RSAConference2022
San Francisco & Digital | June 6 – 9

TRANSFORM

SESSION ID: **HT-T09**

Malicious Uses of API Frameworks and Scanning Tools

Jason Kent

Hacker in Residence
Cequence Security

Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.

Who Am I

- Jason Kent
- Jason.kent@Cequence.ai
- linkedin.com/in/n0handle

- 20+ Years of Application Security focus
- Garage Door Opener API
- Kasa Security Camera API
- AppSec Tools
- OWASP, ISSA, etc…

RSAConference2022

Let's look at the breaches that have occurred.

Who is making the news?

Significant API Breaches…

Peloton's leaky API let anyone grab riders' private account data

But the company won't say if it has evidence of malicious exploitation

Zack Whittaker @zackwhittaker / 7:00 AM EDT • May 5, 2021 Comment

Peloton allows access to all rider's data.

…sed data for over 200,000 sha…

…l Toulas October 8, 2021

…ewDog, the Scottish brewery and pub chain famous for its crowd-ownership model and the ta…
…s irreversibly exposed the details of 200,000 of its shareholders and customers.

Brewdog allowed one person to use another patron's rewards coupons.

Smart Door Locks are an easy target, privilege escalation via API.

experian

Clubhouse

JOHN DEERE

And many more…

CEQUENCE® SECURITY

RSAConference2022

Where are we in Global API Security Maturity

Maybe not your organization but the overall trend.

API Security History

- Code Reviews
- Static Analysis
- Penetration Testing
- Dynamic Analysis
- Shift Left, DEVSECOPS, etc…

Challenges For Traditional API Security Tools

- "Scanners"
  - APIs don't have HREFS
  - APIs don't have sitemaps
  - Where can the scanner learn the application endpoints?
- Inline
  - Automatic Discovery
  - Automatic Analysis

RSAConference2022

Tools and humans need a map, documentation, if you will.

Let's look at some examples.

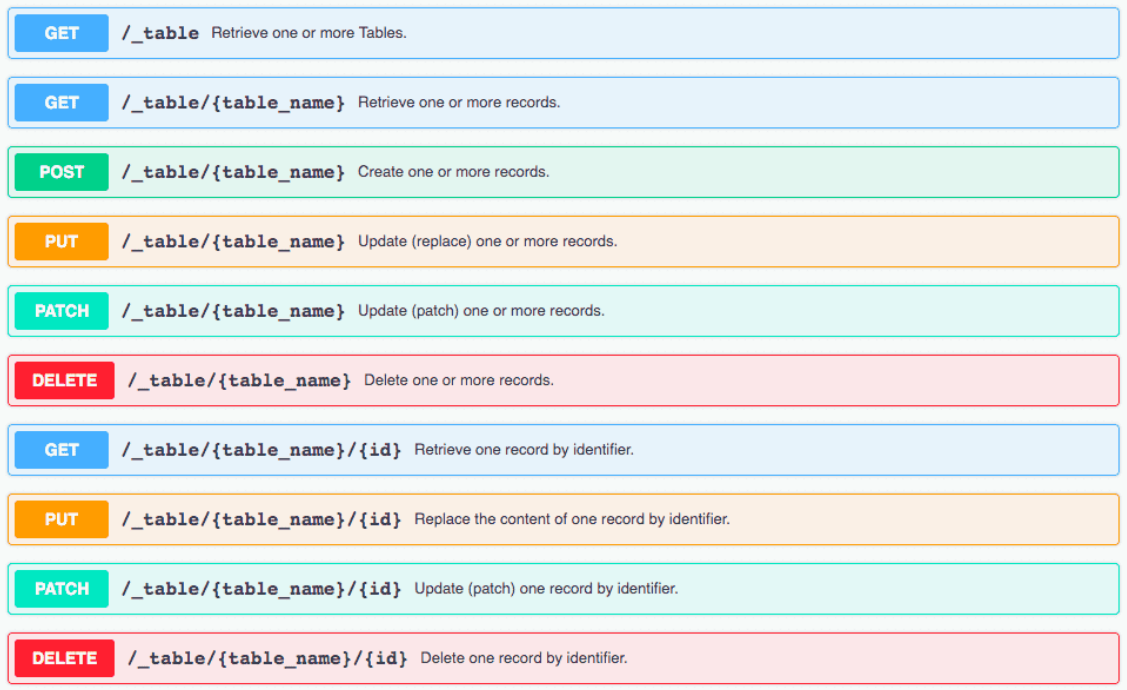
API Security Documents

API DOCUMENTATION

API DEFINITIONS

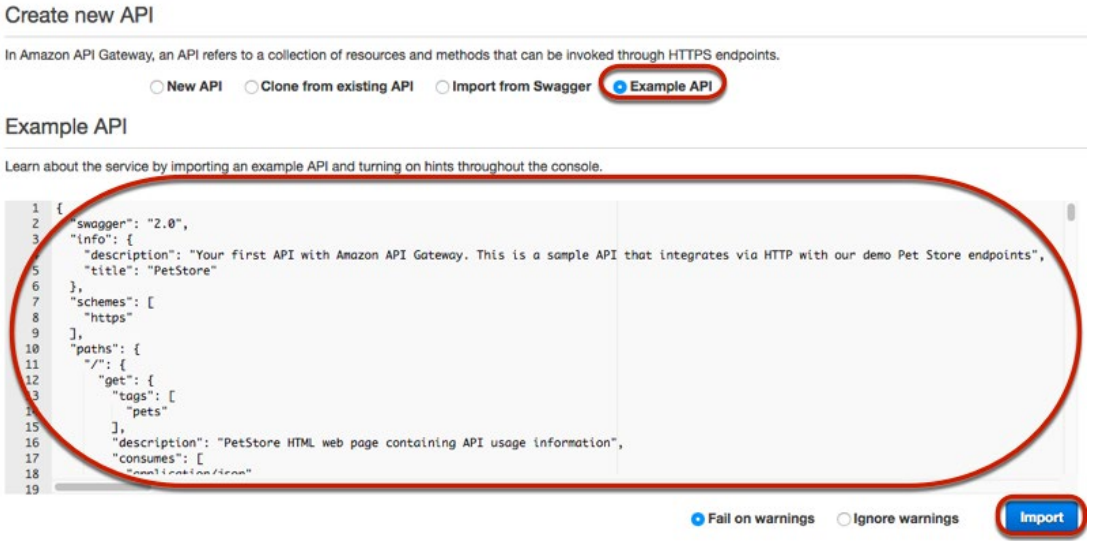
API SPECIFICATIONS

API Documentation



- Consumed by humans, usually developers
- Typically includes:
  - Getting started
  - Tutorials
  - Syntactically correct calls
  - Code samples for various languages
- Needs to be actively maintained and up to date

API Definition



- Consumed by machines and automated systems — not humans. https://nordicapis.com/difference-api-documentation-specification-definition/
- Can be used to configure API Gateways, for example.

API Specification

- The API Spec explains how the API behaves.
- Identifies:
  - Objects
  - How to call objects
  - What objects do
- DAST scanners can consume this data.

RSAConference2022

How can I protect my APIs from being easily found by attackers?

The Attack Chain and API Specifications

- Find a public API
- Find API documentation
- Read the spec
- Test for vulnerabilities
- Instrument the attack

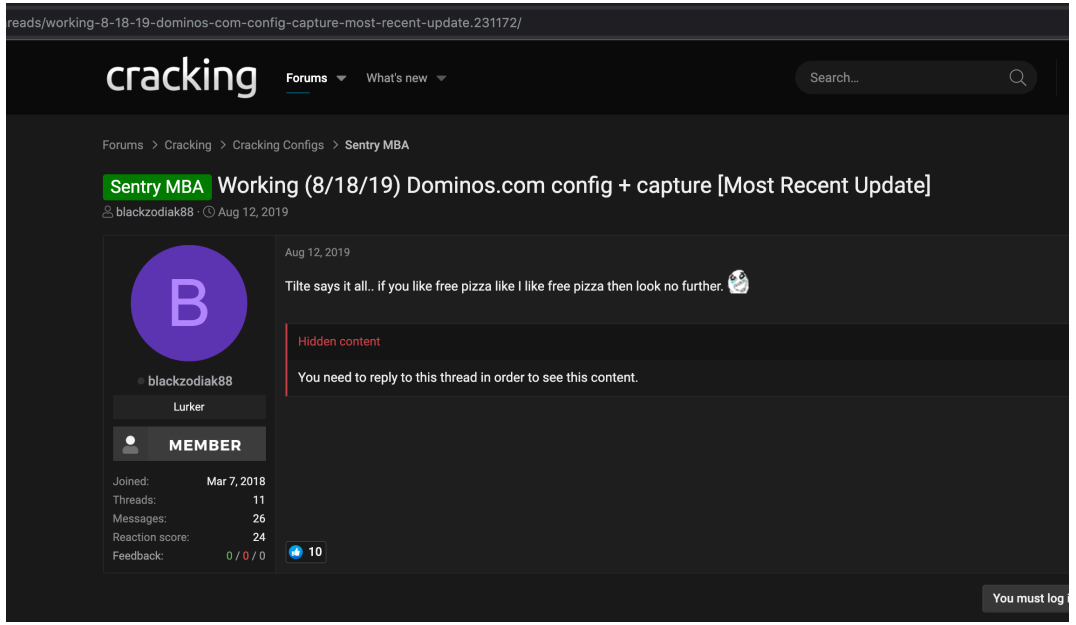
Finding YOUR Public APIs

- APIs.guru
- Programmableweb.com
- Web/Mobile Application Instrumentation – Reverse Proxy
- Partner programs/3rd party integration
- DNS searches
- APK mining
- Cracking

CEQUENCE® SECURITY

A Quick Aside on Cracking.org

- This is a host for OpenBullet and SentryMBA configs.
- If a config for your company is on this page, your APIs are under attack
- Use it to your advantage
  - Step one: analyze it to see what the attack does.
  - Step two: analyze traffic on the endpoints listed to ensure they aren't getting through.



RSAConference2022

Found the public API, let's find out if it has a spec.

Manual or Automatic?

Manual Analysis – Difficult and Repetitive

- /api/docs
- /api/v1/docs
- /api/shareddocs
- /rest_config
- /v3/api-docs/swagger-config
- /api/swagger

https://raw.githubusercontent.com/tananaev/traccar/master/swagger.json

Kiterunner From Assetnote

- OpenSource AIO tool for finding APIs, specs and vulnerabilities.
  - Attackers are using it too
  - Use it to your advantage - see what they may see
  - Take steps to shrink your blast radius

Kiterunner: Contextual Content Discovery

05 Apr 2021

Kiterunner

https://labs.assetnote.io/tool/release/2021/04/05/contextual-content-discovery.html

Kiterunner - Wordlists

- Gigs of wordlists
- Generated using BigQuery against GitHub
- Uses “config files” in the form of additional wordlists for finding specs

https://wordlists.assetnote.io/

What to Protect so you Aren't a Target

- The obvious endpoints are those with API subdomains {api.example.com}
- Do you have common endpoints?
  - {api.example.com/swagger.json}
  - {api.example.com/graphql}
  - {api.example.com/healthcheck}
  - {api.example.com/gibberish}
- Some become obvious potential targets
  - {api.example.com/swagger.json}

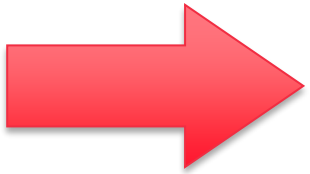
RSAConference2022

On to the attack

We have our target, lets get going

Attack Behavior

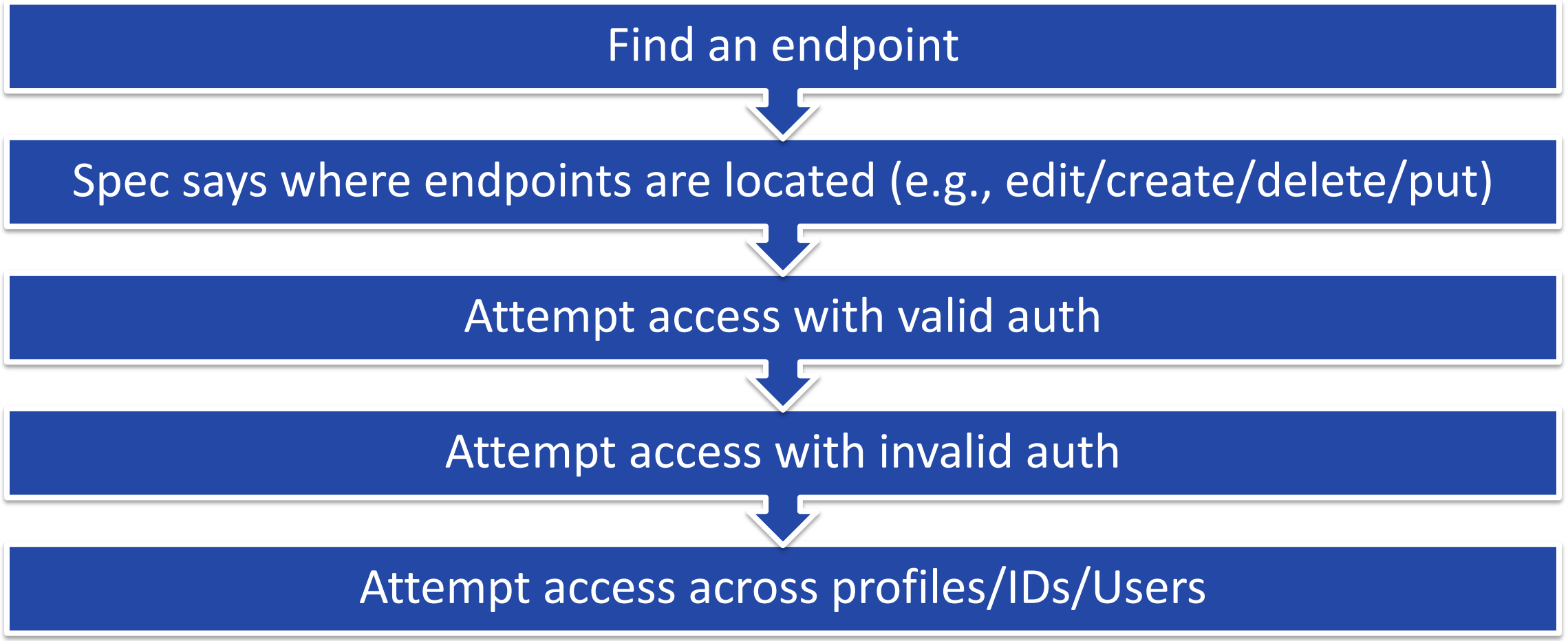
- My Toolkit
  - Infrastructure
  - Tools
  - Behavior
  - Credentials



- My Targets
  - Login Attempts
  - Profile Reads
  - Document Enumeration
  - Logic Attacks

Attack Example

Using specs to attempt BOLA



Specification Docs Tell Us Where To Look

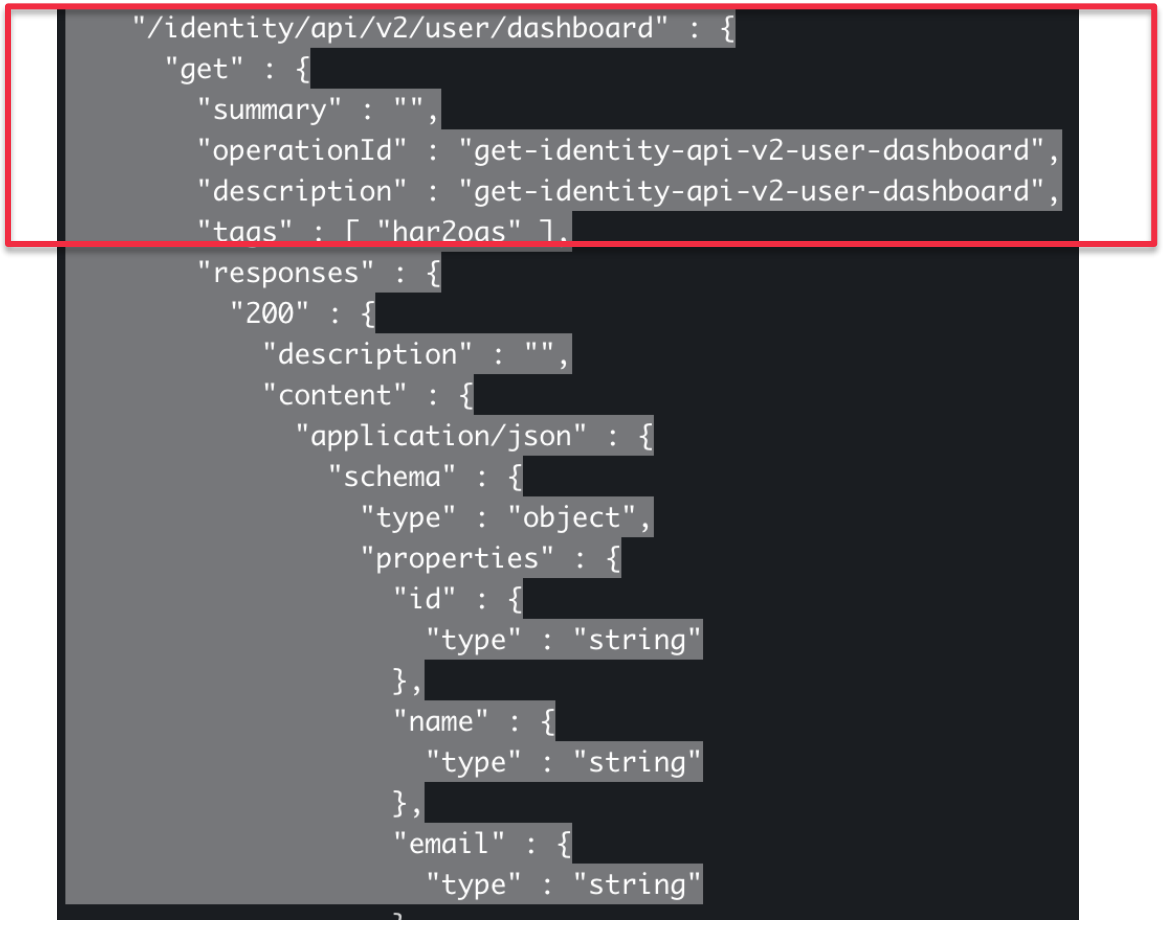
```
158    "/api/users/{group_id}": {
159      "post": {
160        "callbacks": {},
161        "deprecated": false,
162        "description": "Create a user",
163        "operationId": "UserController.create",
164        "parameters": [
165          {
166            "description": "Group ID",
167            "example": 1,
168            "in": "path",
169            "name": "group_id",
170            "required": true,
171            "schema": {
172              "type": "integer"
173            }
174          }
175        ],
```

```
205    "/api/users/{id}": {
206      "get": {
207        "callbacks": {},
208        "deprecated": false,
209        "description": "Show a user by ID",
210        "operationId": "UserController.show",
211        "parameters": [
212          {
213            "description": "User ID",
214            "example": 123,
215            "in": "path",
216            "name": "id",
217            "required": true,
218            "schema": {
219              "type": "integer"
220            }
221          }
222        ],
223        "responses": {
224          "200": {
225            "content": {
226              "application/json": {
227                "schema": {
228                  "$ref": "#/components/schemas/UserResponse"
229                }
230              }
231            },
232            "description": "User"
233          }
234        },
235        "summary": "Show user",
236        "tags": [
237          "users"
238        ]
239      }
```

CEQUENCE® SECURITY

An Example From crAPI

- HTTP Method GET
- Note /v2/
- Are there other user elements besides dashboard?



When You Get Back To The Office

- Find your API specs and understand who builds them and where they are stored.
  - This allows for versioning and an understanding of where changes might need to be made.
- Review your specs to understand where an attacker would use them to bypass your security protections
  - Noting things like Delete being available on a flow allows for testing of privilege escalation.
- Use specs as part of your internal programs - not external ones.
  - Protect with NDAs/Auth etc... Having unfettered access means mapping your APIs is simple for an attacker.

Specification Best Practices

1. Embrace specification use – consistency, quality, security
2. Establish policies: no API gets published without a spec
3. test against the spec: does intended function follows the spec?
4. Protect the spec: public accessibility is not a requirement
5. Replace the defaults: make it harder for information to leak out
6. Manage the scanners: use them to see what attackers see, apply preemptive policies

OpenAPI Security Recommendation #1

- Servers
  - Errors can lead to OWASP API 9: Improper assets management vulnerability
- Confirm HTTPs is in use
- Validate version numbers
- Consistent naming conventions

```
{
  "servers": [
    {
      "url": "https://development.gigantic-server.com/v1",
      "description": "Development server"
    },
    {
      "url": "https://staging.gigantic-server.com/v1",
      "description": "Staging server"
    },
    {
      "url": "https://api.gigantic-server.com/v1",
      "description": "Production server"
    }
  ]
}
```

OpenAPI Security Recommendation #2

- Methods and paths
  - Errors can lead to OWASP API 9, Improper assets management vulnerability
- Ensure they are used as intended
  - If GET is intended, make sure POST, PUT or DELETE are not used

```
"paths": {
  "/pets": {
    "get": {
      "summary": "List all pets",
      "operationId": "listPets",
       ...
    },
    "post": {
      "summary": "Create a pet",
      "operationId": "createPets",
      ...
    }
  }
}
```

OpenAPI Security Recommendation #3

- Request parameters
  - Errors can introduce Mass Assignment, #6 on the OWASP API Security Top 10 list
- Never leave parameter values open ended
- Block or remove any added parameters

```
"parameters": [
  {
    "name": "limit",
    "in": "query",
    "description": "How many
items to return at one time (max
100)",
    "required": false,
    "type": "integer",
    "format": "int32"
  }
]
```

OpenAPI Security Recommendation #4

- Response body
  - Errors can introduce Excessive Data Exposure risks, #3 on the OWASP API Security Top 10 list
- Expose as little info as possible
- Specify response structure for individual fields.
- Use scanning tools to flagged undocumented fields

```
"responses":{
  "200":{
    "description":"Successful
operation",
    "content":{
      "application/xml":{
        "schema":{

"$ref":"#/components/schemas/Pet
"
        }
      },
      "application/json":{
        "schema":{

"$ref":"#/components/schemas/Pet
"
        }
      }
    }
  }
}
```

CEQUENCE® SECURITY

OpenAPI Security Recommendation #4

- Security Schemes
  - Can lead to Broken User Authentication, #2 on the OWASP API Security Top 10 list
- Use authentication wherever possible

```
"securitySchemes":{
  "petstore_auth":{
    "type":"oauth2",
    "flows":{
      "implicit":{

"authorizationUrl":"https://petstore3.swagger.io/oauth/authorize"
        "scopes":{
          "write:pets":"modify pets in your account",
          "read:pets":"read your pets"
        }
      }
    }
  },
  "api_key":{
    "type":"apiKey",
    "name":"api_key",
    "in":"header"
  }
}
```

Thank You

- Thank you for attending my talk
- Find me online
  - jason.kent@cequence.ai
  - linkedin.com/in/n0handle