February 6, 2024

Top Things You Need to Know About API Security

Dr. Philippe De Ryck & Isabelle Mauny

Introduction

About the Speakers

Dr. Philippe De Ryck

Web Security Expert

Pragmatic Web Security

Isabelle Mauny

Field CTO & Co-Founder

42Crunch

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

API Security TOP 10

Unpatched bug chain poses ‘mass account takeover’ threat to Yunmai weight monitoring app

Adam Bannister 06 June 2022 at 14:20 UTC
Updated: 06 June 2022 at 15:21 UTC

IoT Mobile Zero-day

User data related to at least 500,000 Android accounts at risk

https://portswigger.net/daily-swig/unpatched-bug-chain-poses-mass-account-takeover-threat-to-yunmai-weight-monitoring-app

MASS ACCOUNT TAKEOVER IN THE YUNMAI SMART SCALE API

The Android and iOS API were discovered to not implement any authorization checks while adding or deleting ‘family member’ accounts to/from other accounts.

https://fortbridge.co.uk/research/mass-account-takeover-yunmai/

Why is authorization so hard to get right?

Permission-based authorization on an API endpoint

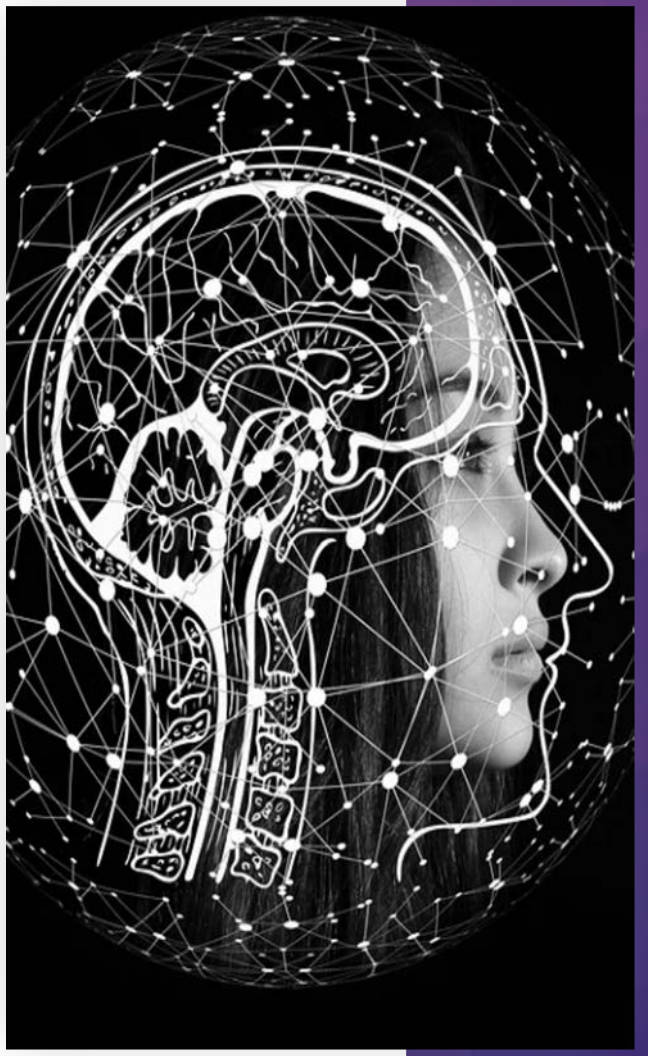
```
1  @PreAuthorize("hasPermission('ADD_FAMILY_MEMBER')")
2  public void addMember(long familyId, FamilyMember member) {
3    familyData.addMember(familyId, member);
4  };
```


BINGO
BINGO
77
46

Question One:

Which OWASP API Top 10 issue have we solved with this permission check?

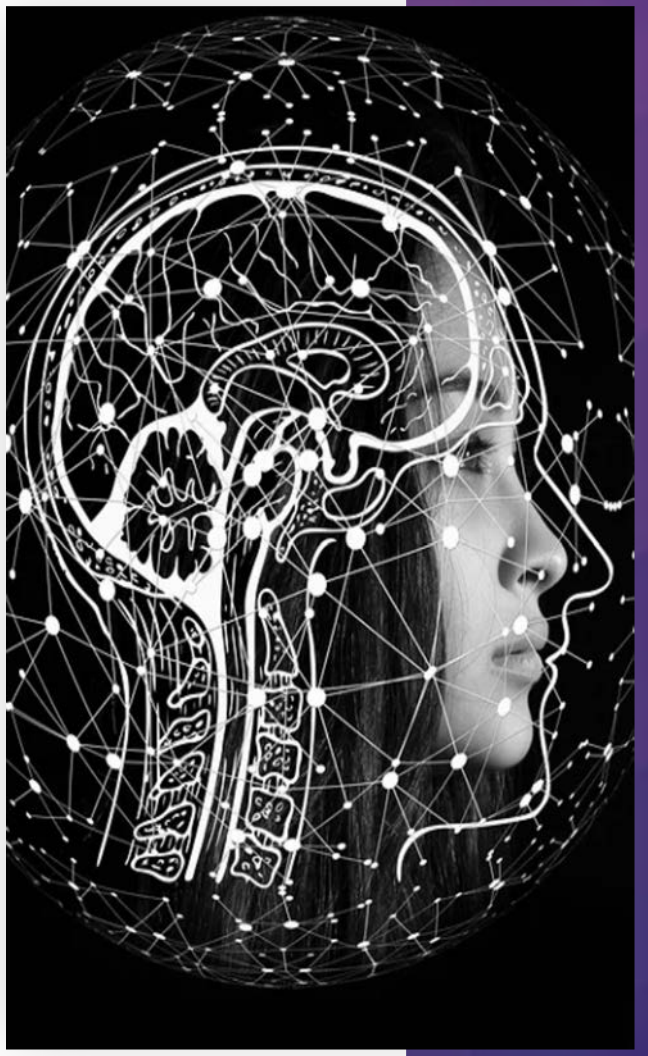
1. Broken Object Level Authorization (BOLA)?
2. Broken Authentication?
3. Broken Function Level Authorization (BFLA)?
4. Broken Object Property Level Authorization (BOPLA)?



Question One:

Which OWASP API Top 10 issue have we solved with this permission check?

Broken Object Level Authorization (BOLA)	29%
Broken Authentication	24%
Broken Function Level Authorization (BFLA)	42%
Broken Object Property Level Authorization (BOPLA)	4%



An example of Broken Object-Level Authorization

```
1  @PreAuthorize("hasPermission('ADD_FAMILY_MEMBER')")
2  public void addMember(long familyId, FamilyMember member) {
3    familyData.addMember(familyId, member);
4  };
```

Adding a family member

```
1 POST /family/1/member HTTP/1.1
2 { name: … }
```

This adds a new member to your family

Adding a family member

```
1 POST /family/7/member HTTP/1.1
2 { name: … }
```

This adds a new member to someone else's family

So we covered BFLA and BOLA, but what about BOPLA?


DEMO TIME

Function-level authorization

Is this entity allowed to perform **this particular operation**?

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

MASS ACCOUNT TAKEOVER IN THE YUNMAI SMART SCALE API

Account takeover through ‘forgot password’ functionality.

The victim will get an email with a unique 6 digit code that allows to reset the password.

https://fortbridge.co.uk/research/mass-account-takeover-yunmai/

Unrestricted access to brute-forceable 'forgot password' functionality is a critical security failure

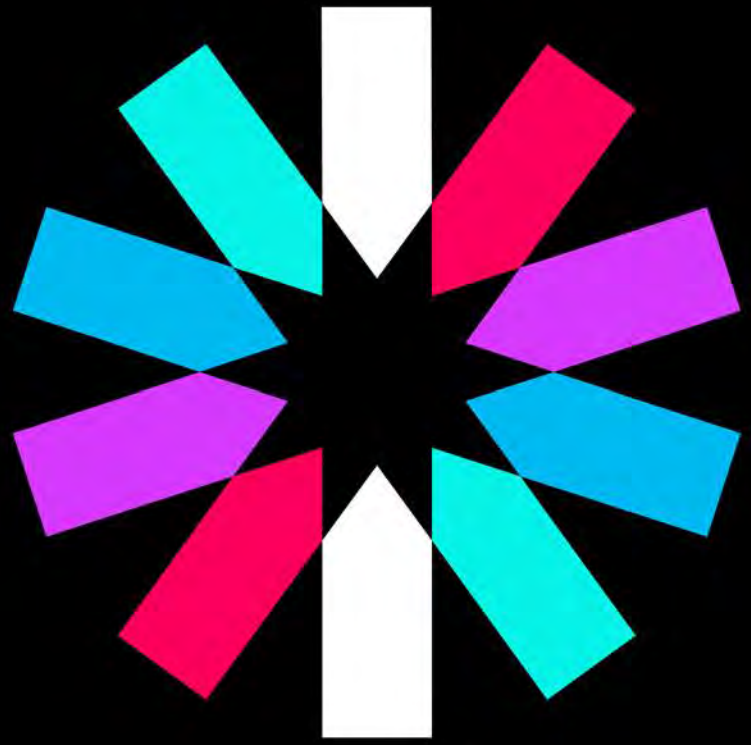
An account enumeration vulnerability significantly amplifies authentication threats

How do you prevent brute force attacks?


DEMO TIME

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

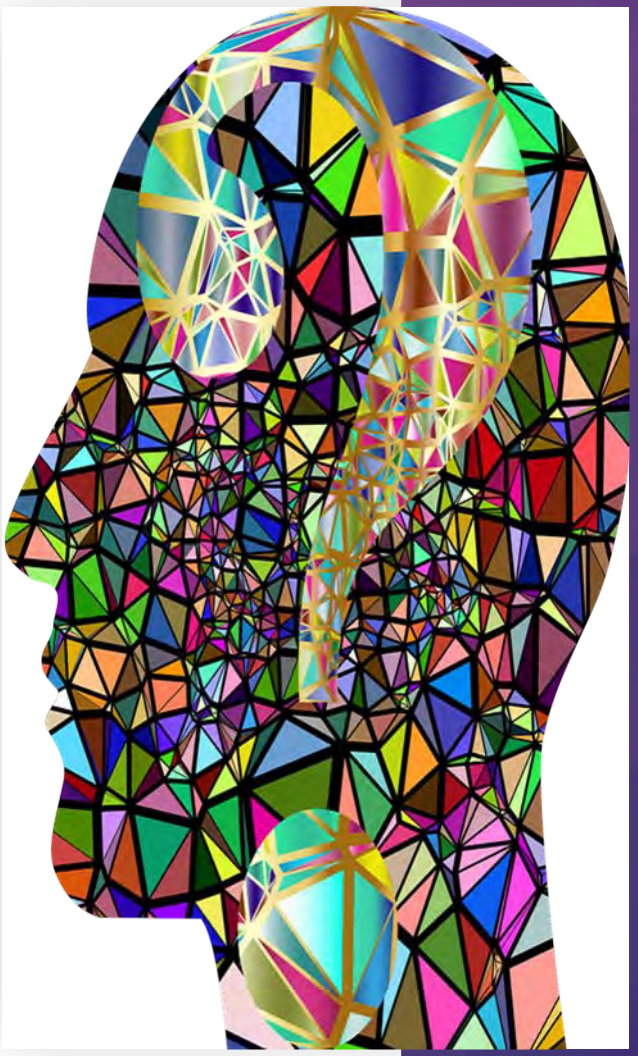


JUNIT

Question Two:

By default, JSON Web Tokens (JWT), are

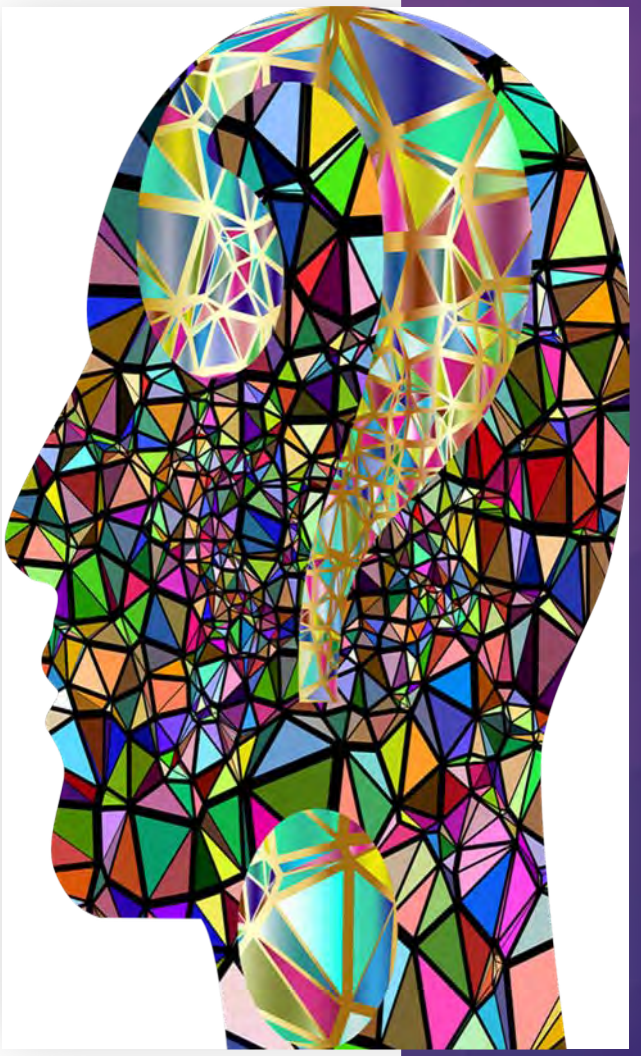
1. Base64 encoded
2. Signed
3. Encrypted
4. Not sure



Question Two:

By default, JSON Web Tokens (JWT), are

Base64 encoded	64%
Signed	55%
Encrypted	18%
Not sure	0%



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
J1c2VyIjoiZTcyZDFhMjZmNDBlNGU4Nzk5NjciL
CJ0ZW5hbnQiOiJkOGNmM2ZhMzAxYTM0Yzk2ODUw
MmE3MDUxYmZkYzBhOCIsImlhdCI6MTYyMDE5MjY
0NDkxNCwiZXhwIjoxNjIwMTk2MjQ0OTE0fQ.bnd
YFgq1sHD-
vH8h1lARD8M0uZgoALThQu7CURkuSVs
```

The base64-encoded header and payload, along with the signature

The signature is crucial to ensure the integrity of the header and payload

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

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

PAYLOAD: DATA

```
{
  "user": "e72d1a26f40e4e879967",
  "tenant": "d8cf3fa301a34c968502a7051bfdc0a8",
  "iat": 1620192644914,
  "exp": 1620196244914
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  SuperSecretHMACKey
) ☐ secret base64 encoded
```

Internet Engineering Task Force (IETF) M. Jones
Request for Comments: 7515 Microsoft
Category: Standards Track J. Bradley
ISSN: 2070-1721 Ping Identity

Internet Engineering Task Force (IETF) M. Jones
Request for Comments: 7516 Microsoft
Category:
ISSN: 2

Internet Engineering Task Force (IETF) M. Jones
Request for Comments: 7519 Microsoft
Category: Standards Track J. Bradley
ISSN: 2070-1721 Ping Identity
N. Sakimura
NRI
May 2015

JSON Web Token (JWT)

Abstract

JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted.

Internet Engineering Task Force (IETF)
Request for Comments: 8725
BCP: 225
Updates: 7519
Category: Best Current Practice
ISSN: 2070-1721

Y. Sheffer
Intuit
D. Hardt

M. Jones
Microsoft

February 2020

JSON Web Token Best Current Practices

Abstract

JSON Web Tokens, also known as JWTs, are URL-safe JSON-based security tokens that contain a set of claims that can be signed and/or encrypted. JWTs are being widely used and deployed as a simple security token format in numerous protocols and applications, both in the area of digital identity and in other application areas. This Best Current Practices document updates RFC 7519 to provide actionable guidance leading to secure implementation and deployment of JWTs.

alg: none

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "PS256",
  "typ": "JWT",
  "kid": "Ae42SFaYAECQQ"
}
```

PAYLOAD: DATA

```
{
    "file_id": "d8cf3fa301a34c968502a7051bfdc0a8",
    "sub": "5e4fd699d6b84cd8b1bee5f0428c0918",
    "iss": "https://sts.restograde.com",
    "aud": "https://files.restograde.com",
    "iat": 1521314123,
    "exp": 1621314123
}
```

VERIFY SIGNATURE

```
RSAPSSSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  yxDgWk4VRLF4mE63BpwVNFAcRCZCiU
  ATZm
  VEuby8E99kaThn98oQIDAQAB
  -----END PUBLIC KEY-----
  ,
  kIF89+6u7zNilOE1iSZ9kICE1iIs89
  9+ML
  87akDERXFlPLhrhel+NlG+TPP288sE
  J9r21nE4eTOOXj
  -----END RSA PRIVATE KEY-----
)
```

The application uses signed JWTs and rejects JWTs with invalid signatures

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "none",
  "typ": "JWT",
  "kid": "Ae42SFaYAECQQ"
}
```

PAYLOAD: DATA

```
{
    "file_id": "502a7051bfdc0a8d8cf3fa301a34c968",
    "sub": "5e4fd699d6b84cd8b1bee5f0428c0918",
    "iss": "https://sts.restograde.com",
    "aud": "https://files.restograde.com",
    "iat": 1521314123,
    "exp": 1621314123
}
```

By using *none* as the signature, the attacker can create a JWT that is not signed

An unsigned JWT can hold arbitrary data, giving access to arbitrary files on this system

Apache Pulsar bug allowed account takeovers in certain configurations

Ben Dickson 02 June 2021 at 11:43 UTC
Updated: 02 June 2021 at 14:32 UTC

GitHub | Open Source Software | Secure Development

Software maintainers downplay real-world impact of JWT vulnerability

4 ...on/src/main/java/org/apache/pulsar/broker/authentication/AuthenticationProviderToken.java

@@ -172,9 +172,7 @@ private static String validateToken(final String token) throws AuthenticationExc

```
172         @SuppressWarnings("unchecked")
173         private Jwt<?, Claims> authenticateToken(final
    String token) throws AuthenticationException {
174             try {
175 -               Jwt<?, Claims> jwt = Jwts.parser()
176 -                       .setSigningKey(validationKey)
177 -                       .parse(token);
178
179                 if (audienceClaim != null) {
180                     Object object =
    jwt.getBody().get(audienceClaim);
```

```
172         @SuppressWarnings("unchecked")
173         private Jwt<?, Claims> authenticateToken(final
    String token) throws AuthenticationException {
174             try {
175 +               Jwt<?, Claims> jwt =
    Jwts.parserBuilder().setSigningKey(validationKey).build()
    .parseClaimsJws(token);
176
177                 if (audienceClaim != null) {
178                     Object object =
    jwt.getBody().get(audienceClaim);
```

https://github.com/apache/pulsar/pull/9172/commits/94247dac93542bbcb45fb7104f7204363aad7441

JWT security is hard


DEMO TIME

BLOG

7 Ways to Avoid JWT Security Pitfalls

December 22, 2021 | by Dr Philippe de Ryck

Blog, Popular

Dec 22nd 2021. Author: Dr. Philippe de Ryck, Pragmatic Web Security,

Like them or hate them, JSON Web Tokens (JWT) are everywhere.

https://42crunch.com/7-ways-to-avoid-jwt-pitfalls/

Learning more: How to Best Leverage JWTs for API security

WEBINAR

How to Best Leverage JWTs for API Security

December 10, 2020

JSON Web tokens (JWTs) are used massively in API-based applications as access tokens or to transport information across services. Unfortunately, JWT standards are quite complex and it's very easy to get the implementation wrong. As a result, data breaches and API vulnerabilities due to poor JWT implementation, token leakage, and lack of proper validation remain widespread.

This webinar focuses on JWT best practices, most common JWT attacks and how the 42Crunch API Security Platform leverages OpenAPI (Swagger) Specification extensions to prevent them.

In this webinar you will learn:

- How to best use JWTs for API Security
- Most common attacks as illustrated in RFC 8725
- How 42Crunch can help you protect your APIs from those attacks

Speaker

Isabelle Mauny

Field CTO and Co-founder

Dmitry Sotnikov

CPO and Curator of APIsecurity.io

https://42crunch.com/webinar-json-web-tokens-api-security/

1	Broken object level authorization
2	Broken authentication
3	Broken object property-level authorization
4	Unrestricted resource consumption
5	Broken function level authorization
6	Unrestricted access to sensitive business flows
7	Server-side request forgery
8	Security misconfiguration
9	Improper inventory management
10	Unsafe consumption of APIs

Learning more: APIs and Request Forgery (CSRF & SSRF)

WEBINAR

Defending APIs with Jim Manico

November 10, 2022 | 9am PST | 5pm BST

Episode 1: Request Forgery on the Web - CSRF & SSRF

In this first episode Jim and Colin will discuss request forgery and how to prevent it. This technical talk is intended for the software developer who needs to build secure web applications and APIs. it will cover the two variants of request forgery — client-side (CSRF) and server-side (SSRF).

- CSRF is most widely associated with vulnerable web applications that trick a user in a client browser into submitting transactions they never intended to use in their current authenticated session. We will discuss historical CSRF attacks and investigate various well-proven defense strategies. **For API developers** we will investigate whether APIs are vulnerable to CSRF, and how to prevent it.
- SSRF attacks allow a malicious client to trick a vulnerable server into submitting requests to an unintended location, typically by submitting malformed URLs in payloads and relying on vulnerabilities in the URL parsing code. We will discuss prevention strategies and examine some well-known examples. **For API developers**, we will investigate ways in which SSRF can be directed at vulnerable APIs and examine a few recent API breaches and the latest research.

Colin Domoney

Developer Advocate & API Security Researcher

42Crunch

Jim Manico

CEO

Manicode Security

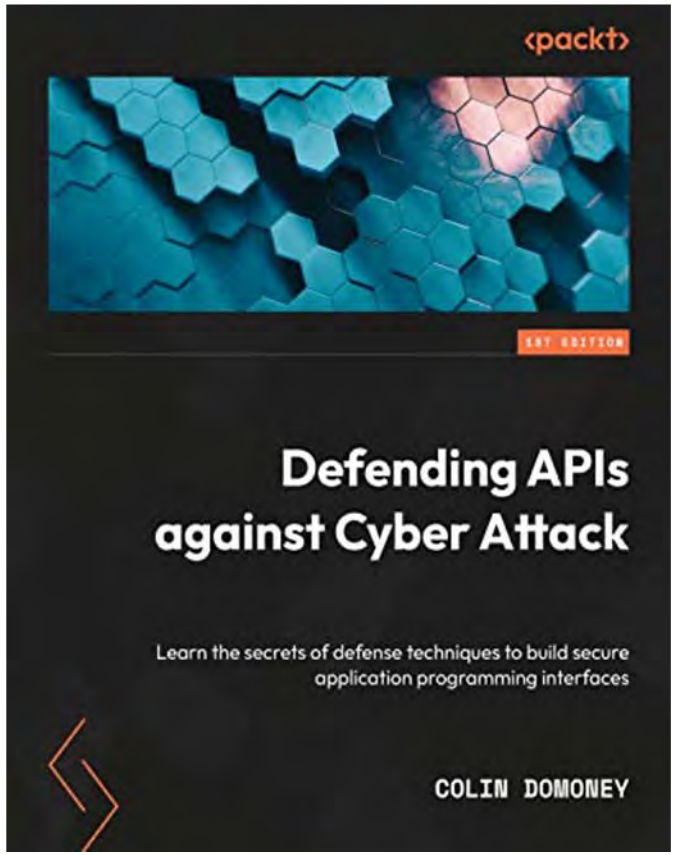
https://42crunch.com/defending-apis-with-jim-manico-episode-1/

Learning more

#1 API Security Newsletter
APISecurity.io

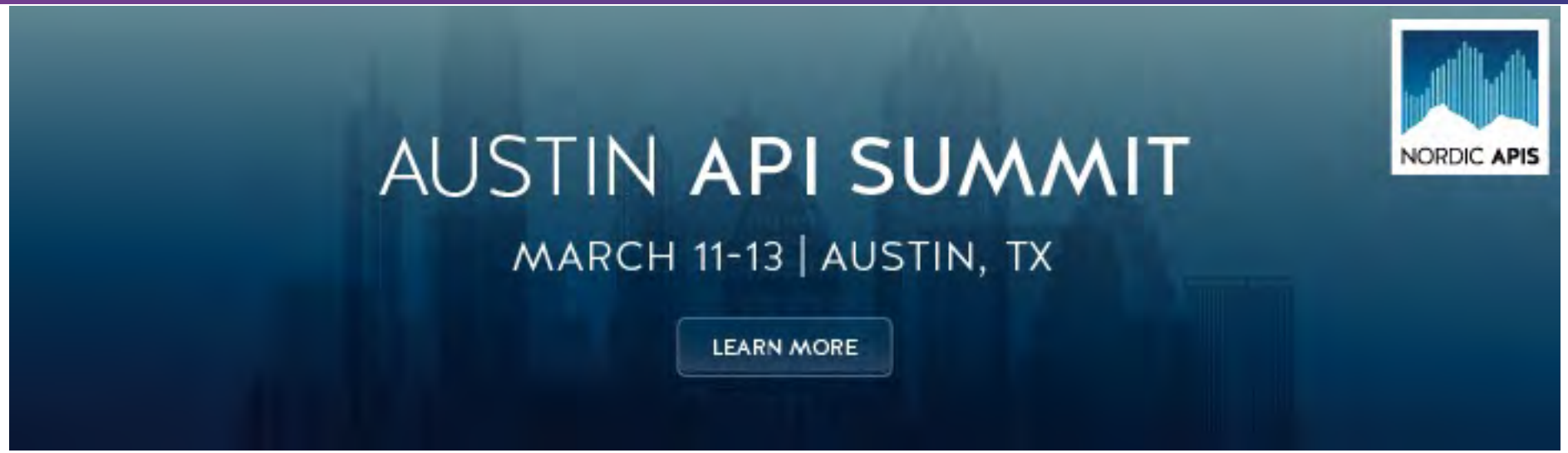
https://apisecurity.io/

“Defending APIs against Cyber Attack”
by Colin Domoney



https://amzn.to/3fHp8Mz

Events



Keynote:
You've had an API breach, Now what?

Workshop:
API Security Done Right – Understanding the Threat and Best Practices for Secure API Development

AXEL GRÖSSE

HESHAAM ATTAR

DEFENDING APIs WORKSHOP
London
April 18, 2024

World-renowned expert-led half-day practical workshop including:

- *Introduction to API Security*
- *Understand the OWASP API Top 10*
- *API Security Testing*
- *API Runtime Protection*
- *Latest vulnerabilities and much more*

Lunch & Drinks provided

Register: https://42crunch.com/api-security-workshop-defending-apis/

Audience Question One:

As organizations increasingly rely on microservices architectures, what new considerations arise for API security in distributed systems?

Audience Question Two:

Can BOLA and IDOR be considered the same or are there small differences?

Audience Question Three:

What part of a system should enforce the additionalProperties being set as false?

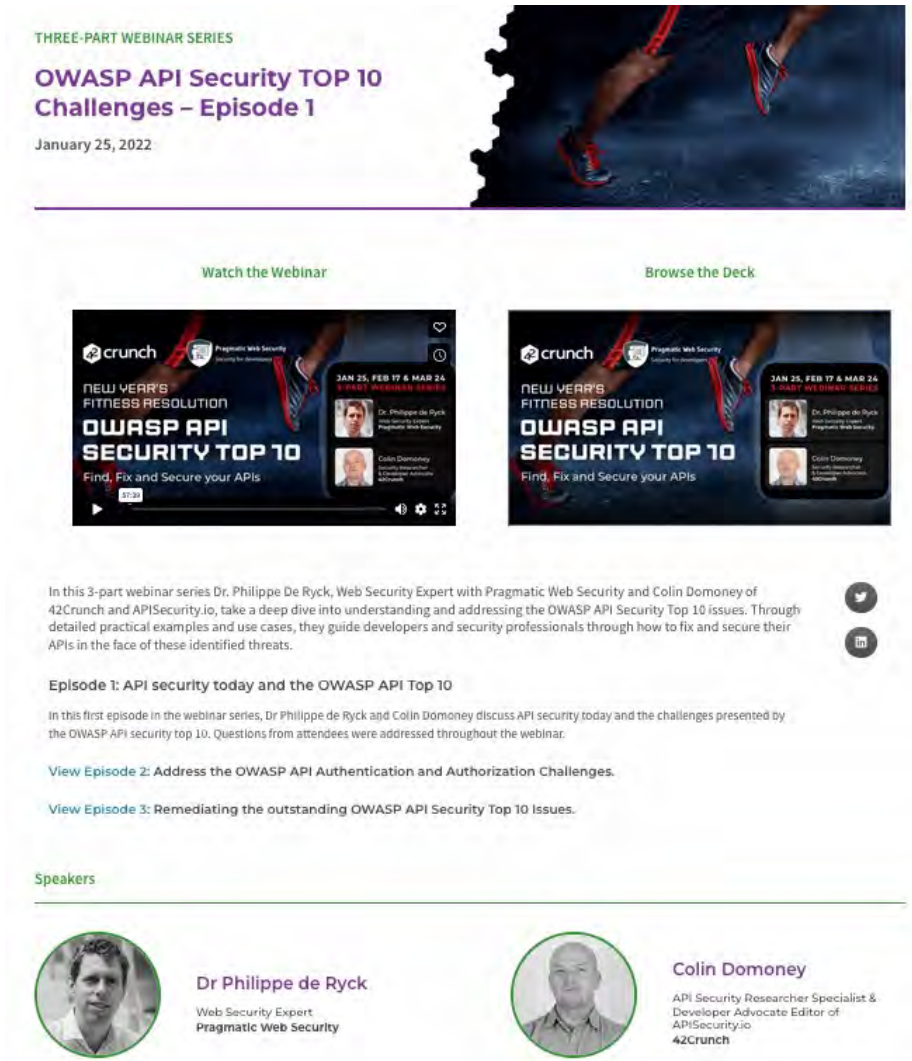
Audience Question Four:

Is there any automated tool you would recommend to test the API security in our organization?

Audience Question Five:

How does the architecture look with placing the firewall between the API gateway and the API?

Learning more: OWASP API Security Top 10: 2019



OWASP API Security Top 10 Challenges - 3 Part Webinar Series

Episode 1: API security today and the OWASP API Top 10
In this first episode in the webinar series, Dr Philippe de Ryck and Colin Domoney discuss API security today and the challenges presented by the OWASP API security top 10. Questions from attendees were addressed throughout the webinar.
https://42crunch.com/owasp-api-security-top-10-webinar-series-recordings/

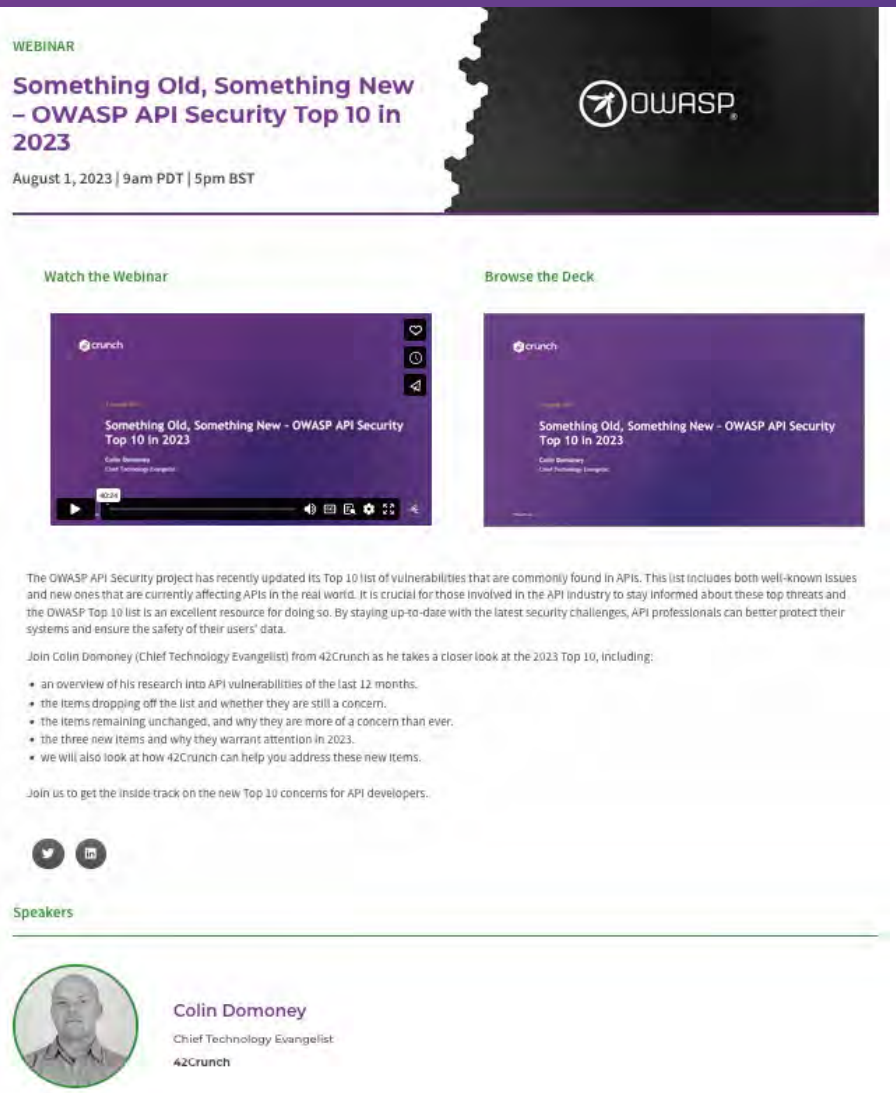
Episode 2: Address the OWASP API Authentication and Authorization Challenges
In this second episode in the webinar series, Dr Philippe de Ryck and Colin Domoney address one-by-one, the OWASP API Authentication and Authorization Challenges. Through detailed practical examples and use cases, they guided developers and security professionals through how to fix and secure their APIs in the face of these identified threats.
https://42crunch.com/owasp-api-security-top-10-webinar-series-recordings-2/

Episode 3: Remediating the outstanding OWASP API Security Top 10 Issues.
Learn as Dr Philippe De Ryck, Web Security Expert with Pragmatic Web Security and Colin Domoney of 42Crunch and APISecurity.io, address one-by-one, the remaining 5 OWASP API Challenges:

- Issue 4: Lack of resources & rate limiting.
- Issue 7: Security misconfiguration.
- Issue 8: Injection.
- Issue 9: Improper assets management.
- Issue 10: Insufficient logging and monitoring.

https://42crunch.com/owasp-api-security-top-10-webinar-series-recordings-3/

Learning more: OWASP API Security Top 10: 2023



Something Old, Something New - OWASP API Security Top 10 Challenges in 2023

The OWASP API Security project has recently updated its Top 10 list of vulnerabilities that are commonly found in APIs. This list includes both well-known issues and new ones that are currently affecting APIs in the real world. It is crucial for those involved in the API industry to stay informed about these top threats and the OWASP Top 10 list is an excellent resource for doing so. By staying up-to-date with the latest security challenges, API professionals can better protect their systems and ensure the safety of their users' data.

Join Colin Domoney (Chief Technology Evangelist) from 42Crunch as he takes a closer look at the 2023 Top 10, including:

- an overview of his research into API vulnerabilities of the last 12 months.
- the items dropping off the list and whether they are still a concern.
- the items remaining unchanged, and why they are more of a concern than ever.
- the three new items and why they warrant attention in 2023.
- we will also look at how 42Crunch can help you address these new items.

https://42crunch.com/something-old-something-new-owasp-api-security-top-10-2023/

February 6, 2024

Top Things You Need to Know About API Security

Dr. Philippe De Ryck & Isabelle Mauny