

Learning from the mistakes of others

Understanding web security through counterexamples

Clemens Hübner
inovex GmbH





Clemens Hübner

Software Security Engineer @ inovex
Helps secure applications, still hacks them
Located in Munich

 @ClemensHuebner

 clemens.huebner@inovex.de

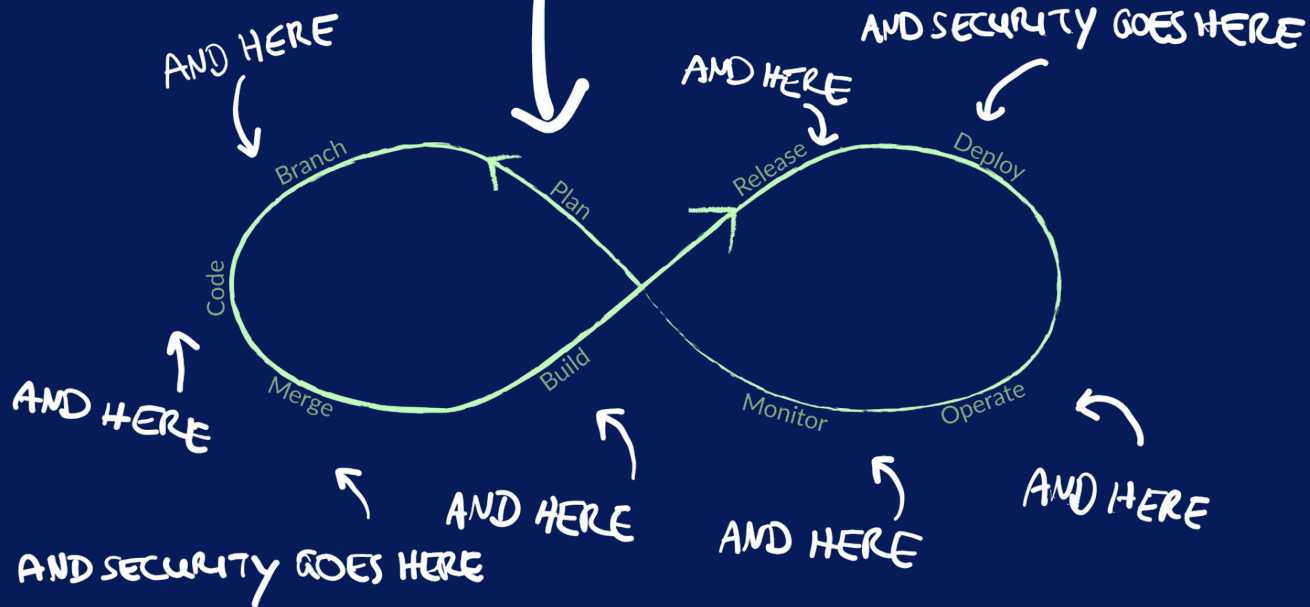
 @clemens@infosec.exchange

 /clemens-huebner

 @inovexlife

blog.inovex.de

SECURITY GOES HERE



Preliminary

learning from
the mistakes of others



fingering & blaming



OWASP Top Ten

1. Broken Access Control
2. Cryptographic Failures
3. Injection
4. Insecure Design
5. Security Misconfiguration
6. Vulnerable and Outdated Components
7. Identification and Authentication Failures
8. Software and Data Integrity Failures
9. Security Logging and Monitoring Failures
10. Server-Side Request Forgery (SSRF)

OWASP Top Ten



- 1. Broken Access Control → **IDOR**
 - 2. Cryptographic Failures → **JWT Attacks**
 - 3. Injection → **XSS-Phishing**
 - 4. Insecure Design → **Web Race Conditions**
 - 5. Security Misconfiguration → **Public S3 Buckets**
 - 6. Vulnerable and Outdated Components → **log4shell**
 - 7. Identification and Authentication Failures → **Credential Stuffing**
 - 8. Software and Data Integrity Failures
 - 9. Security Logging and Monitoring Failures
 - 10. Server-Side Request Forgery (SSRF)
- Insecure APIs** (points to the top of the list)

OWASP Top Ten



- 1. **Broken Access Control** → **IDOR**
- 2. Cryptographic Failures → **Insecure APIs**
- 3. Injection → **JWT Attacks**
- 4. Insecure Design → **XSS-Phishing**
- 5. Security Misconfiguration → **Web Race Conditions**
- 6. Vulnerable and Outdated Components → **Public S3 Buckets**
- 7. Identification and Authentication Failures → **log4shell**
- 8. Software and Data Integrity Failures
- 9. Security Logging and Monitoring Failures → **Credential Stuffing**
- 10. Server-Side Request Forgery (SSRF)

Insecure Direct Object Reference (IDOR)

- DiGA Novega: Treatment of depression via web application through video/audio and interactive exercises
- Data download (required by GDPR)
 - GET <https://www.novego.com/myaccount/participant/data-export/21378>
 - 21378 = User ID

Insecure Direct Object Reference (IDOR)

GET <https://www.novego.com/myaccount/participant/export/21377>

- Email address
- Gender
- therapy program registered
- self-assessment results

found by

ZERFORSCHUNG

GAD-7 Anxiety

Over the last two weeks, how often have you been bothered by the following problems?	Not at all	Several days	More than half the days	Nearly every day
1. Feeling nervous, anxious, or on edge	0	1	2	3
2. Not being able to stop or control worrying	0	1	2	3
3. Worrying too much about different things	0	1	2	3
4. Trouble relaxing	0	1	2	3
5. Being so restless that it is hard to sit still	0	1	2	3
6. Becoming easily annoyed or irritable	0	1	2	3
7. Feeling afraid, as if something awful might happen	0	1	2	3

Insecure GraphQL API

- Grocery delivery start-ups Flink und Gorillas both use GraphQL-APIs
- GraphQL
 - Introspection: self-documented API
 - client defines return object
- Similar flaws found at Flink (03/2021) and Gorillas (05/2021)

found by

ZERFORSCHUNG

Insecure GraphQL API: Flink

- self-documented GraphQL API

- `order (id ID!) Order`
Look up an order by ID.
- `orders (sortBy OrderSortingInput, filter OrderFilterInput, created ReportingPeriod, status OrderStatusFilter, before String, after String, first Int, last Int) OrderCountableConnection`
List of orders.
- `draftOrders (sortBy OrderSortingInput, filter OrderDraftFilterInput, created ReportingPeriod, before String, after String, first Int, last Int) OrderCountableConnection`
List of draft orders.

Insecure GraphQL API: Flink

- self-documented GraphQL API

```
1 {  
2   orders(first: 1){  
3     totalCount  
4   }  
5 }
```

```
1 {  
2   "data": {  
3     "orders": {  
4       "totalCount": 3953  
5     }  
6   }  
7 }
```

Insecure GraphQL API: Flink

- self-documented GraphQL API

```
1 {
2   "data": {
3     "order": {
4       "billingAddress": {
5         "firstName": "e",
6         "lastName": "n",
7         "streetAddress1": "7",
8         "city": "BERLIN",
9         "postalCode": "10825",
10        "phone": "+491",
11      },
12      "userEmail": ".de",
13      "lines": [
14        {
15          "productName": "Innocent Smoothie Berry Good 250ML",
16        }
17      ],
18      "payments": [
19        {
20          "creditCard": {
21            "brand": "visa",
22            "lastDigits": "",
23          }
24        }
25      ]
26    }
27 }
28 }
```

Insecure GraphQL API: Gorillas

Fields

```
activeOrders(  
  filter: GraphFilterScalar  
  feed: GraphFeedInput  
): [MarketOrderData]
```

Get Order Data

```
ordersData(  
  filter: GraphFilterScalar  
  feed: GraphFeedInput  
): [OrderData]
```

Get Order Data

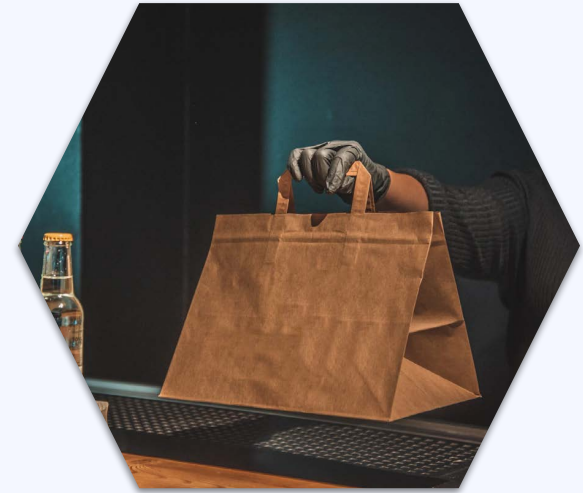
```
tenantConfig(  
  filter: GraphFilterScalar  
  feed: GraphFeedInput  
): [TenantConfig]
```

Get Tenant Configuration Data

```
{  
  "data": {  
    "ordersData": [  
      {  
        "completedOn": "2020-10-31",  
        "createdOn": "2020-10-31",  
        "customer": {  
          "email": "  
          "name": "  
          "phone": "  
          "uid": "  
          "userId": "  
        },  
        "workerData": {  
          "activeWorker": {  
            "name": "  
            "phoneNumber": "+49 "  
          }  
        },  
        "description": "Im hof nach links "  
        "paymentData": {  
          "transactions": [  
            {  
              "amount": 6.600000023841858,  
              "creditCard": {  
                "id": "  
                "name": "  
                "expiry": "  
                "paymentGateway": "STRIPE"  
              }  
            }  
          ]  
        },  
        "dispatch": {  
          "dropOff": {  
            "address": {  
              "addressDetailInfo": "  
              "coordinates": {
```

Takeaways

- IDOR:
 - Don't use incrementing IDs, but e.g. UUIDs
 - Don't rely on secrecy of IDs
- Insecure API:
 - GraphQL needs other Authz measures
- In general:
 - Design Authz early
 - Follow principle of least privilege
 - Deny by default
 - Test for misuse cases



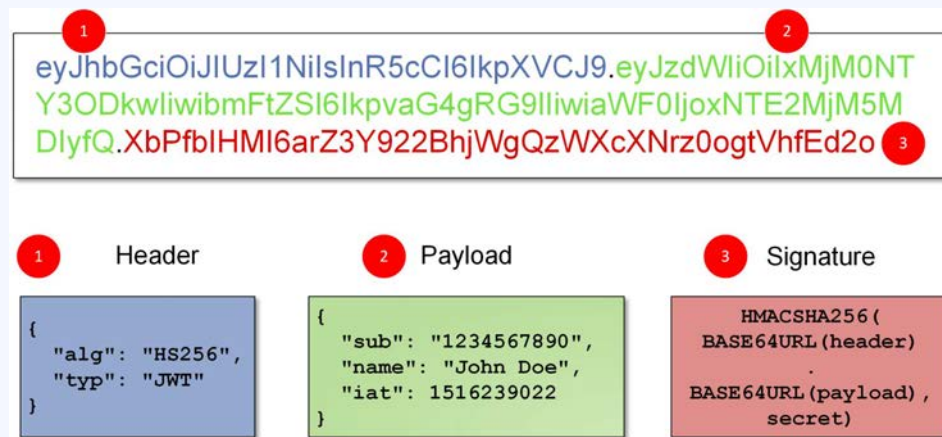
OWASP Top Ten



- 1. Broken Access Control → **IDOR**
- 2. **Cryptographic Failures** → **Insecure APIs**
- 3. Injection → **JWT Attacks**
- 4. Insecure Design → **XSS-Phishing**
- 5. Security Misconfiguration → **Web Race Conditions**
- 6. Vulnerable and Outdated Components → **Public S3 Buckets**
- 7. Identification and Authentication Failures → **log4shell**
- 8. Software and Data Integrity Failures
- 9. Security Logging and Monitoring Failures → **Credential Stuffing**
- 10. Server-Side Request Forgery (SSRF)

JWT Failures

- JWT = JSON Web Token
- standard for JSON-Payload that is signed and/or encrypted
- often used for Authn/Authz tokens



Design flaws of JWT as Authn token

- Logout is impossible
 - self-contained token are valid as long they are not expired
 - logout = revocation is not possible without losing JWT's advantages
- Change of permissions/roles is hard
 - needs re-issue of token
- Good reasons JWT might be no good idea at all
 - [Stop using JWT for sessions](#)

Implementation flaws of JWTs

- complex cryptography without secure defaults
 - RSA PKCS#1 v1.5 problematic since 1998
- over-engineered standard
 - too many use cases leading to unreasonable variants

4.1.3. "jwk" (JSON Web Key) Header Parameter

The "jwk" (JSON Web Key) Header Parameter is the public key that corresponds to the key used to digitally sign the JWS. This key is represented as a JSON Web Key [JWK]. Use of this Header Parameter is OPTIONAL.

- flexible specification, leading to error-prone implementations
 - None-Attack
 - polyglot JWTs

"alg" Param Value
HS256
HS384
HS512
RS256
RS384
RS512
ES256
ES384
ES512
PS256
PS384
PS512
none

Implementation flaws of JWTs: None-Attack

- None is also a valid algorithm, meaning the signature is not checked

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIb251IiwidHlwIjoiS1dUIn0.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG91IiwiaWF0IjoxNTE2MzE2MjM5Lj1KxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

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

PAYLOAD: DATA

```
{  "sub": "1234567890",  "name": "John Doe",  "iat": 1516239022}
```

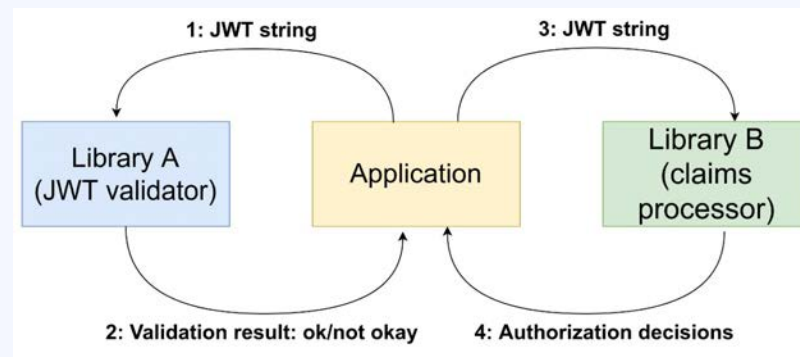
VERIFY SIGNATURE

```
HMACSHA256(  base64UrlEncode(header) + "." +  base64UrlEncode(payload),  your-256-bit-secret)
```

secret base64 encoded



Implementation flaws of JWTs: Polyglot JWTs





Implementation flaws of JWTs: Polyglot JWTs

```
{  
  "AAAA": ".XXXX.",  
  "protected": "AAAA",  
  "payload": "BBBB",  
  "signature": "CCCC"  
}
```





Implementation flaws of JWTs: Polyglot JWTs

```
{  
  "AAAA": ".XXXX.",  
  "protected": "AAAA",  
  "payload": "BBBB",  
  "signature": "CCCC"  
}
```

jwtcrypto ignored unknown JSON fields:

```
{  
  "header": "AAAA",  
  "payload": "XXXX",  
  "protected": "AAAA",  
  "payload": "BBBB",  
  "signature": "CCCC"  
}
```

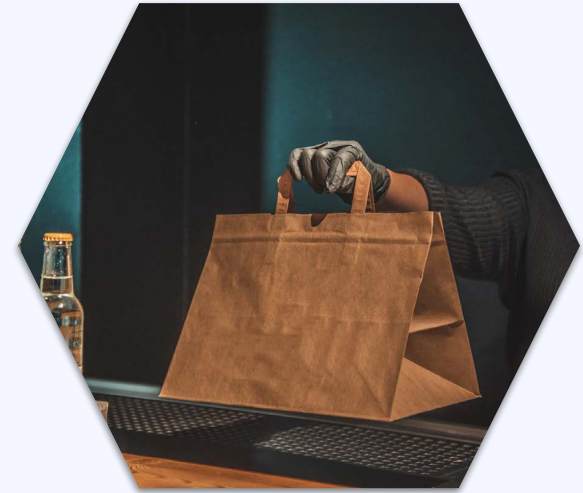
python-jwt split on periods, and ignored non-base64 characters:

```
{  
  "AAAA": ".XXXX.",  
  "protected": "AAAA",  
  "payload": "BBBB",  
  "signature": "CCCC"  
}
```

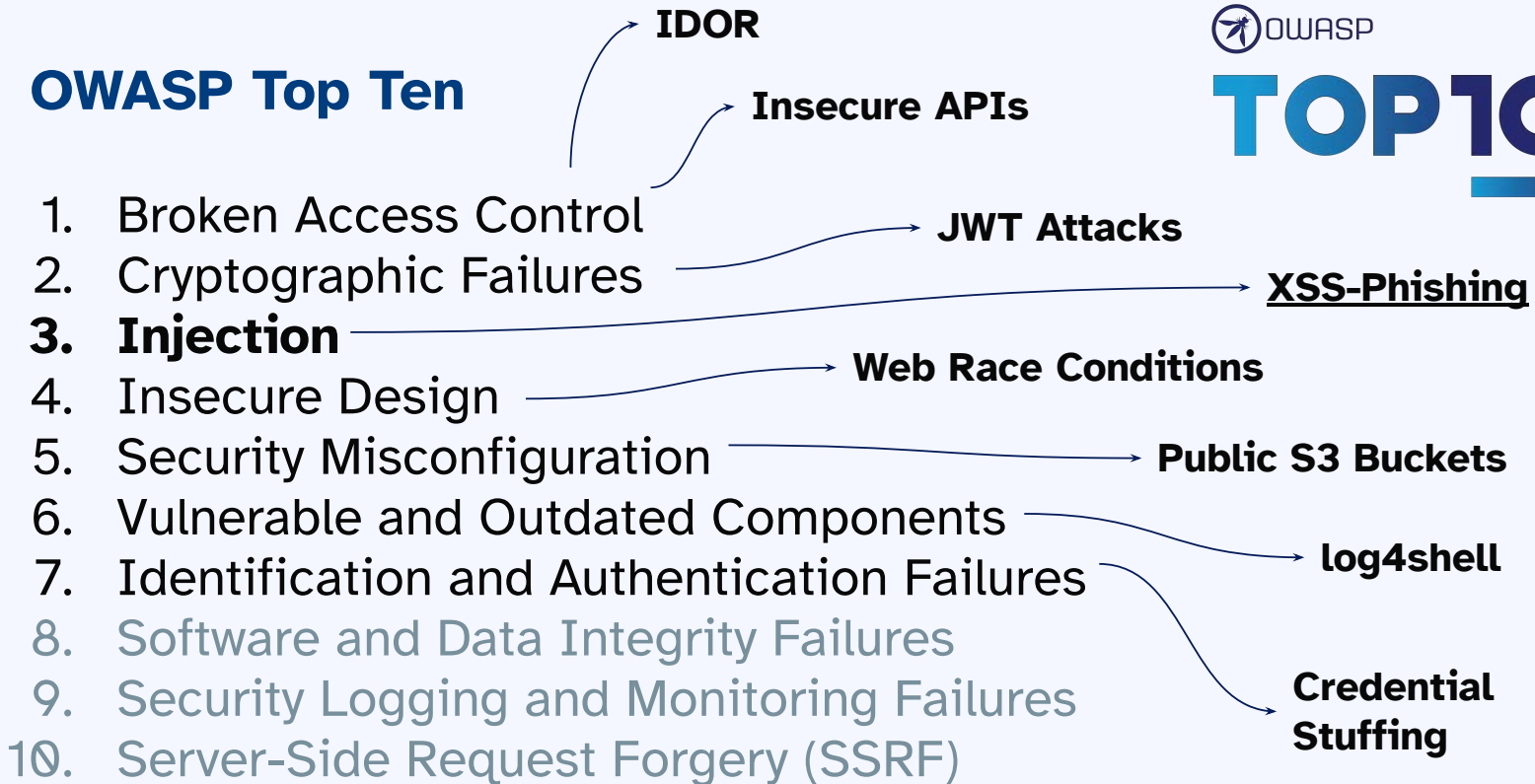


Takeaways

- Consider alternatives to JWT
 - use Session Token
 - use PASETO / Macaroon / Biscuits
- Use modern, patched JWT library
- Set algorithm explicitly



OWASP Top Ten



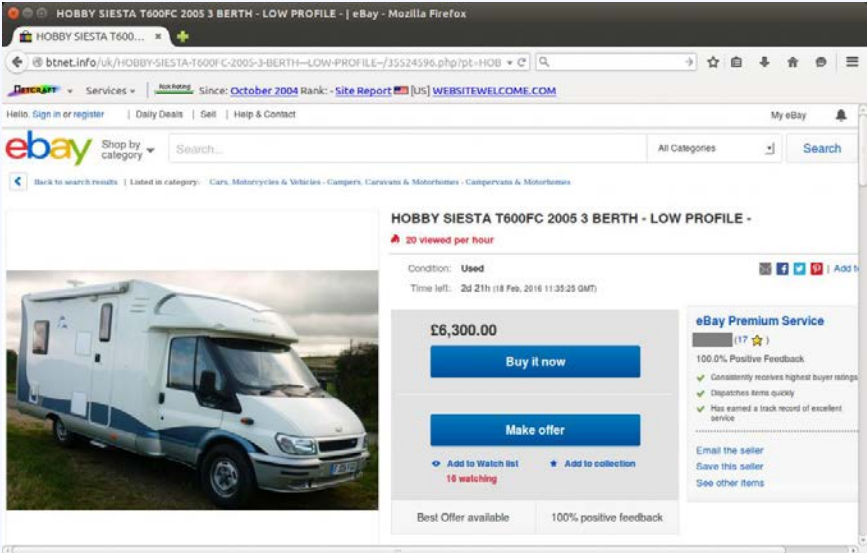
Cross-Site-Scripting (XSS)

- User input is contained in the application in a way allowing an attacker to execute arbitrary scripts in the victim's browser

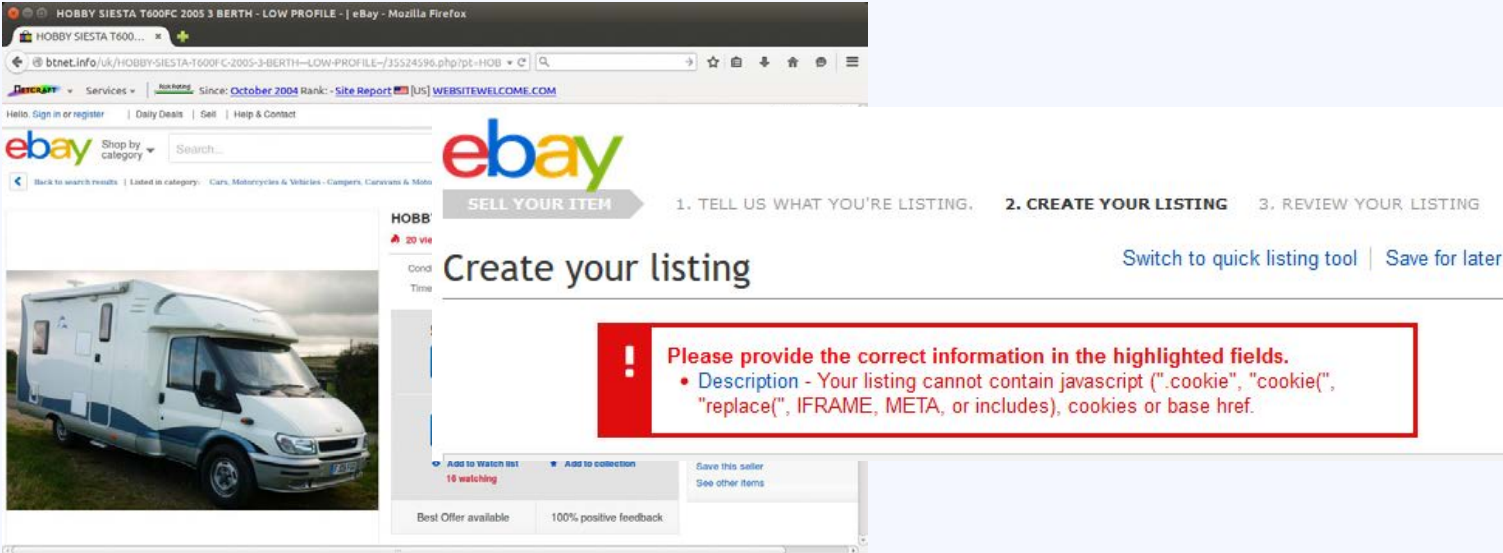
```
<script>alert(42);</script>
```

- Affects all user controlled data, including URL parameter, request body, header and cookies

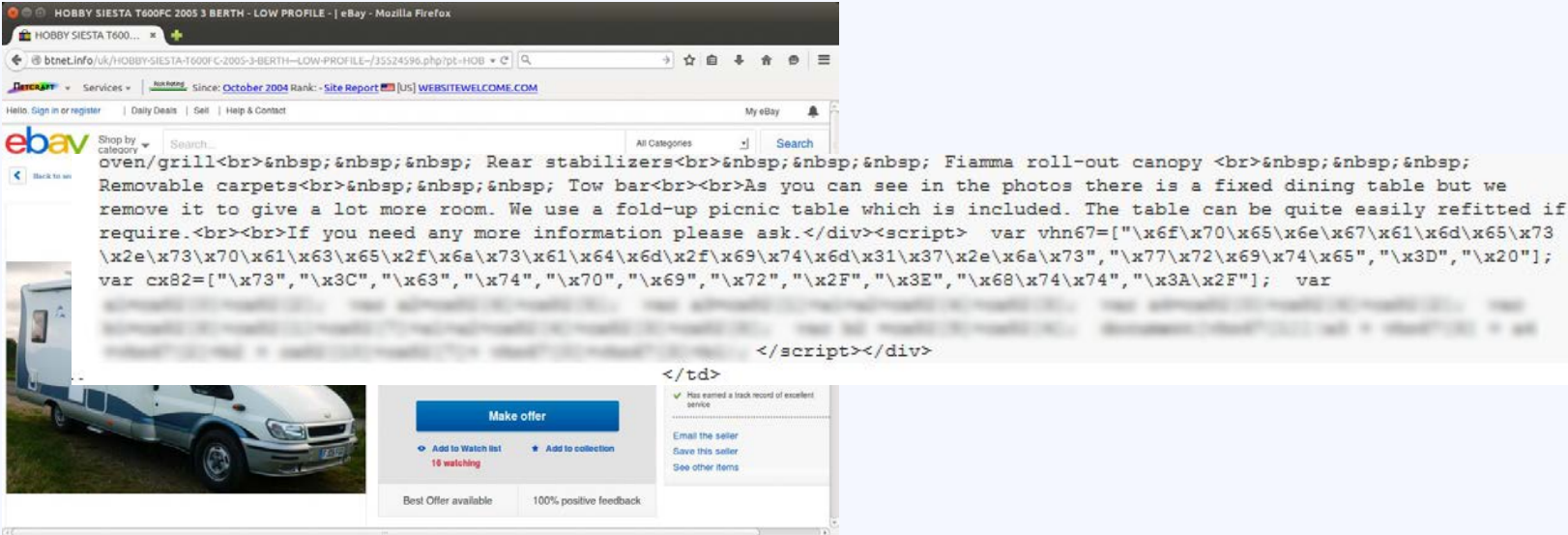
Stored XSS: Ebay (2015)



Stored XSS: Ebay (2015)



Stored XSS: Ebay (2015)



Stored XSS: Ebay (2015)

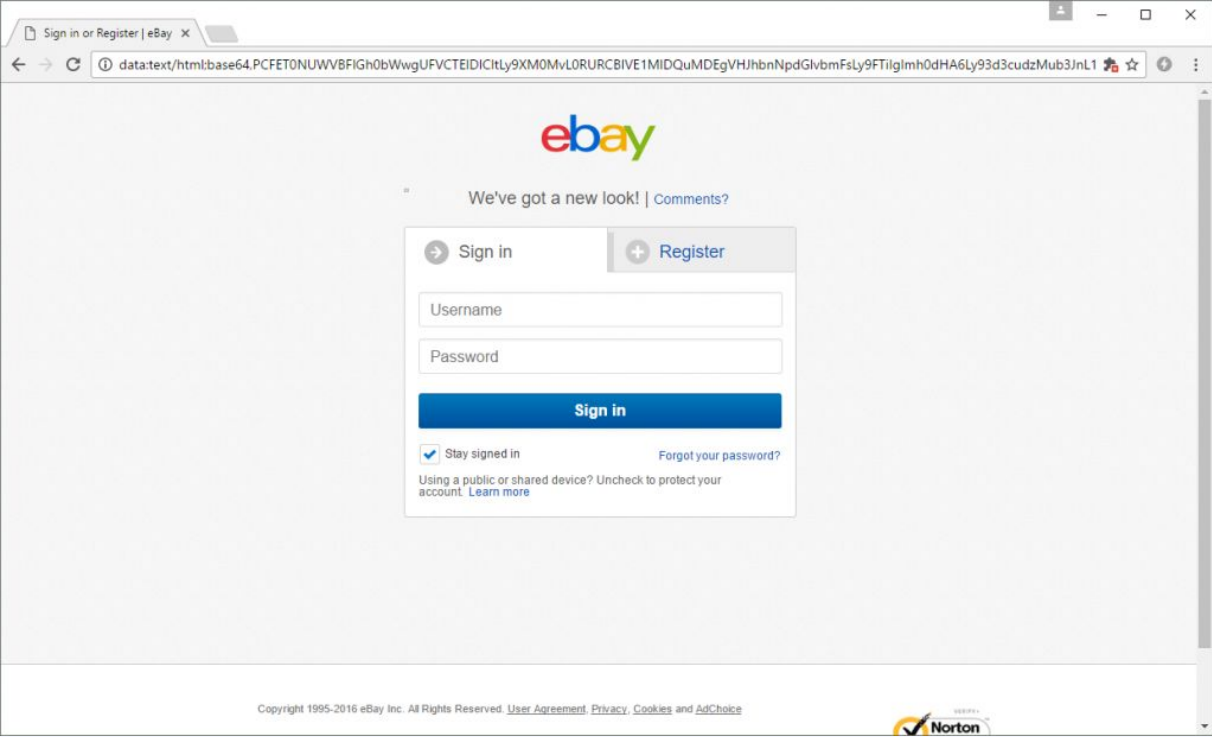


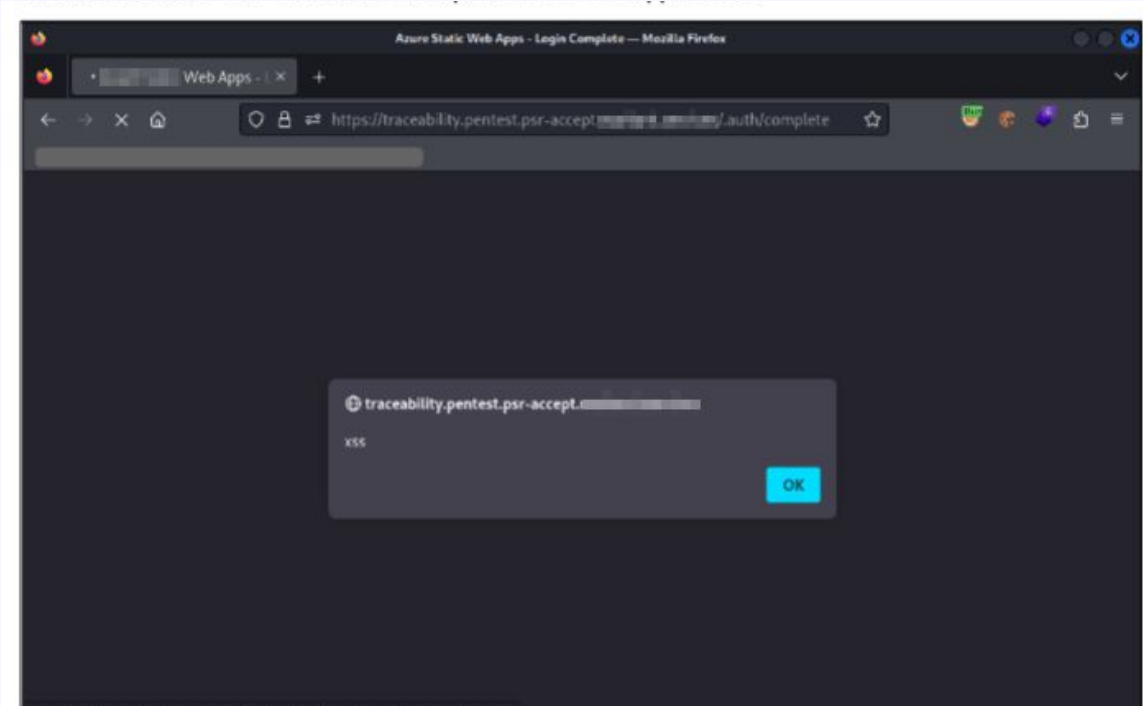


Photo by cottonbro studio

XSS - Yesterday's snow?

- Modern SPA frameworks prevent XSS quite successful
- No longer a mass phenomenon
- XSS tends to be forgotten
- But: still relevant, when no framework is used

OAuth Solution in Hyperscaler Cloud Product...



OAuth Solution in Hyperscaler Cloud Product...

https://traceability.XXXXXXX.com/.auth/login/aad?post_login_redirect_uri=http%253A%252F%252Fredirect.example.org



```
<head>
  <title> Login Complete</title>
  <script type=text/javascript>
    window.onload = function () {
      try { document.getElementById('redirectLink').click()
        }catch(a){};
    }
  </script>
</head>
<body>
  <a id=redirectLink href=/.auth/complete/#/http%3A%2F%2Fredirect.example.org
    Click here if it doesn't automatically redirect.
  </a>
</body>
```

OAuth Solution in Hyperscaler Cloud Product...

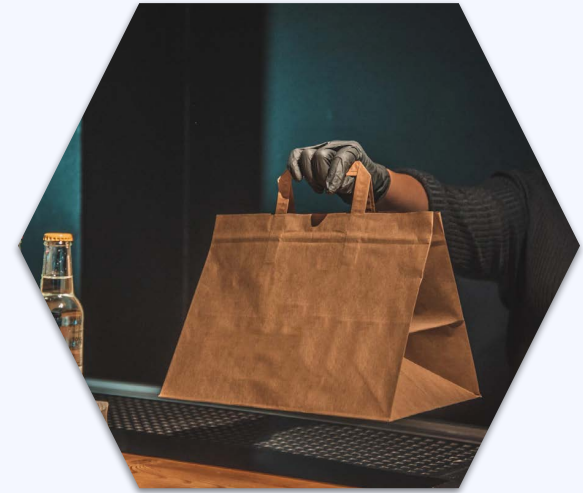
https://traceability.XXXXXXX.com/.auth/login/aad?post_login_redirect_uri=%23/%3E%3Cscript%3Ealert(%27xss%27)%3C/script%3E



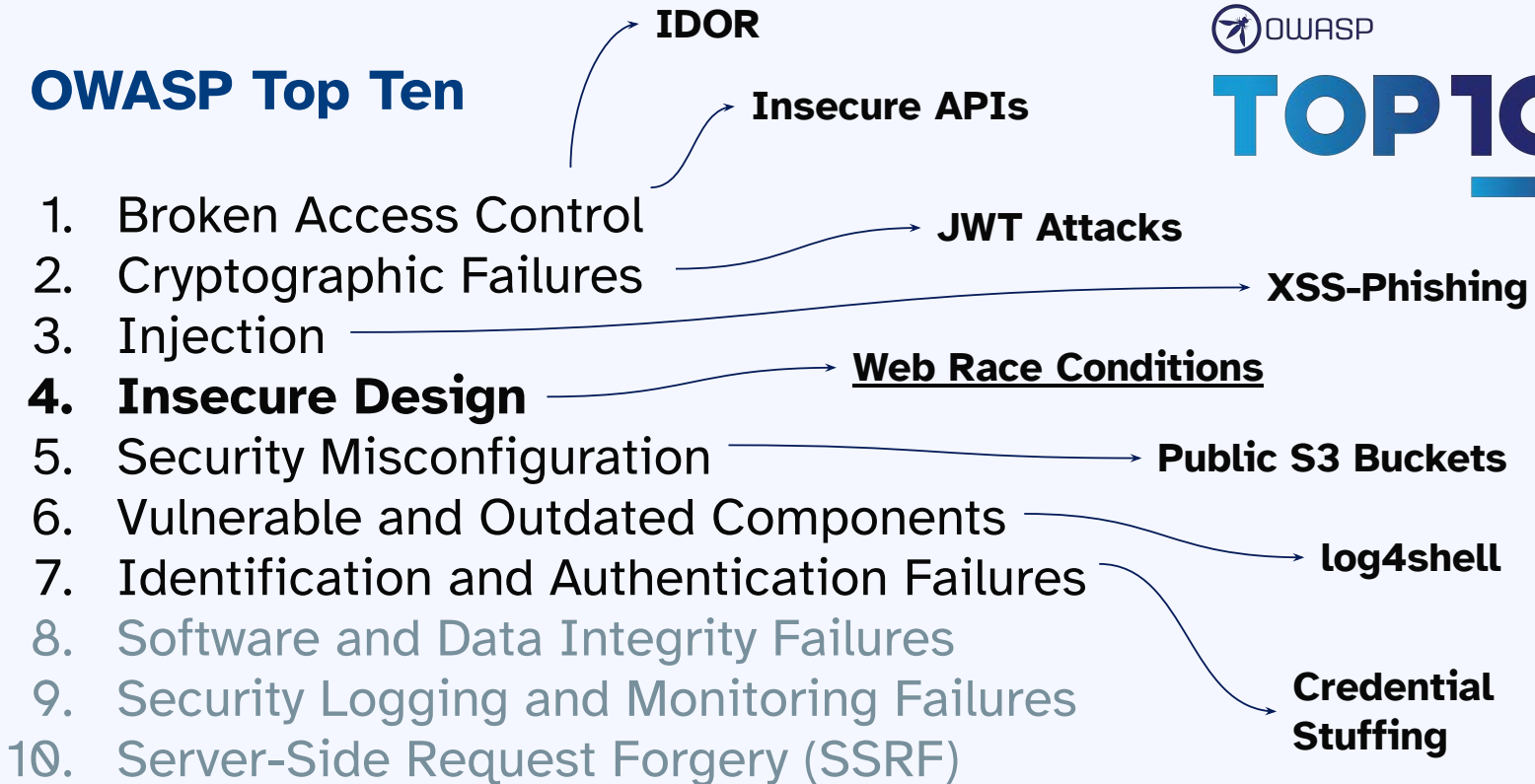
```
<head>
  <title> Login Complete</title>
  <script type=text/javascript>
    window.onload = function () {
      try { document.getElementById('redirectLink').click()
        }catch(a){};
    }
  </script>
</head>
<body>
  <a id=redirectLink href=/.auth/complete/#/><script>alert('xss')</script>>
    Click here if it doesn't automatically redirect.
  </a>
</body>
```

Takeaways

- How are you handling XSS?
 - Modern web frameworks do it quite well
 - Don't forget it everywhere else!
- Do defense in depth
 - HttpOnly Cookies
 - Content Security Policy (CSP)
 - (WAF)



OWASP Top Ten



Insecure Design: Web Race Conditions

- exploit race conditions between security relevant process steps
 - redeeming a voucher
 - usage of one-time-token (e.g. OTP)
 - withdrawing/transferring of money
 - identification/registration
- network latency, jitter and internal latency cause exploitable delays

Web Race Conditions: Bonify (2023)

Bonify: FinTech-Startup for to assess creditworthiness of applicants

Authentication and identification via bank account

bonify — das Schufa-Startup, das jedem sagt, ob Du kreditwürdig bist



Lilith Wittmann · [Follow](#)

8 min read · Jul 24

[bonify — das Schufa-Startup, das jedem sagt, ob Du kreditwürdig bist | by Lilith Wittmann](#)

Web Race Conditions: Bonify (2023)

Microservice architecture with service-to-service communication

Registration process:

1. Creation of user in user-service
2. Login in online banking, under supervision of banking-service
3. Request of name information held in banking-service by the user-service
4. Flagging of user as “validated”

Web Race Conditions: Bonify (2023)

Microservice architecture with service-to-service communication

Registration process:

1. Creation of user in user-service
2. Login in online banking, under supervision of banking-service
3. Request of name information held in banking-service by the user-service
4. Flagging of user as “validated”

Change of user's name



Dein Boniversum-Score ↑



Deine Daten werden von Creditreform Boniversum abgerufen und regelmäßig aktualisiert.

Deine Personendaten
Herr Jens Spahn, 16.4.1980 ✓

Deine Adressdaten
Auf dem Grat 38, 14195 Berlin ✓

Deine positiven Zahlungserfahrungen
Keine Einträge ✓

Deine negativen Zahlungserfahrungen
Keine Einträge ✓

Dein Bonitätsscore

Der Scorewert gibt Auskunft über deine Kreditwürdigkeit. Er bewertet die Fähigkeit, Rechnungen und Zahlungsverpflichtungen vertragsgemäß zu begleichen.

Ein hoher Wert lässt vermuten, dass du zukünftig ein geringeres Risiko darstellst. Ein niedriger Wert kann ein Zeichen dafür sein, dass dir in der Vergangenheit Zahlungsausfälle gekommen sind.

In die Score-Berechnung fließen die unten angezeigten Faktoren ein: Einkommen, Beruf oder Vermögen.

[Mehr Infos](#)

WEITERE INFORMATIONEN

Die Kreditwürdigkeit

Fakten über gute Bonität

Gründe für eine schlechte Bonität

Deine Inkassodaten
Keine Einträge ✓

Deine Gerichtsdaten
Keine Einträge ✓

Deine Unternehmensbeteiligungen
Keine Einträge ✓

Wurde deine Bonität abgefragt?
Keine Einträge ✓

● positiven Effekt auf deinen Bonitätsscore

● neutralen Effekt auf deinen Bonitätsscore

● negativen Effekt auf deinen Bonitätsscore



Berlin, den 22.07.2023

Mieterauskunft

Diese Mieterauskunft wird von bonify zur Verfügung gestellt; bonify ist ein Produkt der Fortell GmbH. Die Bonitätsdaten werden von der Creditreform Boniversum GmbH übermittelt, die Daten zu Gehalts- und Mietzahlungen erhebt bonify durch automatisierte Auswertung von Transaktionsdaten/Bankkonten. Eine Haftung für die Richtigkeit und Vollständigkeit der Angaben ist ausgeschlossen.

Persönliche Daten

Name, Vorname	Spahn, Jens
Geburtsdatum	16.04.1980
Adresse	Auf dem Grat 38 14195 Berlin

Bonitätsinformationen

Bonität Es liegen ausschließlich positive Informationen vor

Einkommen

Auf den vorliegenden Konten konnten keine Daten zu Gehaltseingängen gefunden werden

Mietverhältnis

Auf den vorliegenden Konten können die folgenden Mietdaten gefunden werden:

Durchschnittliche monatliche Mietzahlungen	955.51 €
Anzahl der Mietzahlungen in den letzten 3 Monaten	3

Validierungscodes

Die Gültigkeit der Auskunft kann unter Eingabe des Validierungscodes auf my.bonify.de/vermieter überprüft werden 001768250229

Bonitätsdaten zur Verfügung gestellt von

Dieses Zertifikat ist gültig vom 22.07.2023 bis 22.10.2023

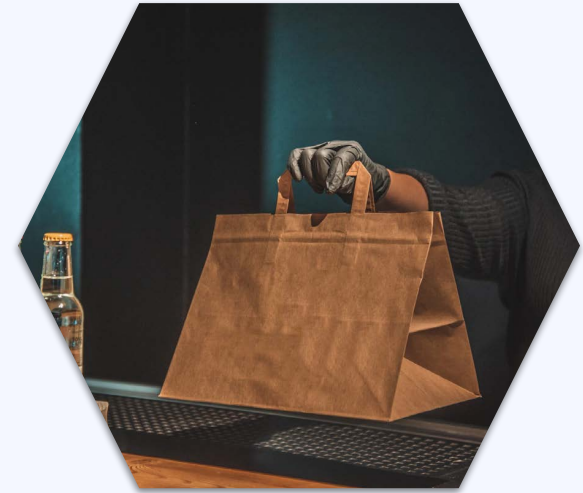
Dieses Zertifikat basiert auf Bonitätsinformationen Dritter sowie der automatisierten Auswertung von Kreditbüchern. Es dokumentiert eine automatische Momentaufnahme. Bei entsprechend guten Werten besteht eine statistische Erwartung, dass Mietausfälle unwahrscheinlicher sind. Eine Sicherheit für den Einzelfall ist jedoch nicht gewährleistet. Trotz sorgfältiger Gestaltung kann außerdem eine Fehlerhaftigkeit nicht ausgeschlossen werden. Diese kann entweder technischer Natur sein, oder aber systembedingt, so ist es nicht möglich, Mietrückstände zu erkennen, die sich



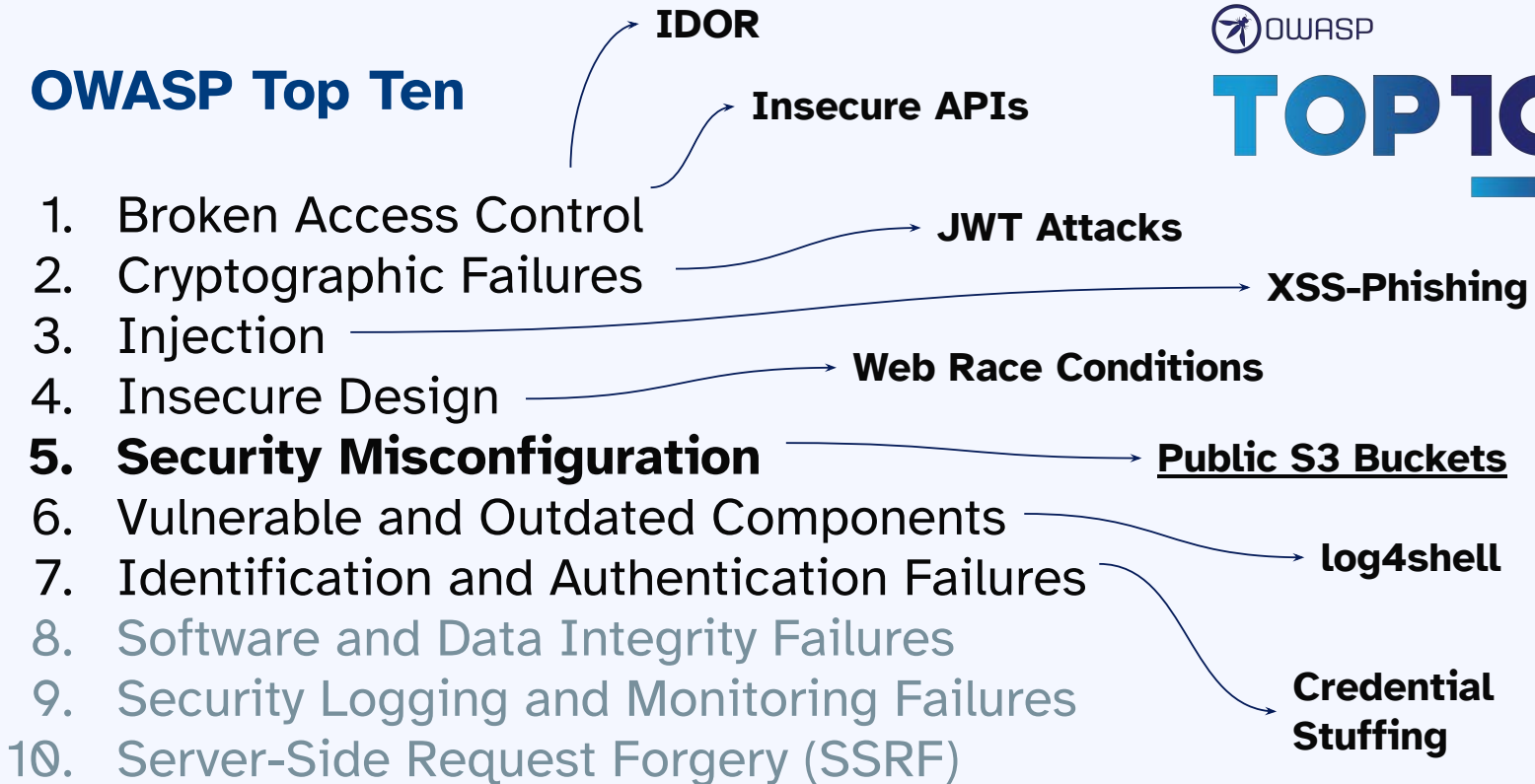
inovex

Takeaways

- Thoroughly design processes
- Take timing into account - don't rely on an implicit order
 - When is state changed?
 - Can two requests change the same record?
- Consider abuse cases
- Perform threat modelling for *attacker's view*



OWASP Top Ten



Security Misconfiguration: Public S3 Buckets

Another S3 Bucket Leads to Breach of 50k Patient Records



by Dennis Sebayon on March 15, 2021

McGraw Hill's S3 buckets exposed 100,000 students' grades and personal info

**ACCENTURE - EMBARRASSING DATA LEAK
BUSINESS DATA IN A PUBLIC AMAZON S3 BUCKET**

SECURITY THROUGH...WHAT EXACTLY? —

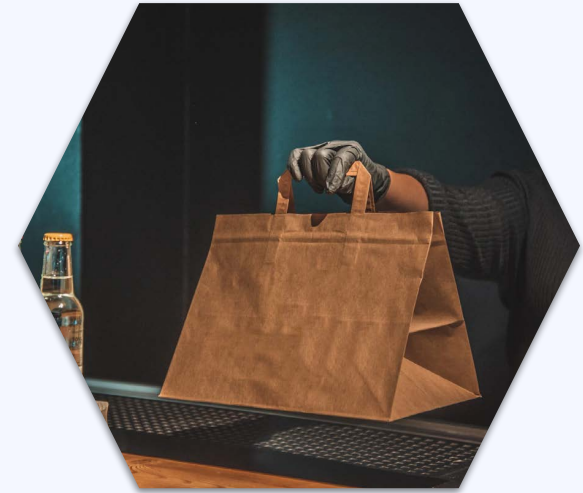
Defense contractor stored intelligence data in Amazon cloud unprotected

Alibaba Victim of Huge Data Leak as China Tightens Security

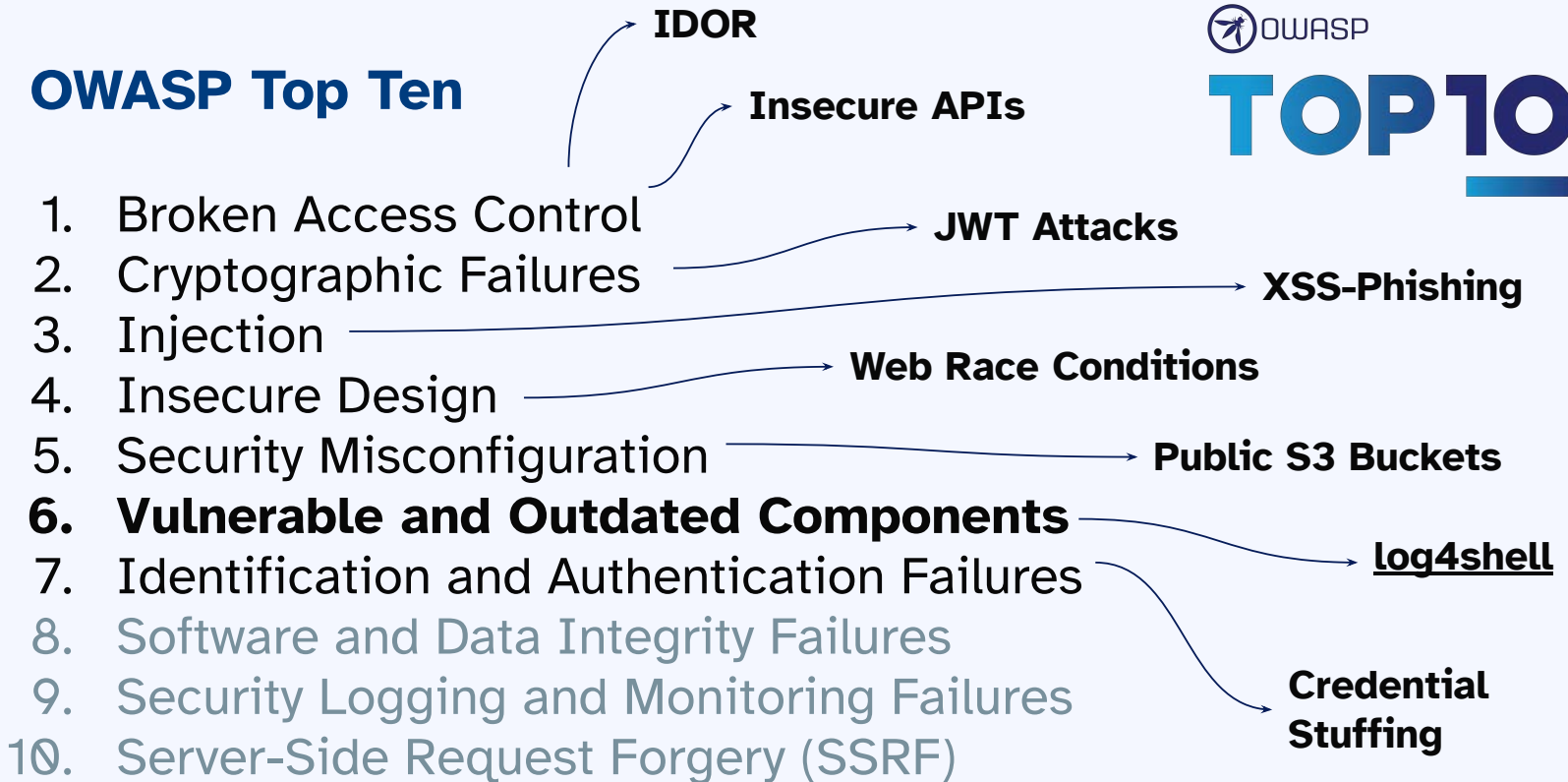
Takeaways

- Use automated, repeatable processes to deploy environments
- Identical setups for all stages (but different secrets)
- careful configuration of all layers using best practices
- Use automated test tools / external testers

- For developers: provide secure defaults



OWASP Top Ten



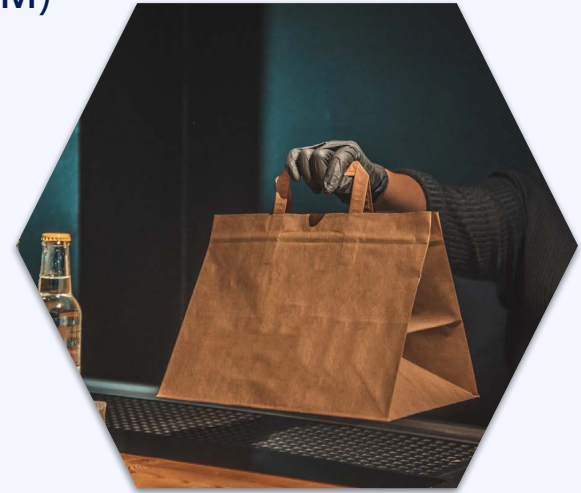
Vulnerable and Outdated Components: Log4Shell

- a zero-day vulnerability in Log4j, a Java logging framework
- allowing arbitrary code execution
- existed unnoticed since 2013, published and patched in 2021

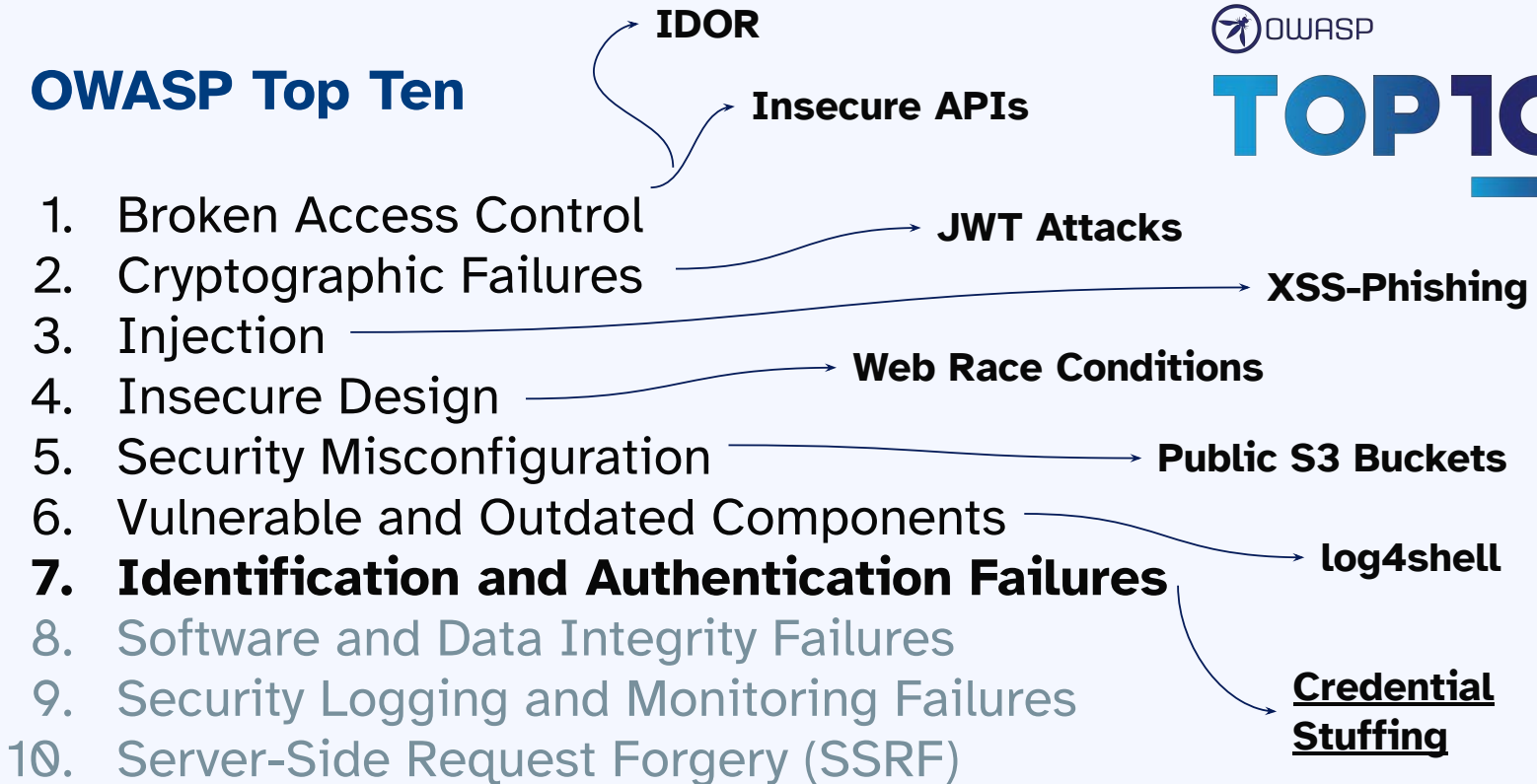
- basic and popular framework, big impact through many transitive dependencies
- common problems: Where do we use it? Are we affected?

Takeaways

- Have a detailed software inventory (e.g. SBOM)
- Reduce unnecessary dependencies
- Tool recommendations:
 - Renovate
 - OWASP Dependency Track
- Plan maintenance of your software
- Ensure fast ability of patch



OWASP Top Ten



Identification and Authentication Failures



User's mistakes with passwords:

- Weak passwords
- Very weak passwords
- Wrong handling of passwords
- Reuse of passwords

Identification and Authentication Failures



User's mistakes with passwords:

- Weak passwords → Brute Force Attacks
try a lot of passwords for a user
- Very weak passwords → Password Spraying
try weak password for multiple users
- Wrong handling of passwords → Phishing Attacks
trick the user to tell you the password
- Reuse of passwords → Credential Stuffing
try a lot of known pairs of
username/password

Credential Stuffing: 23andMe (2023)

23andMe: company to explore your own genetic material

United States

23andMe notifies customers of data breach into its 'DNA Relatives' feature

Breached using Credential Stuffing

Attackers gained access to millions of user profiles, containing name, sex, age and details about genetic ancestry

Data sets were grouped by ancestry, e.g. 1 million-user data set of Ashkenazi Jews



Home

Notify me


Domain search

Who's been pwned

Passwords

API

About

Donate  

';--have i been pwned?

Check if your email address is in a data breach

pwned?

Using Have I Been Pwned is subject to the [terms of use](#)

771

pwned websites

13,080,233,673

pwned accounts

115,769

pastes

228,884,627

paste accounts



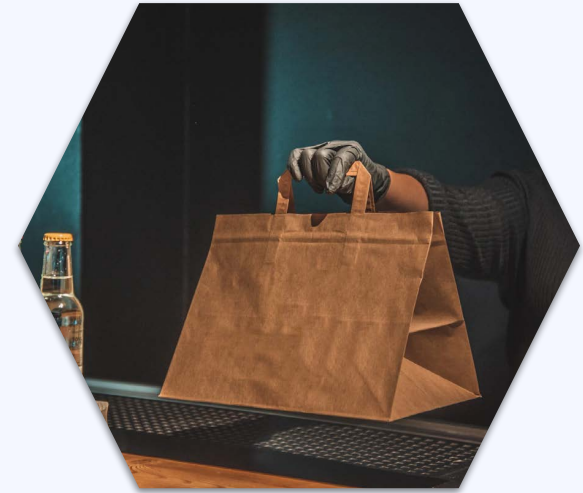
Takeaways

Users:

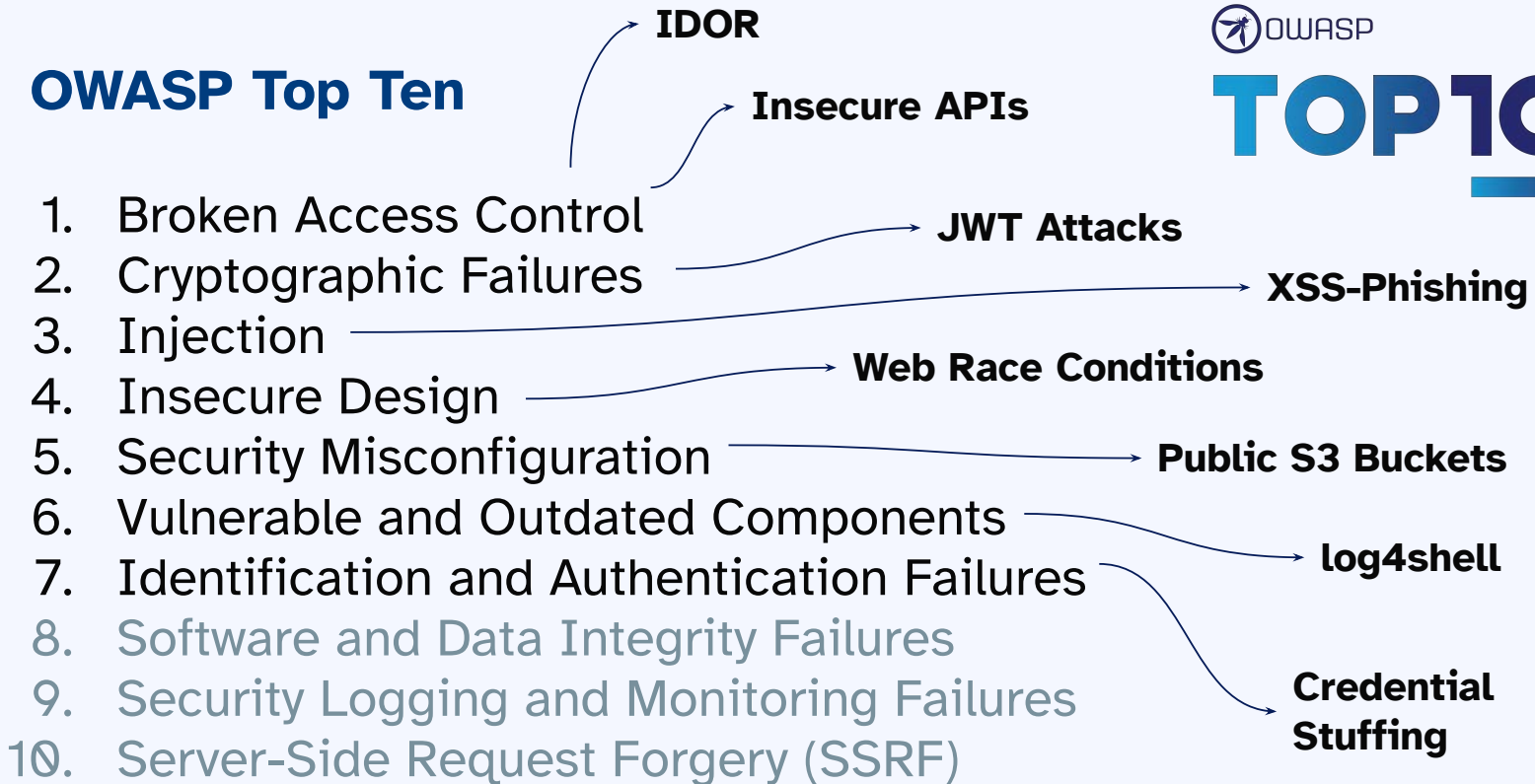
- Use secure passwords
- Check for breached passwords
- Use multi-factor (or passwordless) authentication

Devs:

- Allow multi-factor (or passwordless) authentication
- Defense in depth
 - Brute Force protection
 - CAPTCHAs
 - IP Mitigation
 - Identify leaked passwords
 - Keep usernames secret
 - Notify users about security events



OWASP Top Ten



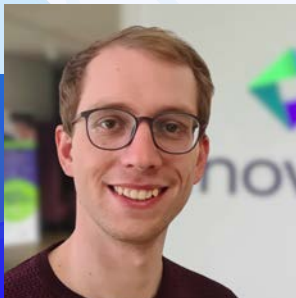
Learn from the mistakes of others

Consider security from the beginning and continuously

Keep up-to-date with threats and check with your attack surface



Thank you!



 /clemens-huebner

 @ClemensHuebner

 clemens.huebner@inovex.de

 @clemens@infosec.exchange

 @inovexlife

blog.inovex.de

inovex is an IT project center driven by innovation and quality, focusing its services on 'Digital Transformation'.

- founded in 1999
- 500+ employees
- 8 offices across Germany



www.inovex.de