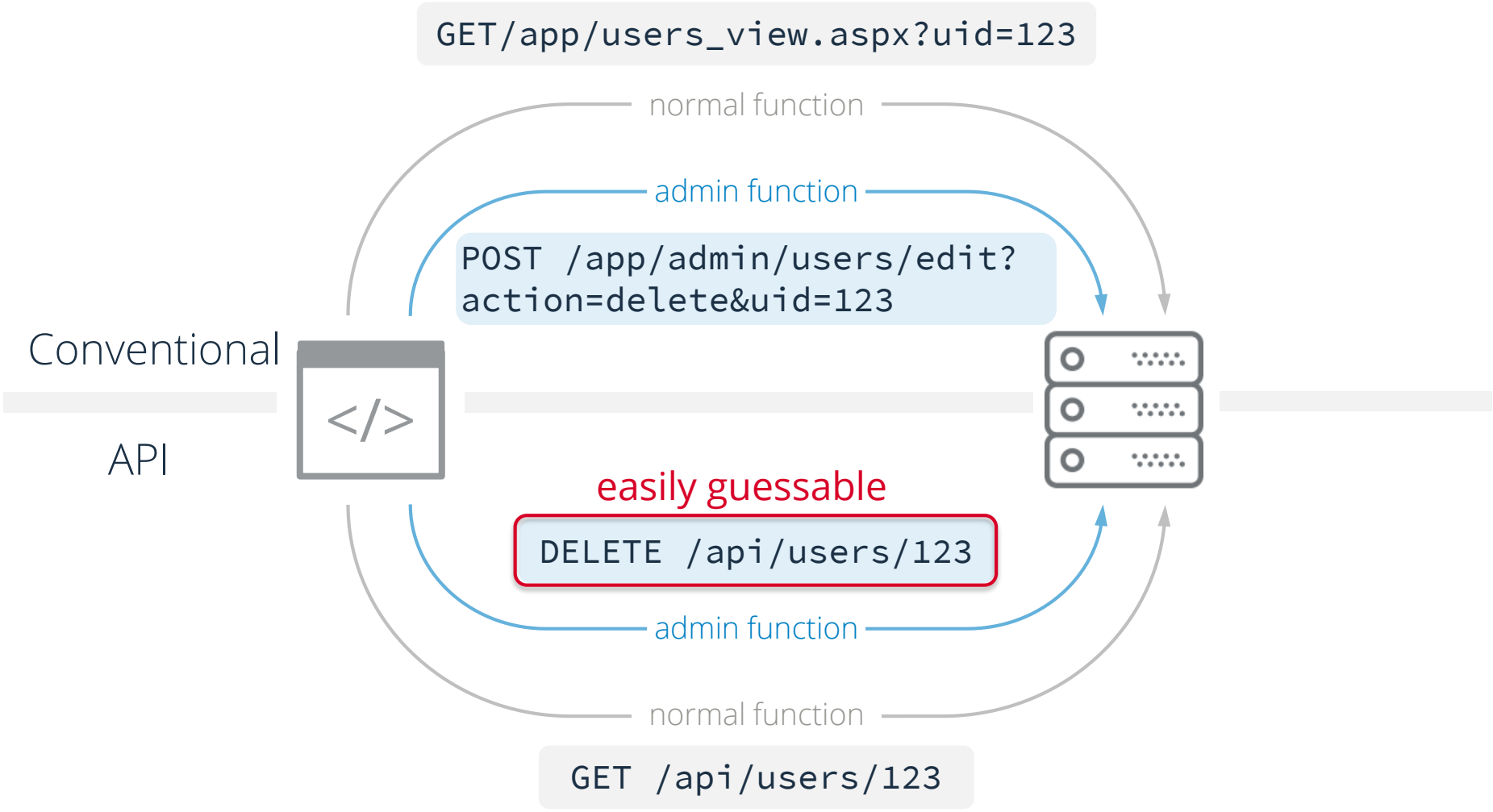
Exploitability **3**

Prevalence **2**

Detectability **1**

Technical impact **2**

IDENTIFY / CAUSE



RESULT

- Exposure of higher-privileged functions
- Vertical privilege escalation

MEASURES

Access to functions

User is authenticated

Deny by default (except for public resources)

User has required role

Access to objects

User has access authorization (direct or indirect owner)

User has access type (CRUD)

oneconsult®

API6:2019
Mass Assignment

OVERVIEW

Description

- Frameworks automate the conversion of client input into object properties
- Attackers can overwrite object properties
- Attackers can escalate privileges, manipulate data or bypass security mechanisms

Risk Assessment

Exploitability 2

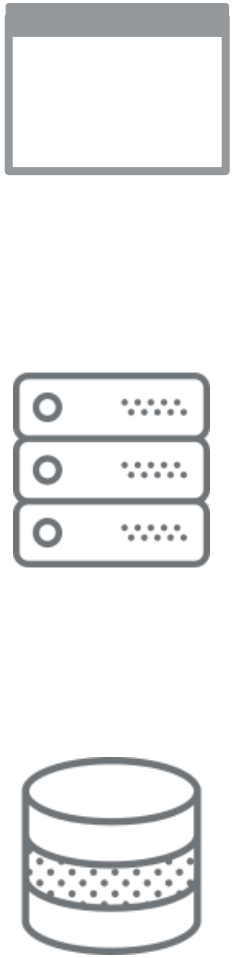
Prevalence 2

Detectability 2

Technical impact 2

CAUSE: CREATE USER – API

```
POST /api/user
{
"username":"Bob"
}
```



```
var user =
      new User(req.body);
user.save();
```

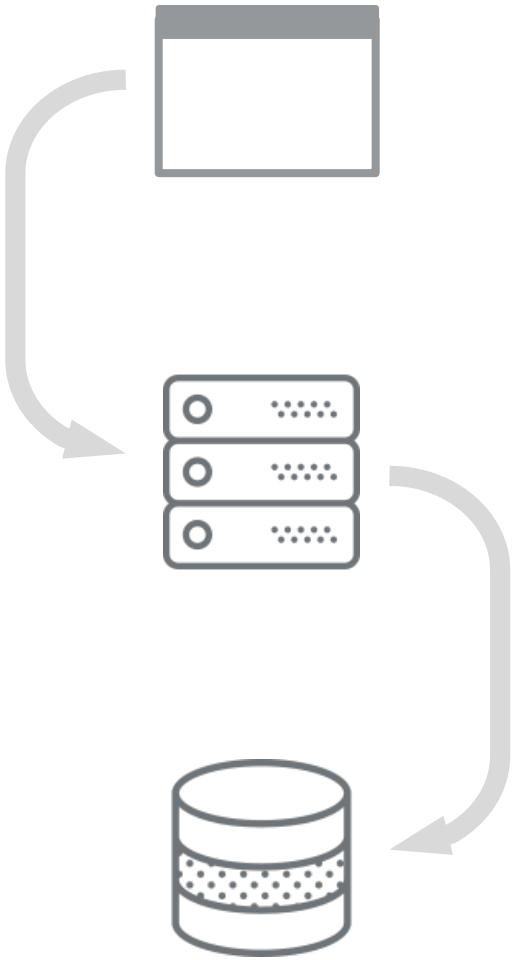
```
{JSON as
apporpriate}
```

ORM

RESULT

```
POST /api/user
{ "username":"Bob" }
```

```
POST /api/user
{ "username":"Bob",
  "role":"admin" }
```



MEASURES

- Do not automatically bind input from the client into objects
- Define and *allowlist* expected parameters / values
- Properties that should not be changed: read-only
- Define schemas, types and patterns

oneconsult®

API7:2019
Security Misconfiguration

OVERVIEW

Description

- Security is depending on the configuration;
  insecure (default) settings are used
- Attackers benefit from information about the components used; this information can simplify further attacks or even help attackers execute code on the server

Risk Assessment

Exploitability **3**

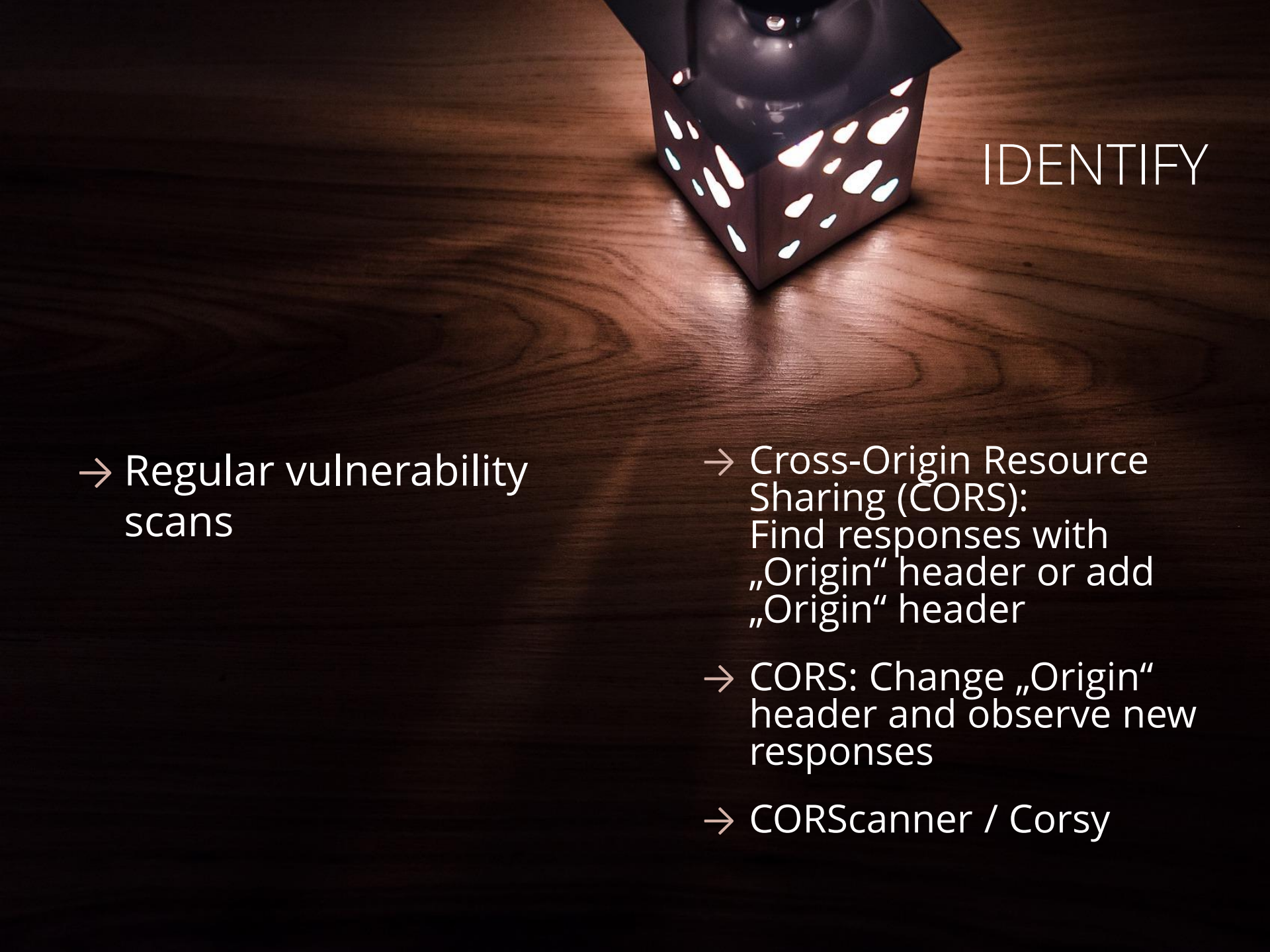
Prevalence **3**

Detectability **3**

Technical impact **2**

RESULT

- Data leakage
- Publicly known vulnerabilities
- Unauthorized access to data
- Takeover of the complete system



IDENTIFY

- → Regular vulnerability scans
- → Cross-Origin Resource Sharing (CORS): Find responses with „Origin“ header or add „Origin“ header
- → CORS: Change „Origin“ header and observe new responses
- → CORScanner / Corsy

MEASURES

- Repeatable hardening process according to good practices
- Same configuration for development, test and production environment
- Simple platform without unnecessary features
- Segmented application architecture
- Automated testing of configurations and settings across all system environments

oneconsult®

API8:2019 Injection

OVERVIEW

Description

- Interpreter does not distinguish between code and data; data interpreted as code
- Attackers can steal or modify data or take over the system completely

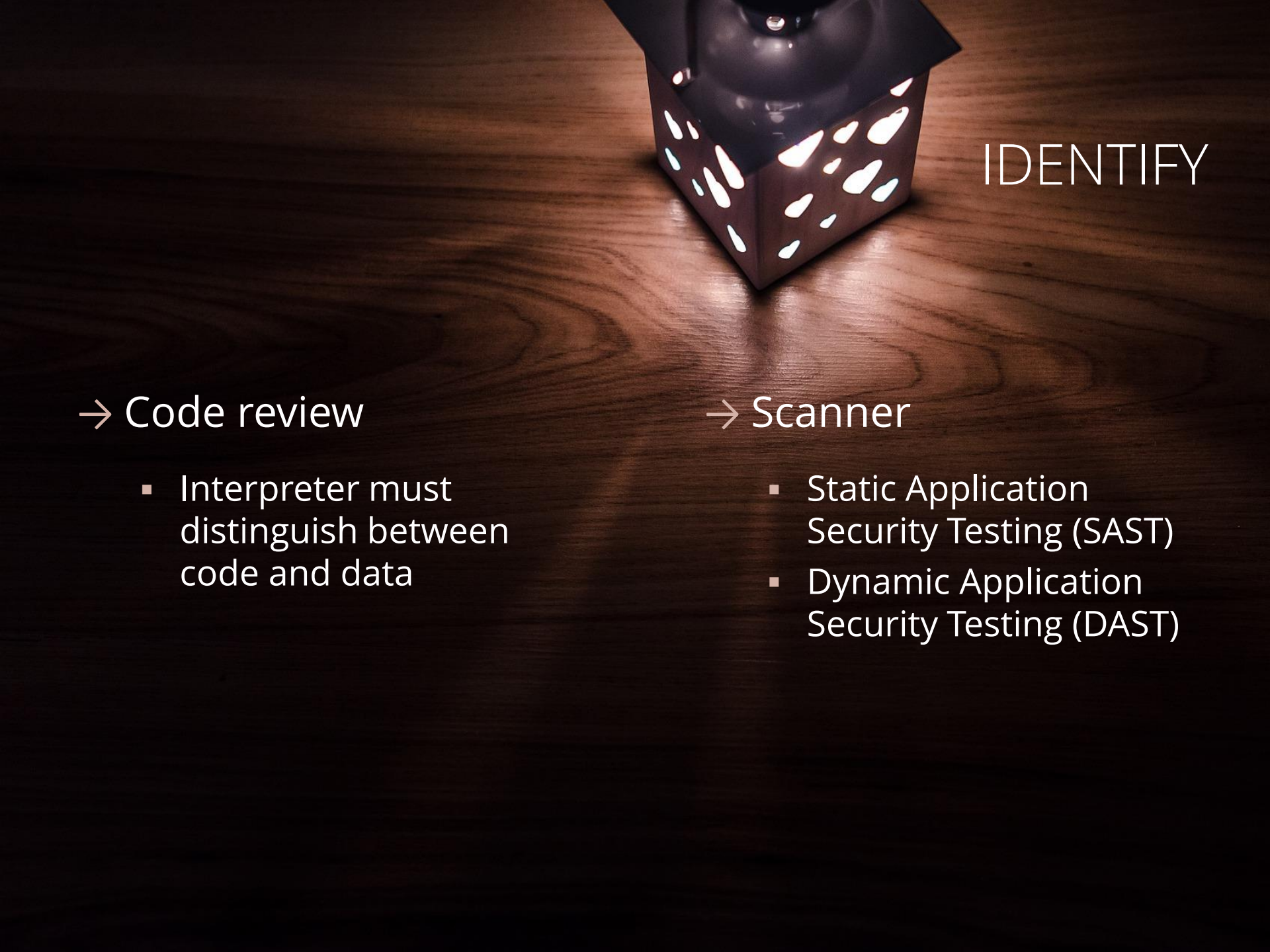
Risk Assessment

Exploitability **3**

Prevalence **2**

Detectability **3**

Technical impact **3**



IDENTIFY

→ Code review

- Interpreter must distinguish between code and data

→ Scanner

- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST)

MEASURES

- Input validation, including data from own systems
- Strict definition of all input data
- Secure transfer to interpreter (e.g. prepared statements)
- Output encoding suitable to context / interpreter

oneconsult®

API9:2019 Improper Assets Management

OVERVIEW

Description

- Vendors lack an overview of existing systems, API versions or API endpoints – therefore they are not retired and not patched for known security vulnerabilities
- Attackers get access to data from the production systems via outdated endpoints, which can lead to complete system takeover

Risk Assessment

Exploitability **3**

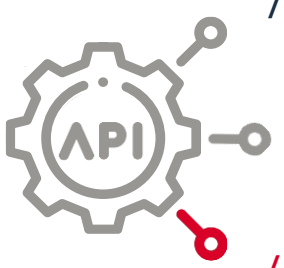
Prevalence **3**

Detectability **2**

Technical impact **2**

CAUSE / RESULT

Missing documentation

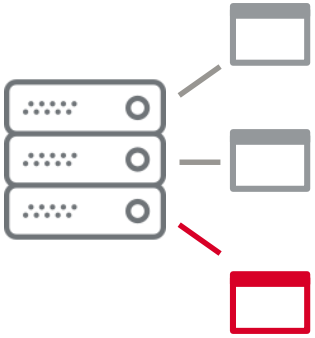


/v2/create_comment

/v2/rate_talk

/v1/get_all_users

Unknown API hosts



view-api.talks.com

mobile-api.talks.com

qa2.dev.talks.com

MEASURES

- Inventory hosts and document API versions
- Separate production from test and development environment
- Same security measures in all environments
- Network and API firewalling, also internally
- Force authentication, login before internal services

oneconsult®

API10:2019 Insufficient Logging & Monitoring

OVERVIEW

Description

- Insufficient logs and monitoring
- No weakness; lack of technical and organizational protective measures
- Attackers can carry out attacks without being noticed
- Difficult to reconstruct in retrospect, if and which data was leaked

Risk Assessment

Exploitability 2

Prevalence 3

Detectability 1

Technical impact 2

CAUSE

No logs

No monitoring

Insufficient or decentralized logs

No incident response process

MEASURES

- Logging errors during authentication & authorization
- Logging on central server
- Logging user context data to identify suspicious accounts (but: no sensitive data!)
- Simple and unified log format
- Automatic alerts and incident response process

Summary

MAIN REASONS AND CORRECTIVE ACTIONS

Interpreters and injections	Input validation Output encoding
Too much trust (especially in client)	Server-side validation Authn/Authz with *every* access
Complexity (authn/z, misconfiguration, ...)	Standardization Documentation Automation
Lack of defense in depth	Isolating infrastructure components from each other

OWASP PROActive CONTROLS

FOR DEVELOPERS

2018 **v 3.0**

OWASP ASVS LEVELS

THANKS FOR YOUR ATTENTION!

→ Frank Ully

Senior Penetration Tester & Security Consultant
Oneconsult Deutschland

frank.ully@oneconsult.com

CONTACT US

Switzerland

Oneconsult AG
Schützenstrasse 1
8800 Thalwil

Tel +41 43 377 22 22
info@oneconsult.com

Oneconsult AG
Bärenplatz 7
3011 Bern

Tel +41 31 327 15 15
info@oneconsult.com

Germany

Oneconsult Deutschland GmbH
Agnes-Pockels-Bogen 1
80992 München

Tel +49 89 248820 600
info@oneconsult.de

Resources

RESOURCES

OWASP I

- OWASP API Security Project: https://owasp.org/www-project-api-security/
- OWASP API Security on GitHub: https://github.com/OWASP/API-Security
- APISecurity.io OWASP API Security Top 10 Cheat Sheet: https://apisecurity.io/encyclopedia/content/owasp/owasp-api-security-top-10-cheat-sheet.htm
- OWASP REST Security Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html

OWASP II

- OWASP ASVS (Application Security Verification Standard): https://owasp.org/www-project-application-security-verification-standard/
- OWASP Top 10 Proactive Controls for Developers: https://owasp.org/www-project-proactive-controls/
- OWASP Web Security Testing Guide: https://owasp.org/www-project-web-security-testing-guide/
- OWASP Code Review Guide: https://wiki.owasp.org/index.php/Category:OWASP_Code_Review_Project

OWASP III

- OWASP Juice Shop
  (intentionally vulnerable application based on Node.js and Angular):
  https://owasp.org/www-project-juice-shop/

- OWASP Top 10 2017:
  https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf

- OWASP Top 10 Raw Data on GitHub:
  https://github.com/OWASP/Top10

- OWASP Cheat Sheets:
  https://cheatsheetseries.owasp.org/

INTERCEPTION PROXYS

- PortSwigger Burp Proxy (Commercial and Community Edition): https://portswigger.net/burp
- OWASP Zed Attack Proxy (ZAP, open source): https://www.zaproxy.org/
- Telerik Fiddler (Windows, free): https://www.telerik.com/fiddler

RESOURCES

OTHER

- PortSwigger Web Security Academy:
  https://portswigger.net/web-security
- Pragmatic Web Security (Philippe de Ryck):
  https://pragmaticwebsecurity.com/resources.html
- API Security Checklist:
  https://github.com/shieldfy/API-Security-Checklist
- APISecurity.io Newsletter:
  https://apisecurity.io/

RESOURCES

BOOKS

- Yaworksi, Peter (2019):
  Real-World Bug Hunting: A Field Guide to Web Hacking

- Stuttard, Dafydd / Pinto, Marcus (2007):
  The Web Application Hacker's Handbook:
  Finding and Exploiting Security Flaws, 2nd Edition

RESOURCES

JSON WEB TOKEN

- Payloads All The Things – JSON Web Token: https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/JSON%20Web%20Token
- The hard parts of JWT security nobody talks about: https://www.pingidentity.com/en/company/blog/posts/2020/jwt-security-nobody-talks-about.html
- JSON Web Token Best Current Practices: https://tools.ietf.org/html/draft-ietf-oauth-jwt-bcp-07

CROSS-ORIGIN RESOURCE SHARING

- The Complete Guide to CORS (In)Security: https://www.bedefended.com/papers/cors-security-guide
- Exploiting CORS misconfigurations for Bitcoins and bounties: https://portswigger.net/blog/exploiting-cors-misconfigurations-for-bitcoins-and-bounties
- Authoritative guide to CORS (Cross-Origin Resource Sharing) for REST APIs: https://www.moesif.com/blog/technical/cors/Authoritative-Guide-to-CORS-Cross-Origin-Resource-Sharing-for-REST-APIs/
- CORScanner / Corsy: https://github.com/chenjj/CORScanner / https://github.com/s0md3v/Corsy

NODE.JS SECURITY

- Node.js Best Practices – Security Best Practices:
  https://github.com/goldbergyoni/nodebestpractices
- Express Best Practices - Security:
  https://expressjs.com/en/advanced/best-practice-security.html
- Awesome Node.js Security resources:
  https://github.com/lirantal/awesome-nodejs-security
- “Essential Node.js Security” (book):
  https://www.amazon.de/gp/product/1365698556/