1 August 2023

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

Colin Domoney

Chief Technology Evangelist

Introduction

About the Speakers

Colin Domoney

Chief Technology Evangelist

Editor of APISecurity.io
CyberProof, CA, Veracode, Deutsche Bank

Housekeeping Rules

- All attendees muted
- Questions via Q&A
- Recording will be shared
- Polling questions

Agenda

- Overview of research into API vulnerabilities of the last 12 months
- OWASP Top 10 methodology
- Two items dropping off the Top 10
- Items that stay the same and why they remain a concern
- Three new items on the Top 10
    - How to address them
    - How 42Crunch can help
- Questions and Answers

OWASP and the Top 10

The Top 10 in 2019 and now

My own research via APISecurity.io

APISecurity.io

https://apisecurity.io/

25 May, 23
Issue 220: API flaw in Booking.com, apps leaking sensitive API data, API security testing checklist

12 May, 23
Issue 219: Money Lover app exposes user data, most web API flaws missed by standard testing

5 May, 23
Issue 218: Three Argo CD API exploits, distributed identity for modern API security

28 April, 23
Issue 217: Wordle API exposes answers, Twitter API breach updates, AWS exposed dangerous API

5 March, 23
Issue 215: API flaws in Lego marketplace, API style guides, 42Crunch joins MISA

9 February, 23
Issue 214: Google Cloud's four pillars of API security, Cerbos for API permissions, attacking predictable GUIDs

26 January, 23
Issue 213: Supply chain vulnerability in IBM Cloud, hardcoded API keys in Algolia portal, JSON-based SQL attacks

The OWASP Top 10 - process and scoring methodology

Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impacts
API Specific	Easy: **3**	Widespread **3**	Easy **3**	Severe **3**	Business Specific
API Specific	Average: **2**	Common **2**	Average **2**	Moderate **2**	Business Specific
API Specific	Difficult: **1**	Difficult **1**	Difficult **1**	Minor **1**	Business Specific

https://owasp.org/API-Security/editions/2023/en/0x10-api-security-risks/

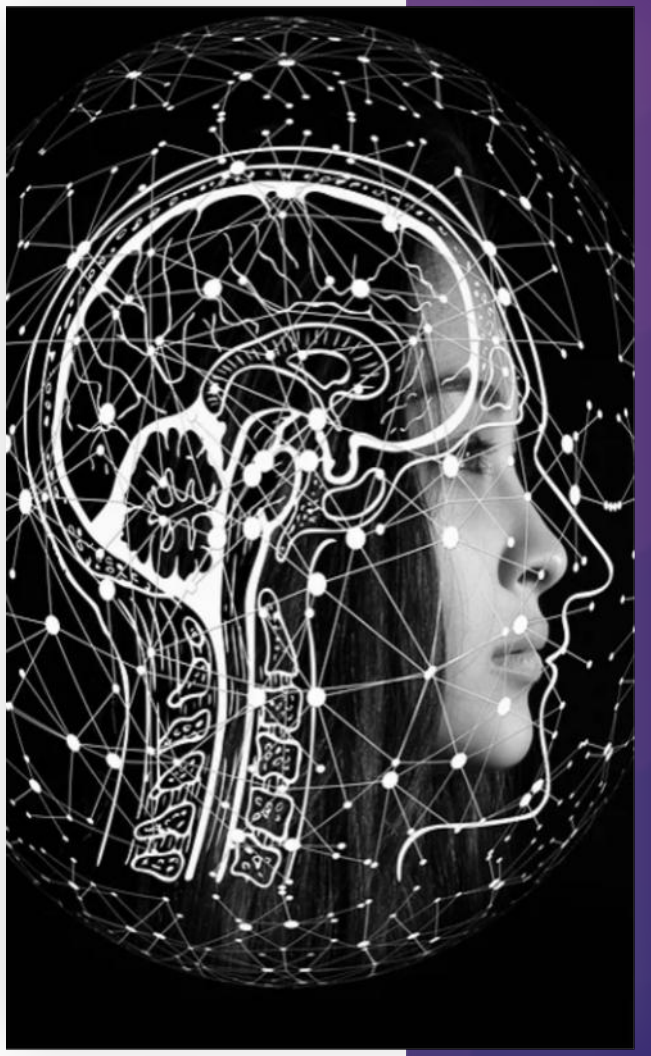
OWASP API Security Top 10 — Then and Now

2019	#	2023
API1:2019 - Broken Object Level Authorization	1	API1:2023 - Broken Object Level Authorization
API2:2019 - Broken User Authentication	2	API2:2023 - Broken Authentication
API3:2019 - Excessive Data Exposure	3	API3:2023 - Broken Object Property Level Authorization
API4:2019 - Lack of Resources & Rate Limiting	4	API4:2023 - Unrestricted Resource Consumption
API5:2019 - Broken Function Level Authorization	5	API5:2023 - Broken Function Level Authorization
API6:2019 - Mass Assignment	6	API6:2023 - Unrestricted Access to Sensitive Business Flows
API7:2019 - Security Misconfiguration	7	API7:2023 - Server Side Request Forgery
API8:2019 - Injection	8	API8:2023 - Security Misconfiguration
API9:2019 - Improper Assets Management	9	API9:2023 - Improper Inventory Management
API10:2019 - Insufficient Logging & Monitoring	10	API10:2023 - Unsafe Consumption of APIs

Question One:

Which one of the OWASP Top 10 causes your organization the most amount of pain?

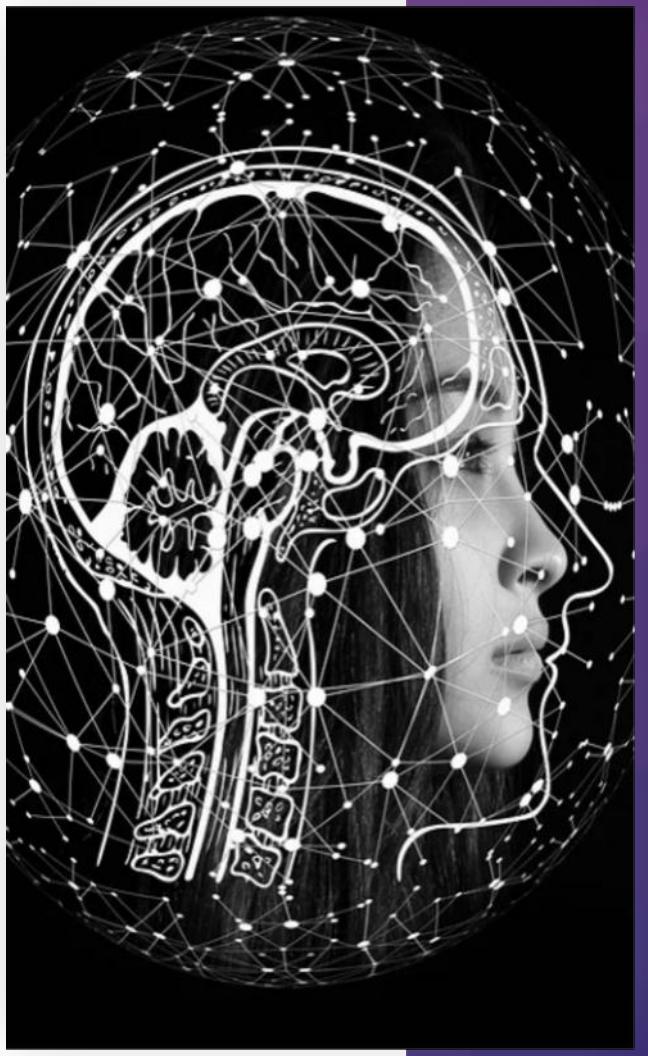
1. Broken Object/Function Level Authorization
2. Broken Authentication
3. Broken Object Property Level Authorization
4. Unrestricted Resource Consumption
5. Unrestricted Access to Sensitive Business Flows



Question One:

Which one of the OWASP Top 10 causes your organization the most amount of pain?

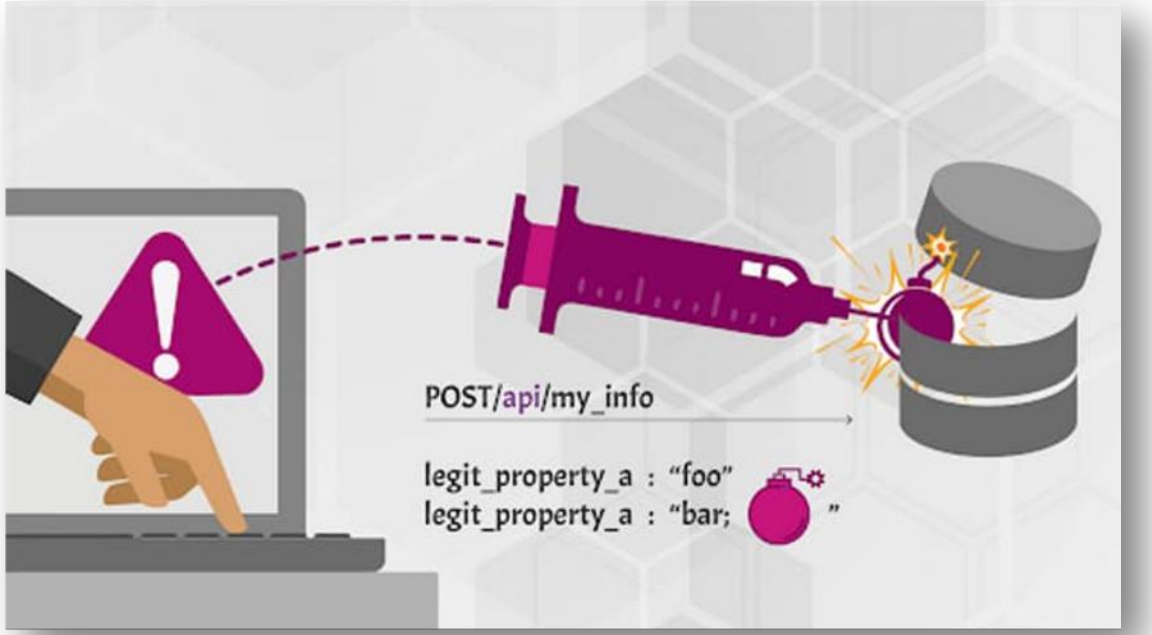
Broken Object/Function Level Authorization	24%
Broken Authentication	34%
Broken Object Property Level Authorization	15%
Unrestricted Resource Consumption	15%
Unrestricted Access to Sensitive Business Flows	12%



Dropping out the Top 10 — API8:2019 — Injection

- Injection attacks affect **all** software systems, not only APIs
- Injection attacks are still **very prevalent** and affect APIs frequently
- Remediation advice as per OWASP Top 10 still applies (nothing API specific)

- 42Crunch offers protection for this category (relies on full specification of input data)



https://apisecurity.io/owasp-api-security-top-10/api8-injection/

Lack of injection in 2023 API10 #86

Open cyn8 opened this issue on Mar 8 · 19 comments

https://github.com/OWASP/API-Security/issues/86

Issue 200: Injection vulnerability in BitBucket, OAuth2 exploitation, and 200th issue prize giveaways

September 1, 2022

Dropping out the Top 10 — API10:2019 — Insufficient logging and monitoring

- Logging and monitoring affect **all** software systems, not only APIs
- Logging and monitoring are seldom reported in API breaches
- Typically they are a symptom rather than a cause
- Remediation advice as per OWASP Top 10 still applies (nothing API specific)

https://apisecurity.io/owasp-api-security-top-10/api10-insufficient-logging-and-monitoring/

What's new in 2023

New entry — API6:2023 - Unrestricted Access to Sensitive Business Flows

- This entails using an API in a way in which it was not designed or abusing the underlying business flow or logic
- In a single word – BOTS
- Becoming the most common attack vector

Threat agents/Attack vectors	Security Weakness	Impacts
API Specific : Exploitability **Easy**	Prevalence **Widespread** : Detectability **Average**	Technical **Moderate** : Business Specific
Exploitation usually involves understanding the business model backed by the API, finding sensitive business flows, and automating access to these flows, causing harm to the business.	Lack of a holistic view of the API in order to fully support business requirements tends to contribute to the prevalence of this issue. Attackers manually identify what resources (e.g. endpoints) are involved in the target workflow and how they work together. If mitigation mechanisms are already in place, attackers need to find a way to bypass them.	In general technical impact is not expected. Exploitation might hurt the business in different ways, for example: prevent legitimate users from purchasing a product, or lead to inflation in the internal economy of a game.

https://owasp.org/API-Security/editions/2023/en/0xa6-unrestricted-access-to-sensitive-business-flows/

New entry — API6:2023 - Unrestricted Access to Sensitive Business Flows

How To Prevent

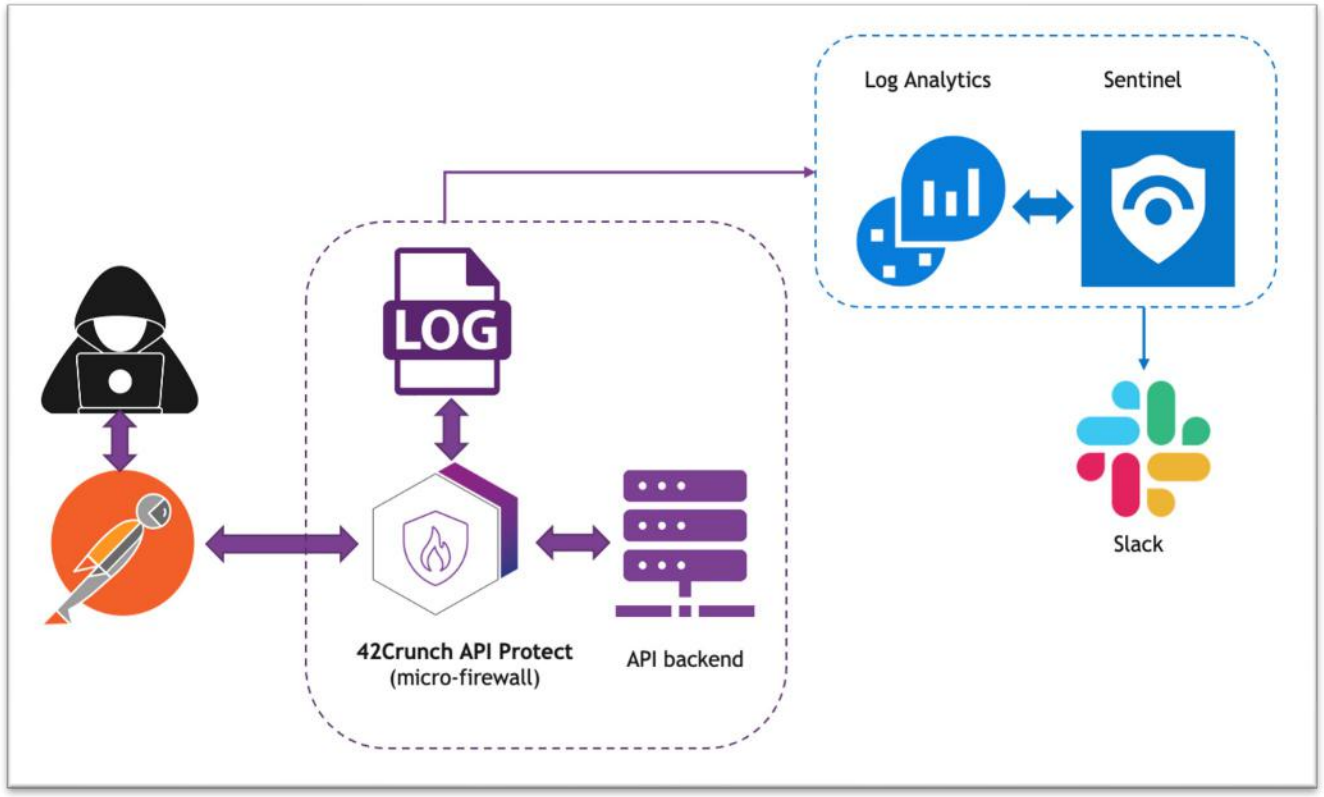
The mitigation planning should be done in two layers:

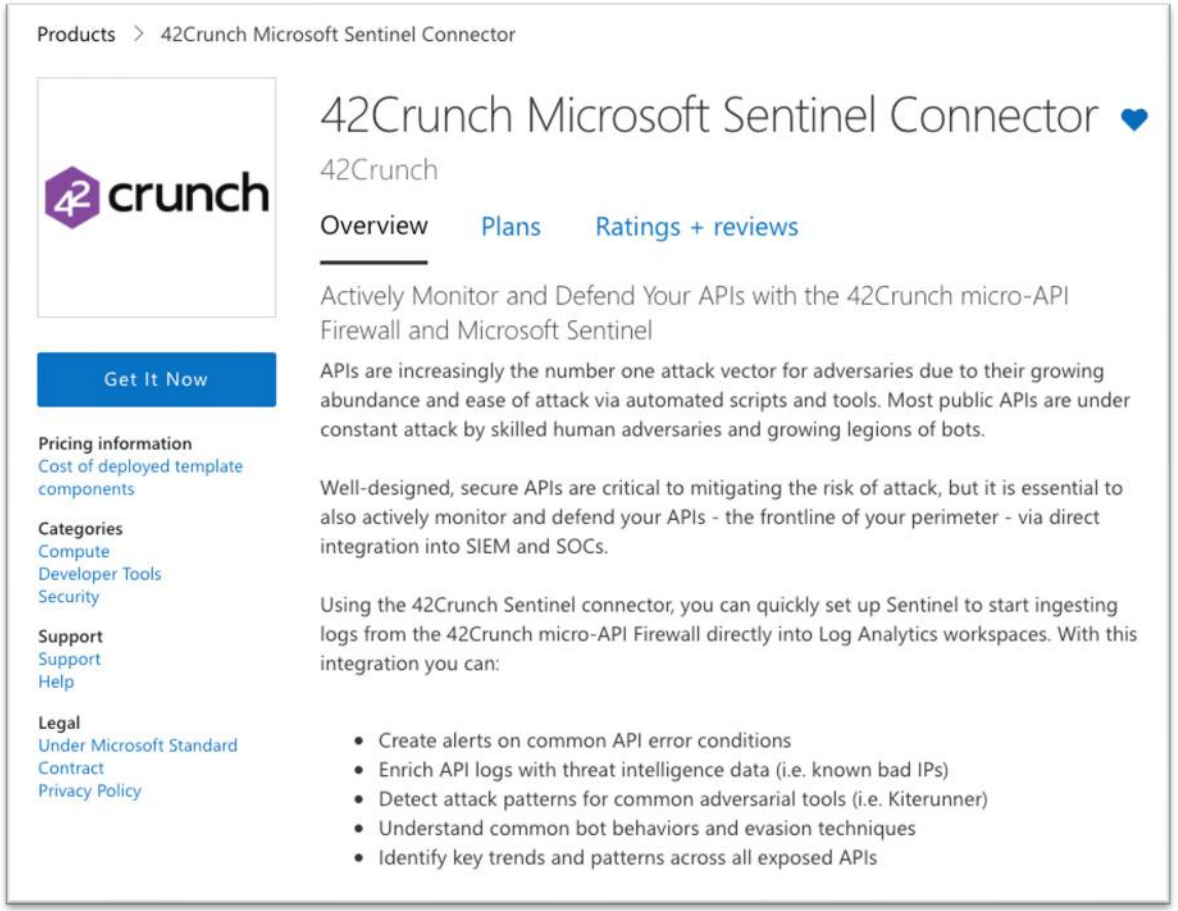
- **Business** - identify the business flows that might harm the business if they are excessively used.
- **Engineering** - choose the right protection mechanisms to mitigate the business risk.

Common methods include:

- **Device fingerprinting**: denying service to unexpected client devices (e.g, headless browsers) tends to make threat actors use more sophisticated solutions, thus more costly for them
- **Human detection**: using either a captcha or more advanced biometric solutions (e.g., typing patterns)
- **Non-human patterns**: analyse the user flow to detect non-human patterns (e.g., the user accessed the "add to cart" and "complete purchase" functions in less than one second)
- Consider blocking IP addresses of Tor exit nodes and well-known proxies

How 42Crunch helps API6:2023 - Unrestricted Access to Sensitive Business Flows





https://42crunch.com/protect-your-apis-with-microsoft-azure-sentinel-and-42crunch-platforms/

https://42crunch.com/actively-monitor-and-defend-your-apis-with-42crunch-and-the-azure-sentinel-platform/

New entry — API7:2023 - Server Side Request Forgery

- Server-Side Request Forgery (SSRF) flaws occur when an API is fetching a unattended URL-based resource because of missing validation of user-supplied URL
  - main target is internal resources
  - can also be external resources accessed taking advantage of trust relationships between server and targeted external resource
- Becoming a more common attack vector

Threat agents/Attack vectors	Security Weakness	Impacts
API Specific : Exploitability **Easy**	Prevalence **Common** : Detectability **Easy**	Technical **Moderate** : Business Specific
Exploitation requires the attacker to find an API endpoint that accesses a URI that's provided by the client. In general, basic SSRF (when the response is returned to the attacker), is easier to exploit than Blind SSRF in which the attacker has no feedback on whether or not the attack was successful.	Modern concepts in application development encourage developers to access URIs provided by the client. Lack of or improper validation of such URIs are common issues. Regular API requests and response analysis will be required to detect the issue. When the response is not returned (Blind SSRF) detecting the vulnerability requires more effort and creativity.	Successful exploitation might lead to internal services enumeration (e.g. port scanning), information disclosure, bypassing firewalls, or other security mechanisms. In some cases, it can lead to DoS or the server being used as a proxy to hide malicious activities.

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

https://owasp.org/API-Security/editions/2023/en/0xa7-server-side-request-forgery/

New entry — API7:2023 - Server Side Request Forgery

How To Prevent

- Isolate the resource fetching mechanism in your network: usually these features are aimed to retrieve remote resources and not internal ones.
- Whenever possible, use allow lists of:
  - Remote origins users are expected to download resources from (e.g. Google Drive, Gravatar, etc.)
  - URL schemes and ports
  - Accepted media types for a given functionality
- Disable HTTP redirections.
- Use a well-tested and maintained URL parser to avoid issues caused by URL parsing inconsistencies.
- Validate and sanitize all client-supplied input data.
- Do not send raw responses to clients.

How 42Crunch helps API7:2023 - Server Side Request Forgery

- Use 42Crunch contract enforcement
- Restrict allowable input values to known paths such as “uploads/”

```
"properties": {
  "_id": {
    "type": "string",
    "format": "uuid",
    "pattern": "^[0-9A-Fa-f]{8}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{12}$",
    "minLength": 36,
    "maxLength": 36,
    "x-42c-format": "o:uuid",
    "example": "a83a29f5-0d63-46f2-8f2e-44c2f1d2e07e"
  },
```

```
"PicturesList": {
  "type": "array",
  "x-examples": {
    "Example 1": [
      {
        "_id": "7ef9b6cb-ddd1-494f-89a0-04da62f18b47",
        "title": "Pensive Parakeet.jpg",
        "image_url": "uploads/91c0cf406c53af7f96b57561b5f526f9",
        "name": "automobile remove",
        "filename": "91c0cf406c53af7f96b57561b5f526f9",
        "description": "Oldie but goodie! a plastic and a lotion brimful hovercraft #TBT",
        "creator_id": "267c6dec-9d8f-4874-9eba-daa060e6324b",
        "money_made": 0,
        "likes": 0,
        "created_date": "2023-02-01T16:21:15.088Z"
      }
    ]
  },
```

https://docs.42crunch.com/latest/content/extras/protection_allowlist.htm

New entry — API10:2023 - Unsafe Consumption of APIs

- Developers tend to trust data received from third-party APIs more than user input
- APIs are part of a supply chain and need to be secured at every point

Threat agents/Attack vectors	Security Weakness	Impacts
API Specific : Exploitability **Easy**	Prevalence **Common** : Detectability **Average**	Technical **Severe** : Business Specific
Exploiting this issue requires attackers to identify and potentially compromise other APIs/services the target API integrated with. Usually, this information is not publicly available or the integrated API/service is not easily exploitable.	Developers tend to trust and not verify the endpoints that interact with external or third-party APIs, relying on weaker security requirements such as those regarding transport security, authentication/authorization, and input validation and sanitization. Attackers need to identify services the target API integrates with (data sources) and, eventually, compromise them.	The impact varies according to what the target API does with pulled data. Successful exploitation may lead to sensitive information exposure to unauthorized actors, many kinds of injections, or denial of service.

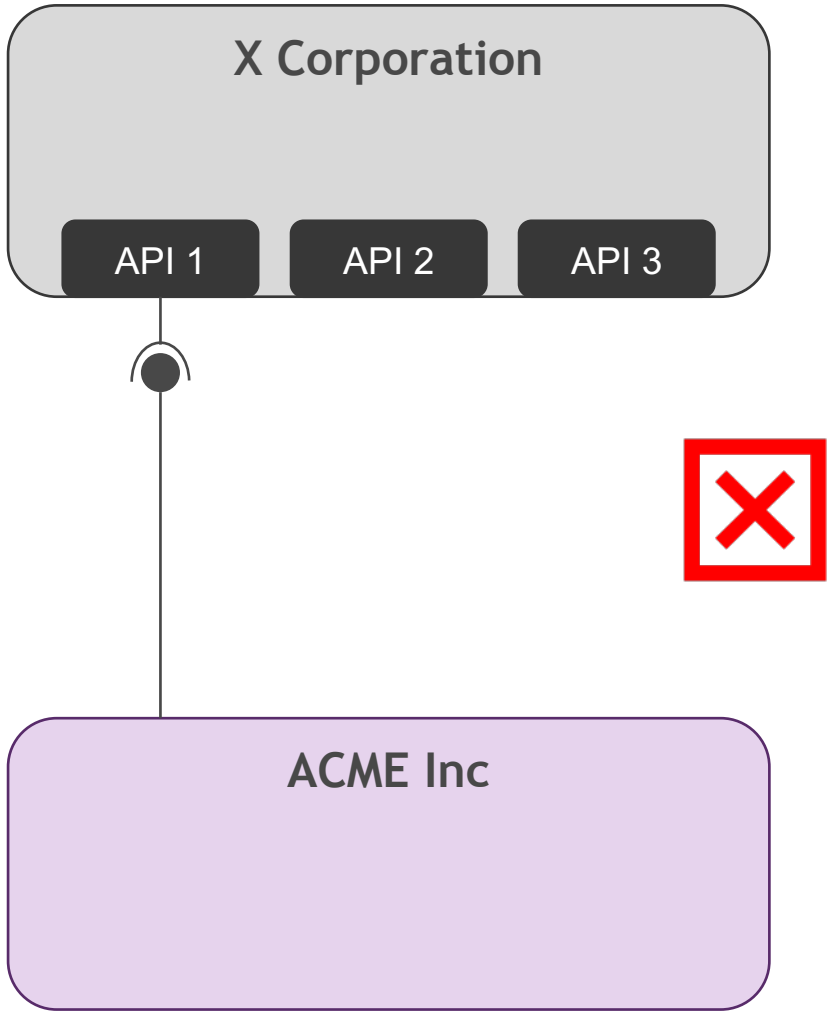
https://owasp.org/API-Security/editions/2023/en/0xaa-unsafe-consumption-of-apis/

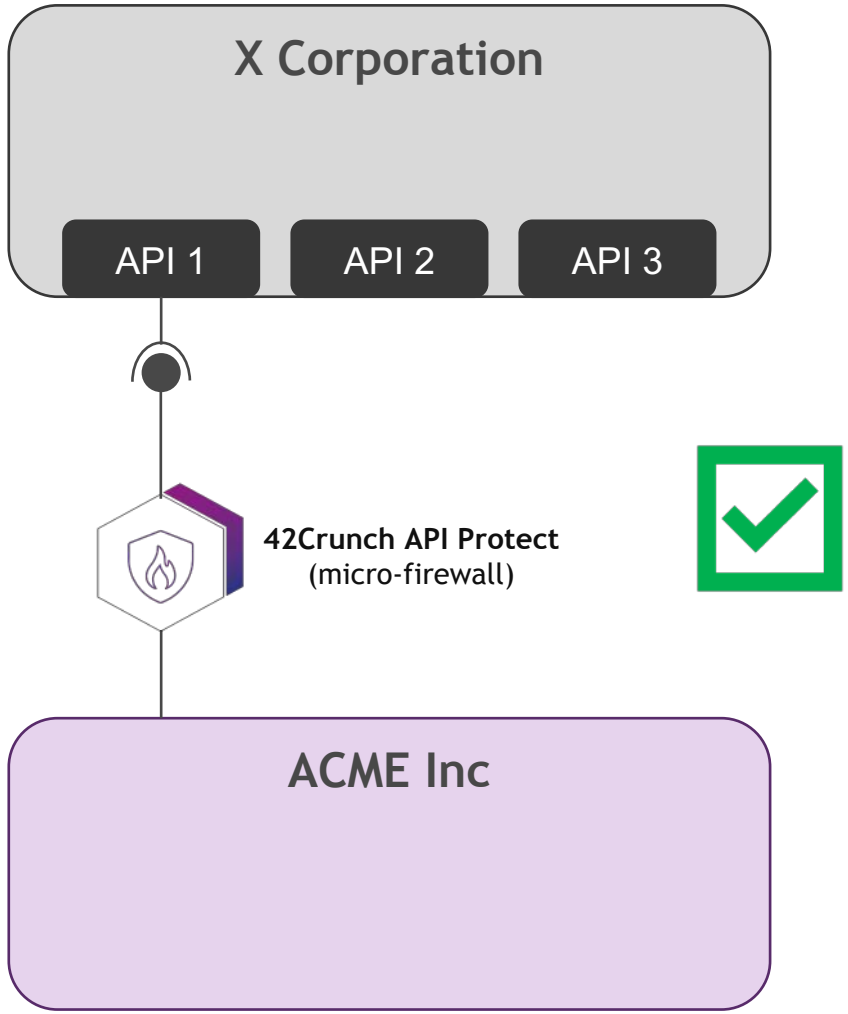
New entry — API10:2023 - Unsafe Consumption of APIs

How To Prevent

- When evaluating service providers, assess their API security posture.
- Ensure all API interactions happen over a secure communication channel (TLS).
- Always validate and properly sanitize data received from integrated APIs before using it.

How 42Crunch helps API10:2023 - Unsafe Consumption of APIs





Question Two:

How sure are you of your upstream API provider's security posture?

1. Audited them and satisfied
2. Asked about their process
3. No idea
4. A growing concern
5. Experienced an incident

Question Two:

How sure are you of your upstream API provider's security posture?

Audited them and satisfied	12%
Asked about their process	23%
No idea	49%
A growing concern	16%
Experienced an incident	0%

Tweaks in terminology and structure

- **API2:** The title was changed from “Broken User Authentication” to “Broken Authentication”
- **API4:** The title was changed from “Lack of Resources & Rate Limiting” to “Unrestricted Resource Consumption.”
- **API9:** The title was changed from “Improper Assets Management” to “Improper Inventory Management”

- **API3:** The 2019 categories “Excessive Data Exposure” (*read*) and “Mass Assignment” (*write*) are merged into “Broken Object Property Level Authorization”

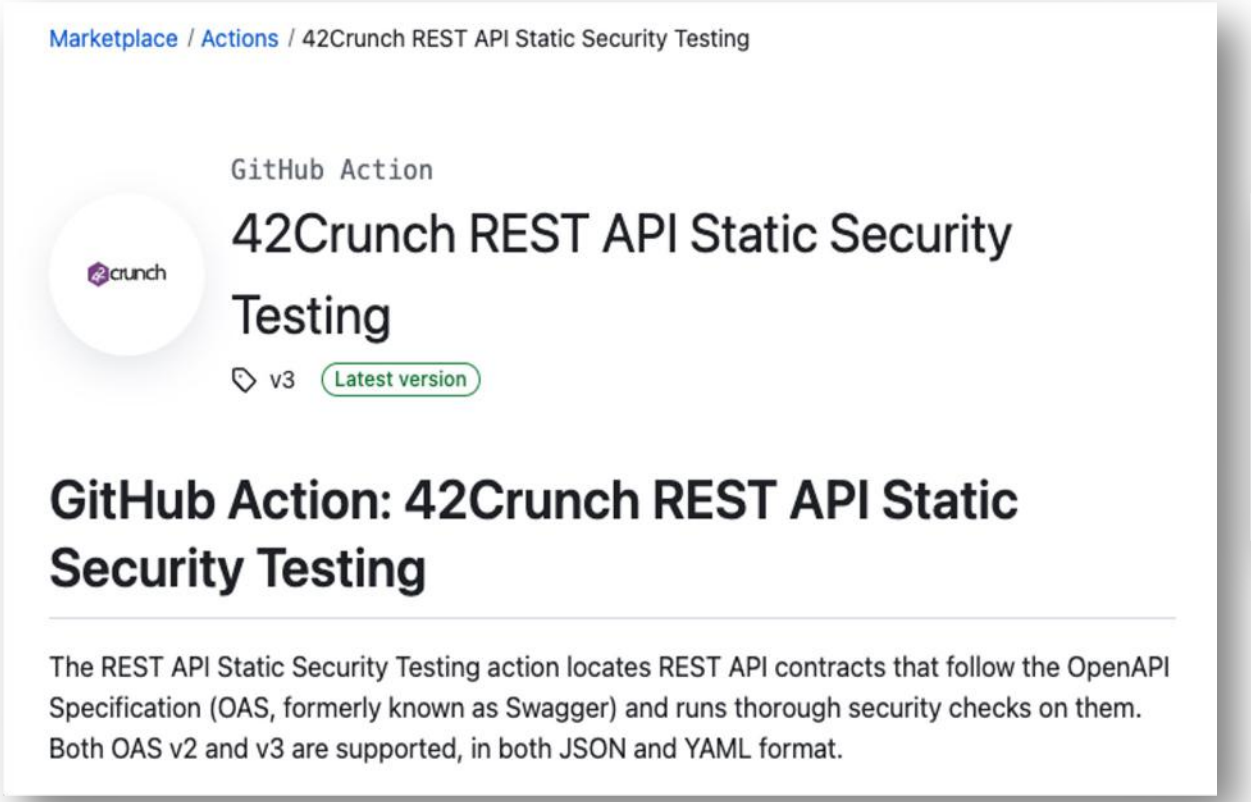
Learning more and next steps

Where to get more information

Learning more

- https://owasp.org/API-Security/editions/2023/en/0x11-t10/
- https://danaepp.com/owasp-api-security-top-10-upcoming-changes-you-need-to-know-about
- https://danaepp.com/exploiting-ssrf-in-an-api
- https://portswigger.net/web-security/ssrf
- https://42crunch.com/defending-apis-with-jim-manico-episode-1/

Using the 42Crunch extensions



https://github.com/marketplace/actions/42crunch-rest-api-static-security-testing

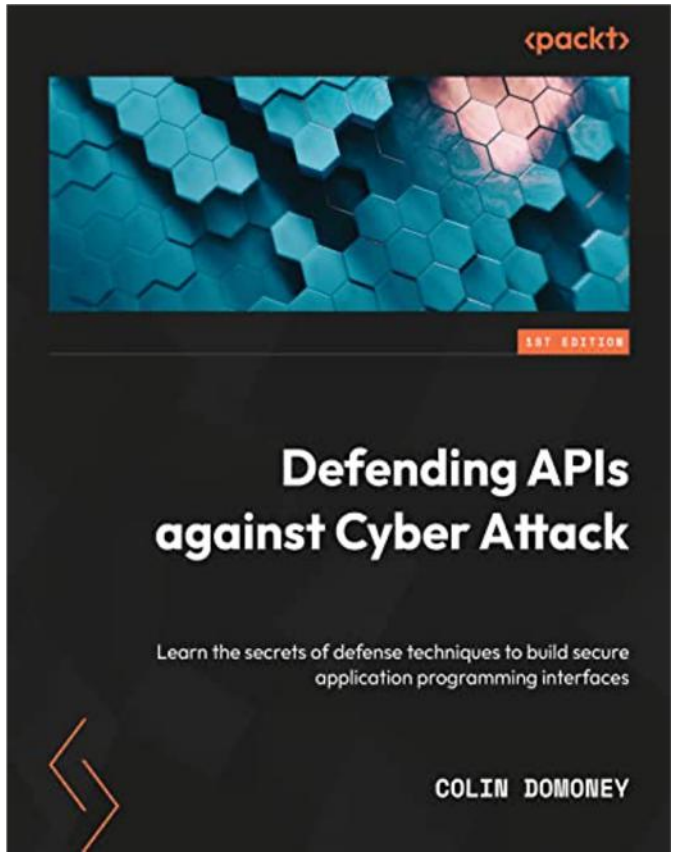
https://marketplace.visualstudio.com/items?itemName=42Crunch.vscode-openapi

Learning more

#1 API Security Newsletter
APISecurity.io

https://apisecurity.io/

“Defending APIs against Cyber Attack”
by Colin Domoney



https://amzn.to/3fHp8Mz

Audience Question One:

Are there any aspects you recommend someone who wants to start using automation?

Audience Question Two:

Does 42Crunch offer any security suggestions at API design beyond threat modeling?

Audience Question Three:

The approaches to manage authentication include JWT, Cookies, Session tokens. Internet is full of articles about them but not so much about what use cases each one of those fulfill. If you can share some resources that compare these approaches based on the use case that would be appreciated.

https://pragmaticwebsecurity.com/

Audience Question Four:

Does 42Crunch offer any solution on API discovery and inventory?

1 August 2023

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

Colin Domoney

Chief Technology Evangelist