42crunch

approov

8 March 2022

# How to Extend Protection of your Data from API to Mobile Application

**Colin Domoney**

API Security Research Specialist & Developer Advocate

**David Stewart**

Chief Executive Officer

Introduction

# About the Speakers

## Colin Domoney

*API Security Research Specialist & Developer Advocate*

**42Crunch**

## David Stewart

*Chief Executive Officer*

**Approov**


# Housekeeping Rules

- All attendees muted
- Questions via chat
- Recording will be shared
- Polling questions

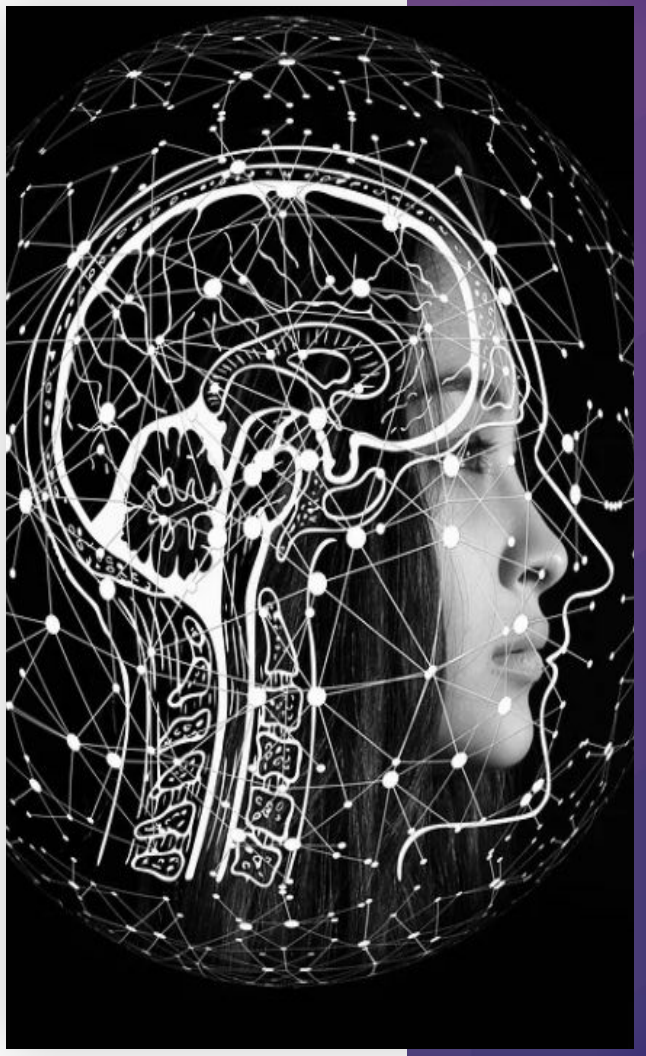
# Agenda

- Mobile API attacks 101
- Approov deep-dive
- 42Crunch deep-dive
- Live demo
- Use cases / benefits
- Questions and answers

# Question One:

## What do you consider the biggest *single* threat to your mobile applications?

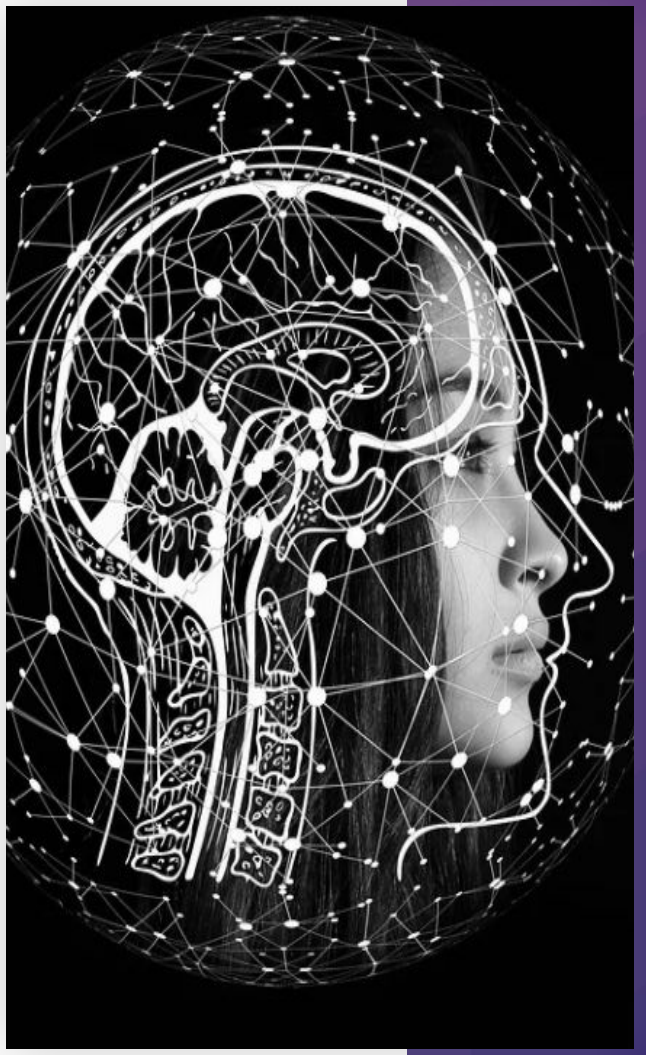
1. User credentials issues
2. Application integrity
3. Device integrity
4. API channel integrity (transport layer)
5. API backend vulnerabilities



# Question One:

## What do you consider the biggest *single* threat to your mobile applications?

| | |
|---|---|
| User credentials issues | 30% |
| Application integrity | 15% |
| Device integrity | 20% |
| API channel integrity (transport layer) | 5% |
| API backend vulnerabilities | 30% |



approov

MOBILE APP PROTECTION

# How to Extend Your Protection of your Data From API to Mobile Application

David Stewart

david.stewart@approov.io

https://approov.io

@approov_io

approov

# Two API Attack Approaches

(1) Attack by exploiting a vulnerability in the API itself

**OWASP API Security Top 10**

- A1 : Broken Object Level Authorization
- A2 : Broken Authentication
- A3 : Excessive Data Exposure
- A4 : Lack of Resources & Rate Limiting
- A5 : Missing Function Level Authorization
- A6 : Mass Assignment
- A7 : Security Misconfiguration
- A8 : Injection
- A9 : Improper Assets Management
- A10 : Insufficient Logging & Monitoring

(2) Attack by automating app API traffic to impersonate a genuine source

approov

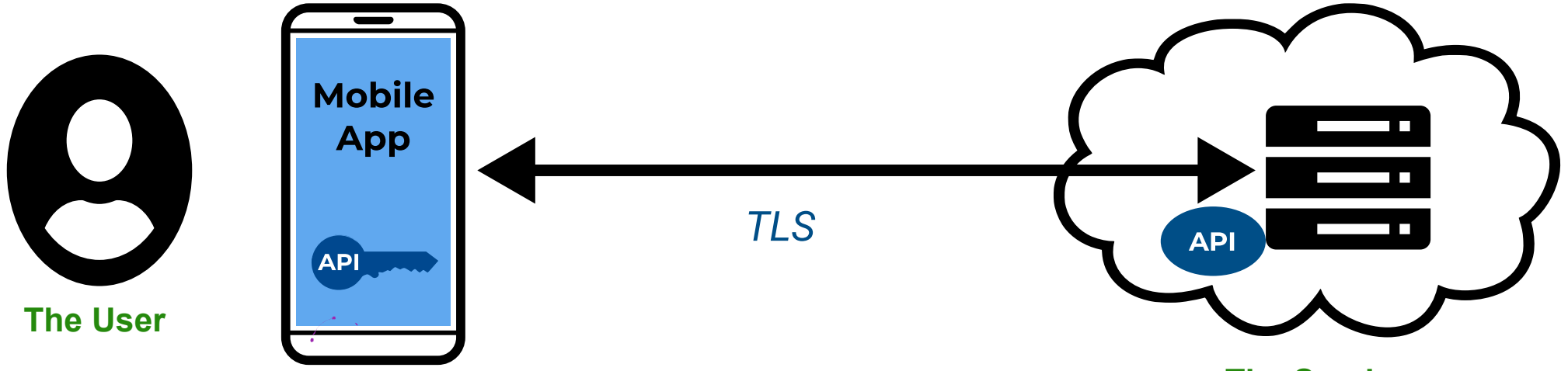
# Two API Defense Approaches

Shift Left...

...AND...

Shield Right

# (Naive) View of Protecting a Mobile Business




*Username/password*
*OAuth2*
*OpenID Connect*
*Biometrics*

*WAF*
*API Gateway*
*CDN*

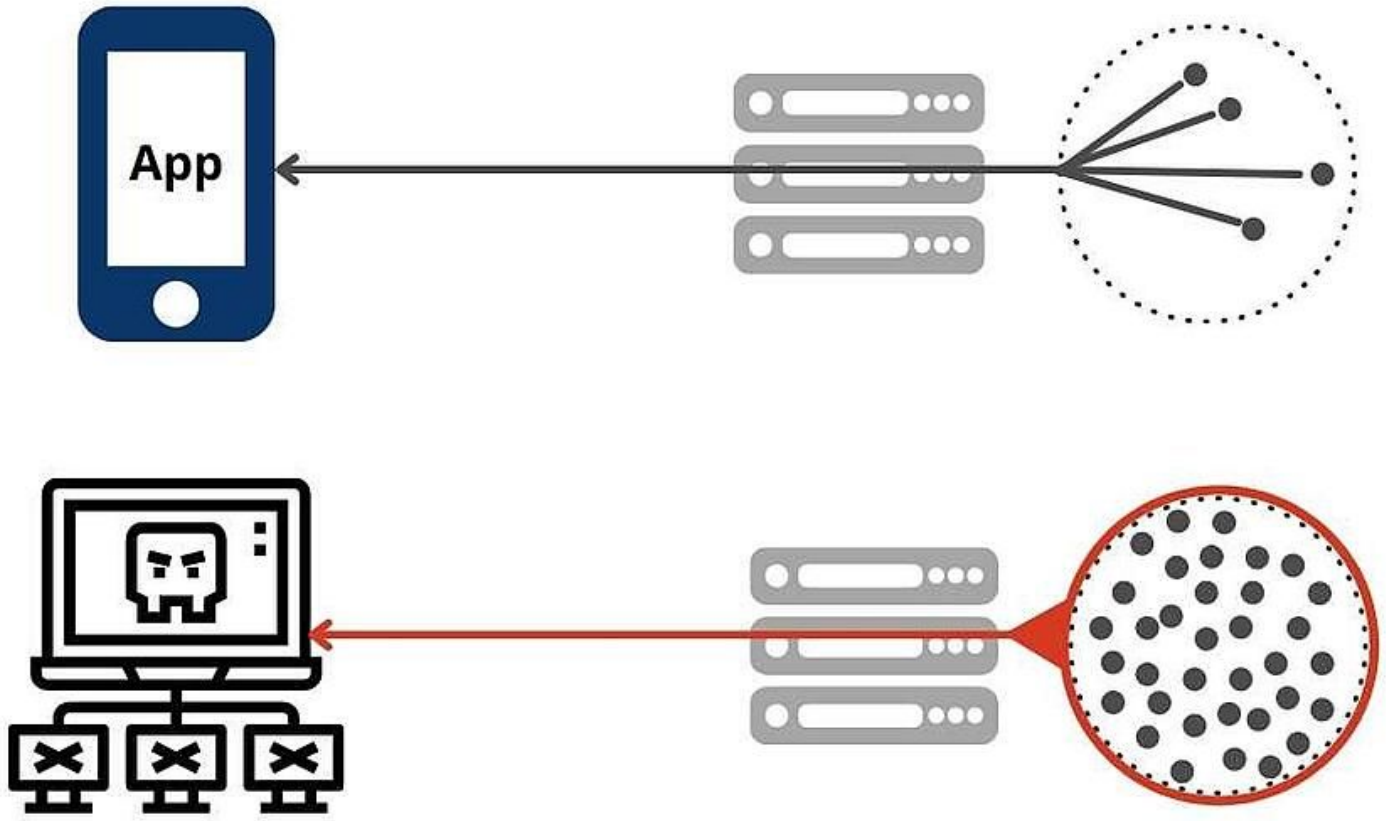
Check user credentials and possession of a valid API key.

# Mobile Apps: Gifts that Keep on Giving



Downloadable and runnable by anyone, anytime, for any duration and on any platform.

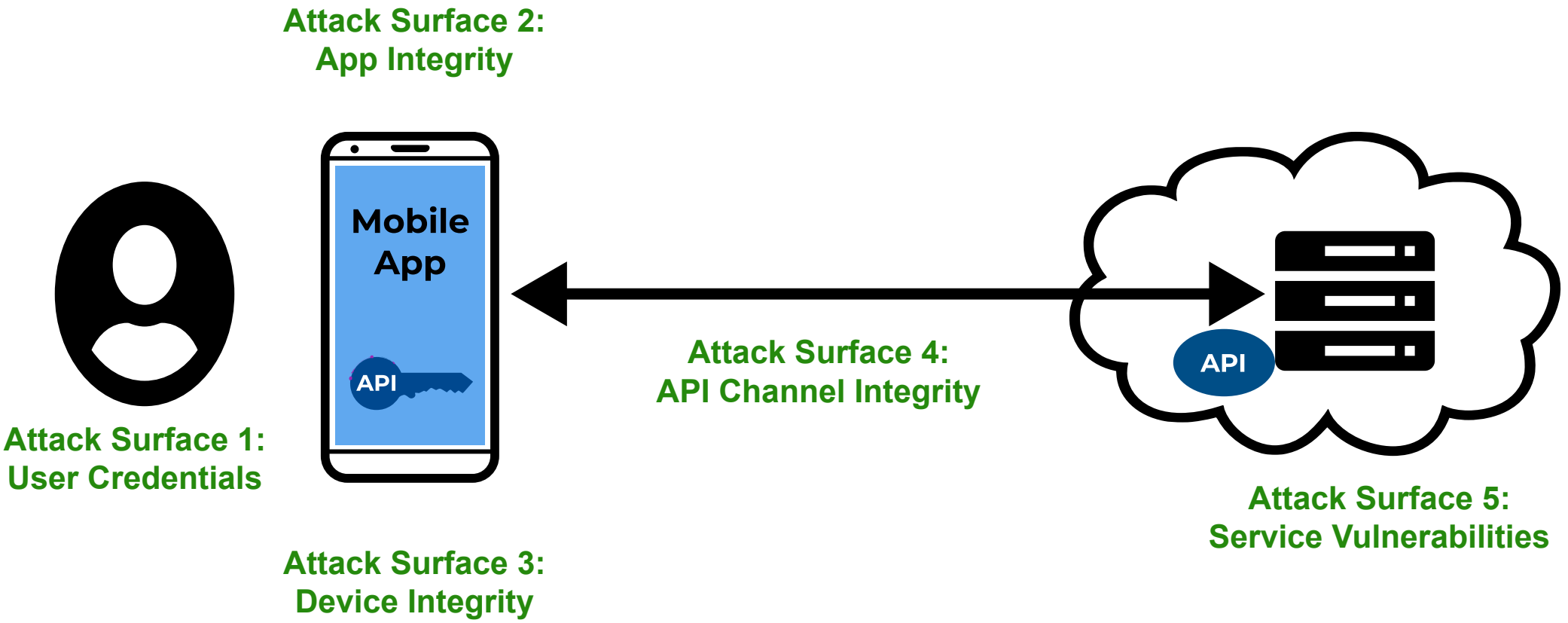
# Mobile APIs: Flood Gates Waiting to Open




*An app limits the range/speed an API can manipulate user data. However, a bot can rapidly manipulate and exfiltrate all your valuable data.*

approov

# The Five Mobile Attack Surfaces




Trust nothing between user and service.

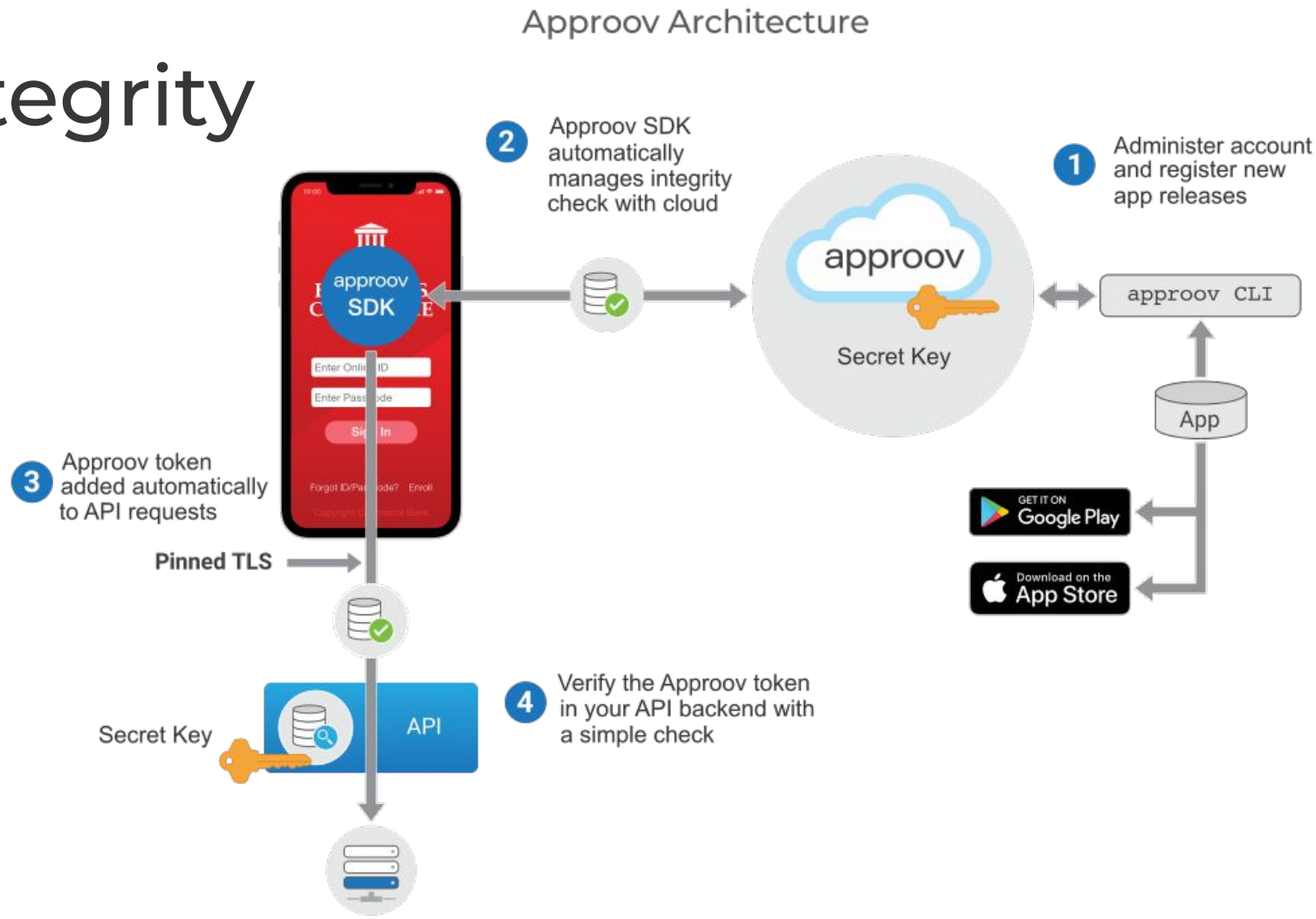
approov

# Approov App Integrity

Under the hood:

1. Register app
2. App authenticated
3. Approov token delivered
4. Token validated

Repeat 2. every 5 minutes




https://approov.io/download/Approov-Whitepaper-Security-Trust-Gap.pdf

approov

# Device Integrity

Approov detects potentially unsafe mobile device environments including device rooting/jailbreaking, emulator or debugger usage, malicious instrumentation frameworks, and cloned apps. Customers specify which policies should be enforced. Changes to security policies roll out immediately to active apps.

Potentially unsafe conditions detected include:

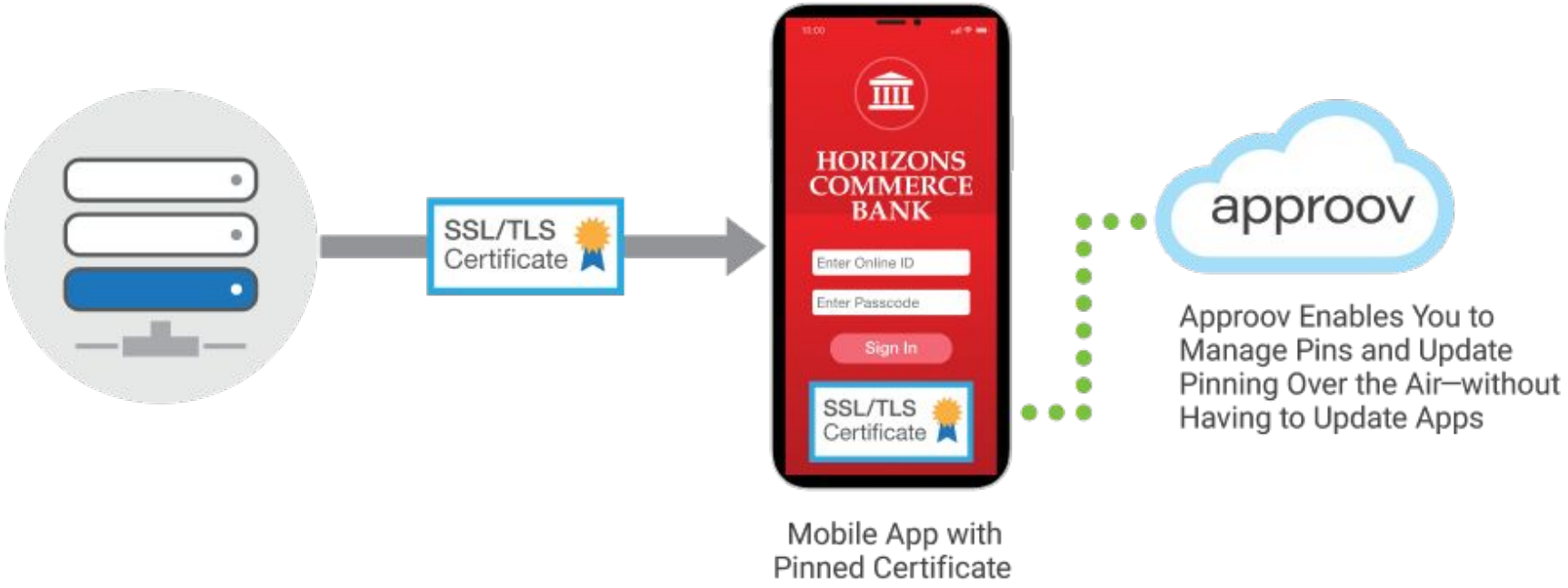
- debug
- device changed
- root risk
- xposed unsafe
- magisk hidden
- scan-fail
- app not registered
- emulator
- cydia
- risky device
- tampered
- rooted
- ios simulator
- frida
- devicecheck ban
- bad hmac
- jailbroken
- multi-app clone
- cycript
- non-standard launch
- automated launch

Environmental checks run at app launch and every 5 minutes during the session

approov

# API Channel Integrity

## Dynamic Certificate Pinning:

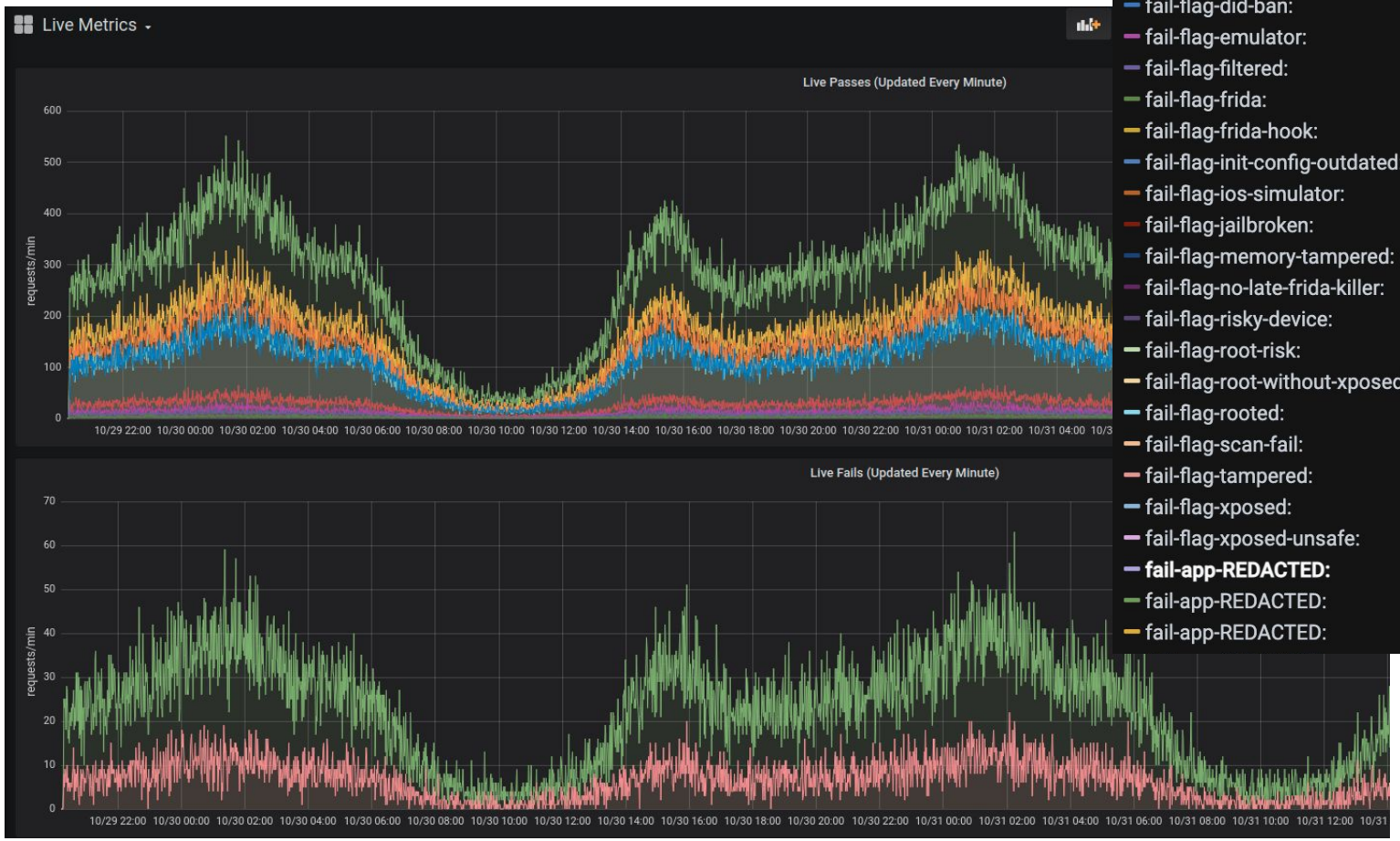



Continuous monitoring of pins from Approov cloud and immediate notification of changes that will cause app pinning failures

approov

# Approov Live Dashboard Analytics

Adjust your security position based on real time data from your app installed base.

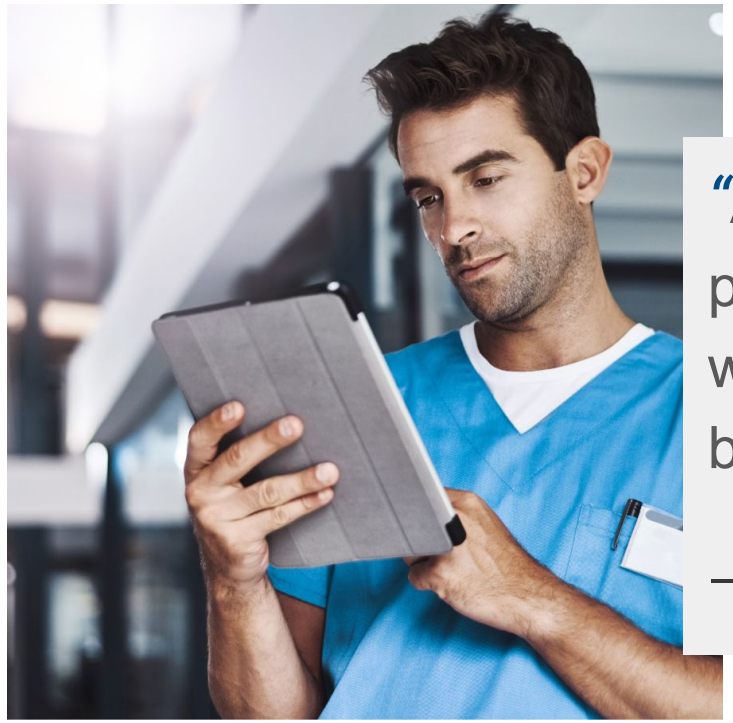


approov

# Case Study: MV

https://approov.io/download/Approov-MV-Story.pdf

## Protecting Patient Data While Delivering Agility To Physicians



"Approov plugged an immediate hole which pentesting had exposed in our platform, and we calculate that the adoption of Approov will bring us a 10x RoI."

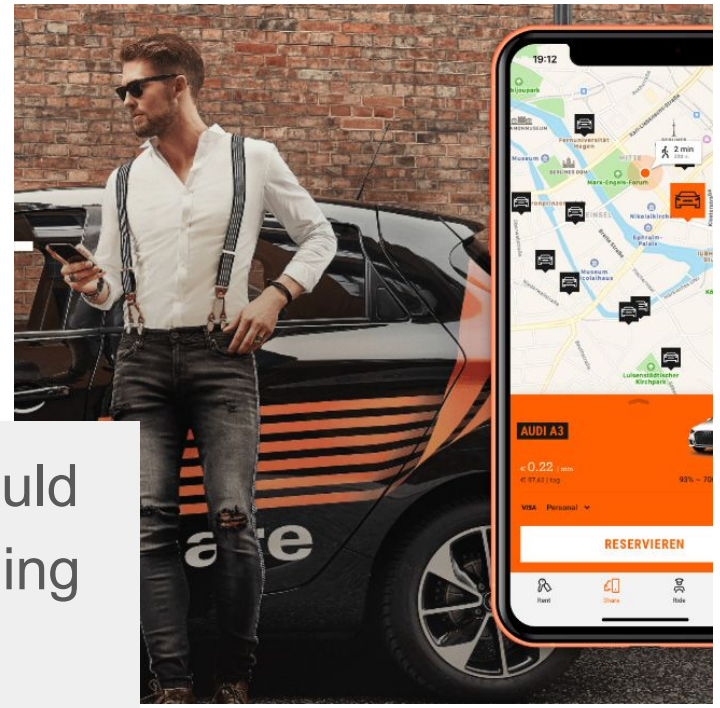
— Tiago Calado, Software Development Mgr, MV.

approov

# Case Study: SIXT

https://approov.io/download/Approov-Sixt-Story.pdf

## Minimizing the Business Impact of Data Scraping




> "We looked around for a solution which could authenticate when API requests were coming from our mobile apps, and that's when we came across Approov."
>
> — Nico Gabriel, President, SixtX.

Approov Mobile App Protection

# Protect Your Apps. Protect Your APIs. Protect Your Revenue.

https://approov.io/signup

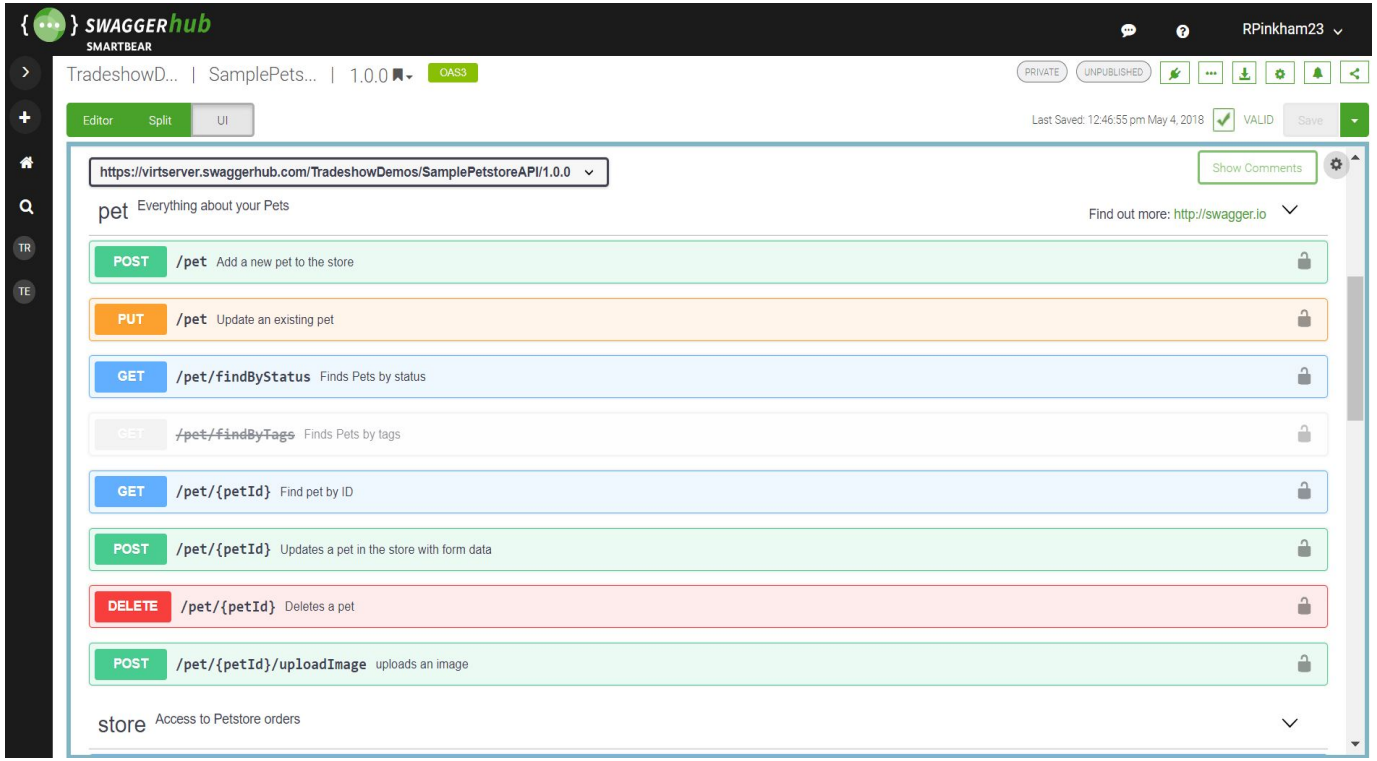
# Why API security is hard ?

# API security is different to web security

| OWASP API Security Top 10 | OWASP Top 10 (2017) |
|---|---|
| API1: Broken Object Level Authorization | A1: Injection |
| API2: Broken User Authentication | A2: Broken Authentication |
| API3: Excessive Data Exposure | A3: Sensitive Data Exposure |
| API4: Lack of Resources & Rate Limiting | A4: XML External Entities (XXE) |
| API5: Broken Function Level Authorization | A5: Broken Access Control |
| API6: Mass Assignment | A6: Security Misconfiguration |
| API7: Security Misconfiguration | A7: Cross-Site Scripting (XSS) |
| API8: Injection | A8: Insecure Deserialization |
| API9: Improper Assets Management | A9: Using Components with Known Vulnerabilities |
| API10: Insufficient Logging & Monitoring | A10: Insufficient Logging & Monitoring |

# APIs are an easy target to attack

- They are easily **discoverable**
- They are well **documented**
- Attacks can be easily **automated**
- Excellent tools exist to automate attacks



# Your existing tools probably don't work well for APIs

- **SAST** — wasn't designed for API-centric applications. Complex data flow paths or unsupported frameworks reduce the accuracy of a SAST analysis since the model may be incomplete or inaccurate.
- **DAST** — lacks context of APIs. DAST tools can't provide an intelligent assessment of API security.
- **SCA** – useful but not sufficient
- **IAST** – complex to install and use


https://thenewstack.io/application-security-tools-are-not-up-to-the-job-of-api-security/

# Question Two:

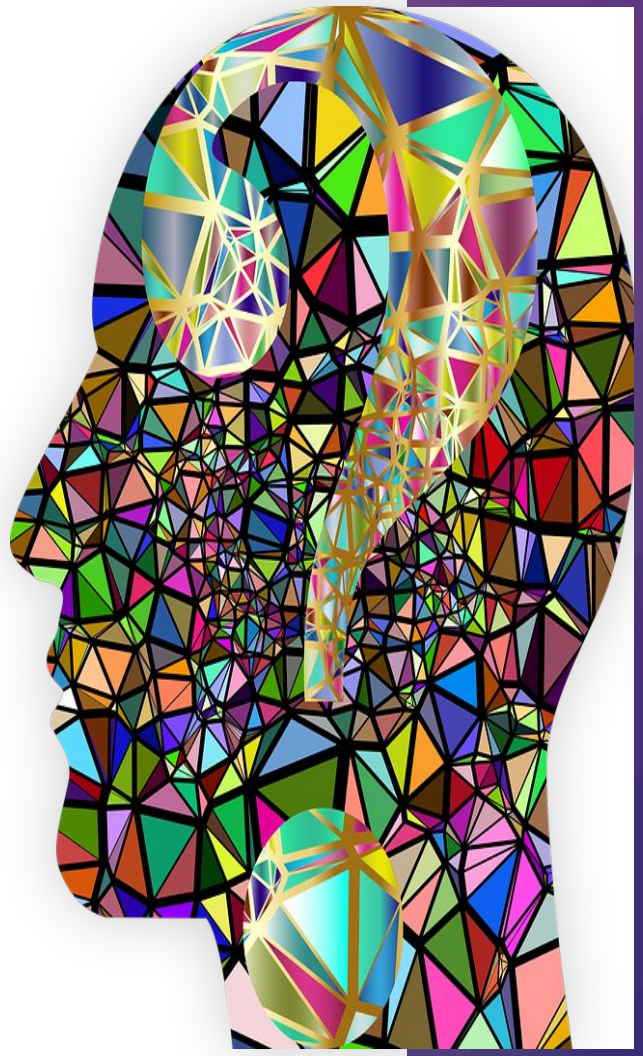
## How are you protecting your APIs?

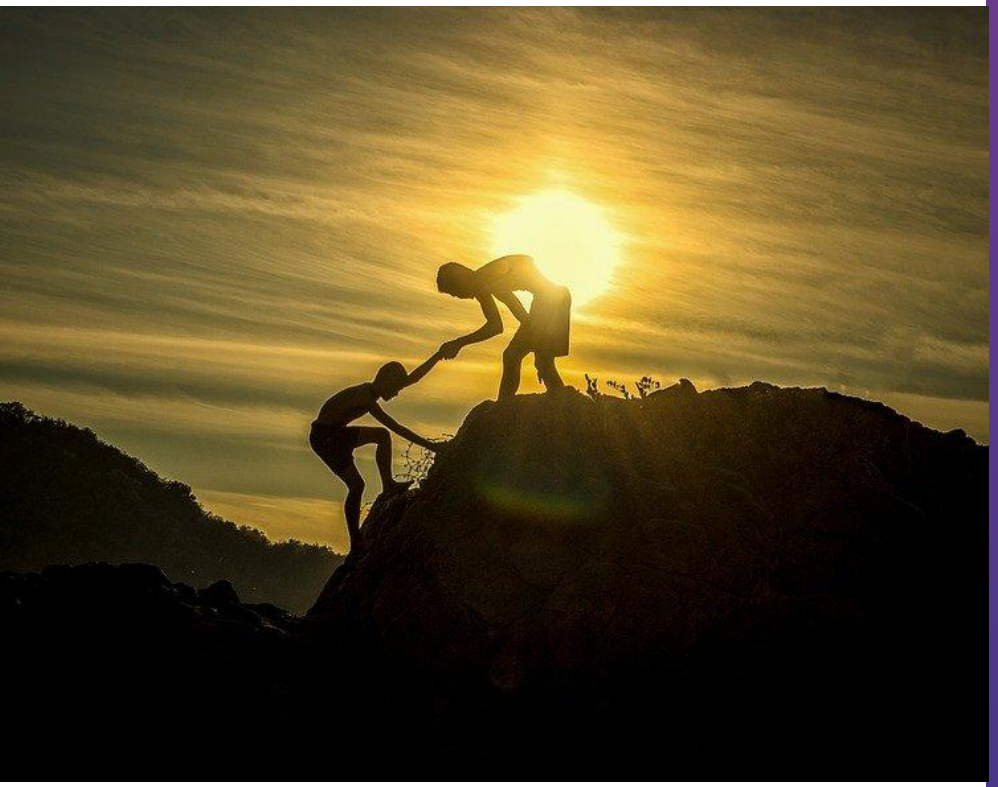
1. WAF
2. API Gateways
3. Dedicated API micro-firewalls
4. Runtime protections (RASP, IAST)

# Question Two:

# How are you protecting your APIs?

| | |
|---|---|
| WAF | 68% |
| API Gateway | 80% |
| Dedicated API micro-firewalls | 4% |
| Runtime protections (RASP, IAST) | 12% |

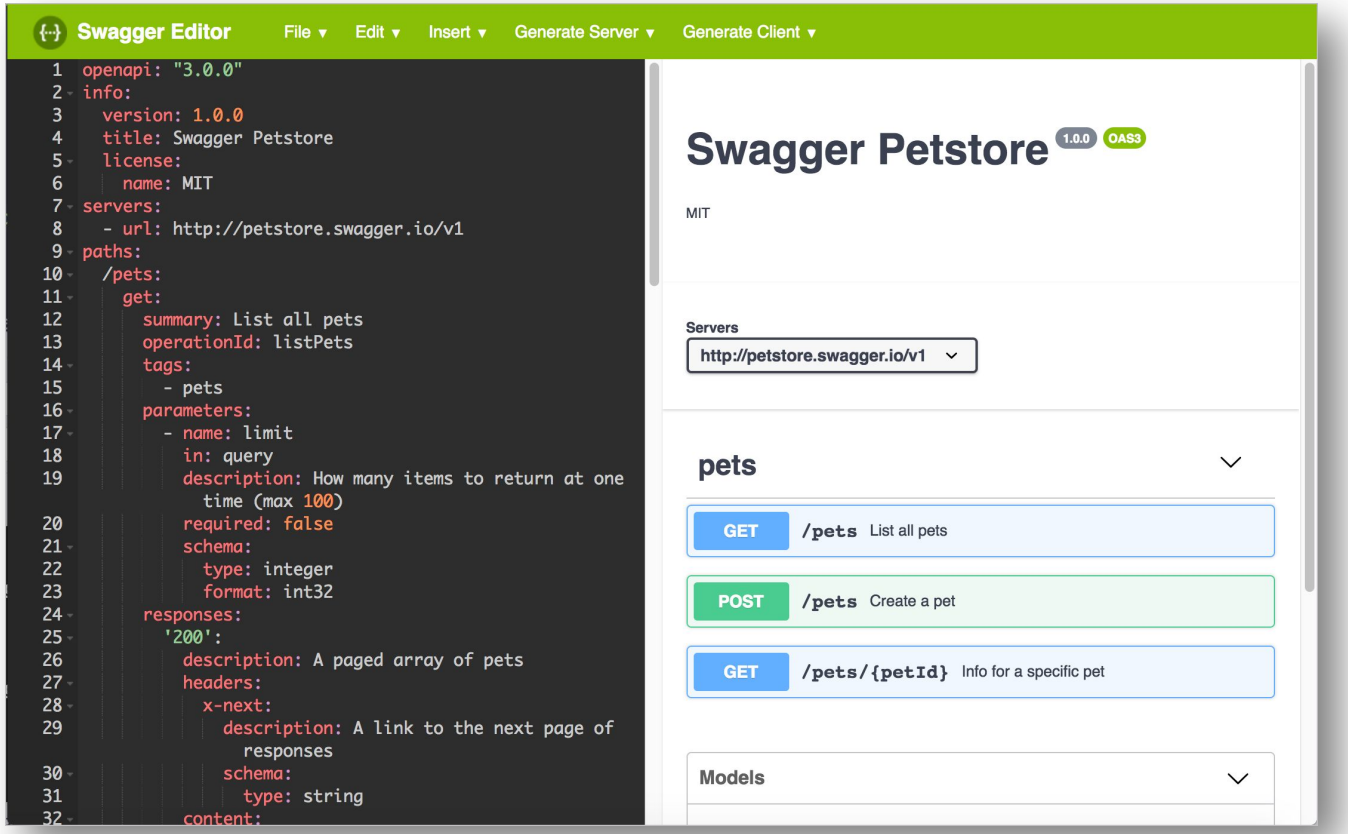




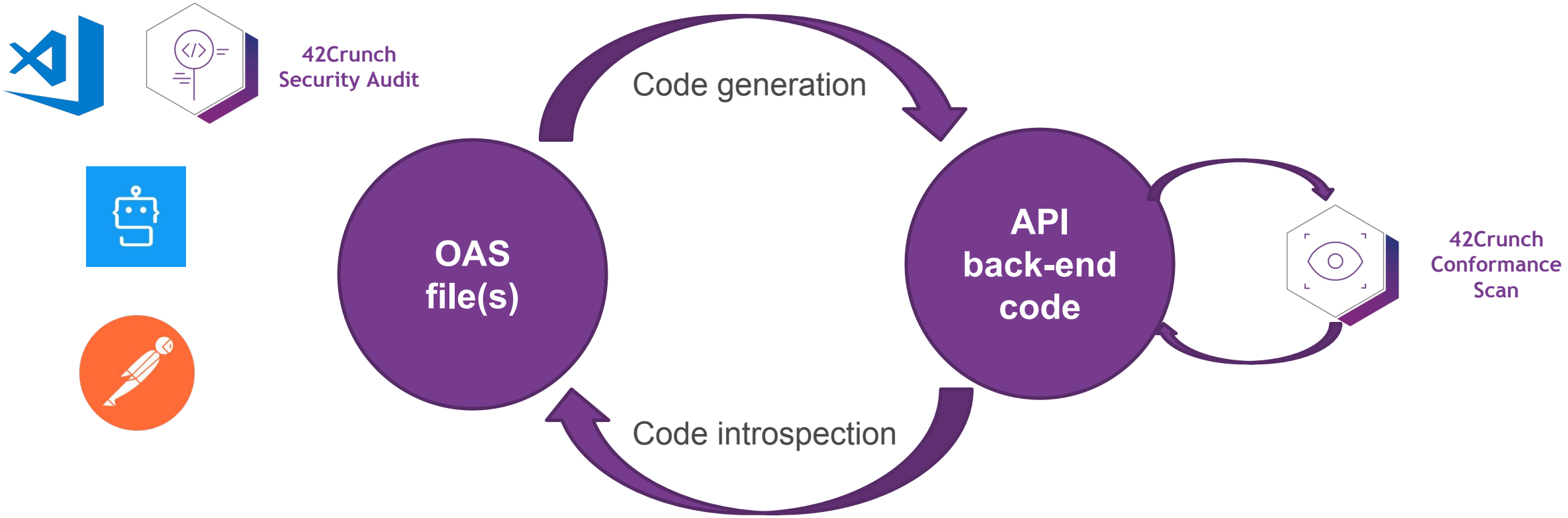
# The unique opportunities for API security

# OpenAPI Specification at the heart

- OAS forms a definitive contract for all downstream development
- OAS allows for a precise definition of request and response data types
- OAS allows operations to be tightly specified
- OAS allows security primitives to be specified
- Extensions allow for additional primitives to be included



# Use OAS as the core of a ‘shift left’ process



# The benefits of a positive security model

## Allowlist

- Allowed data types strongly defined and enforced in OAS mode
- Data format can be precisely defined
- Operations can be fully specified too

- **Only allow data conforming to specification — anything else is an error**
- **Only allows “known good”**

## Vs

## Blocklist

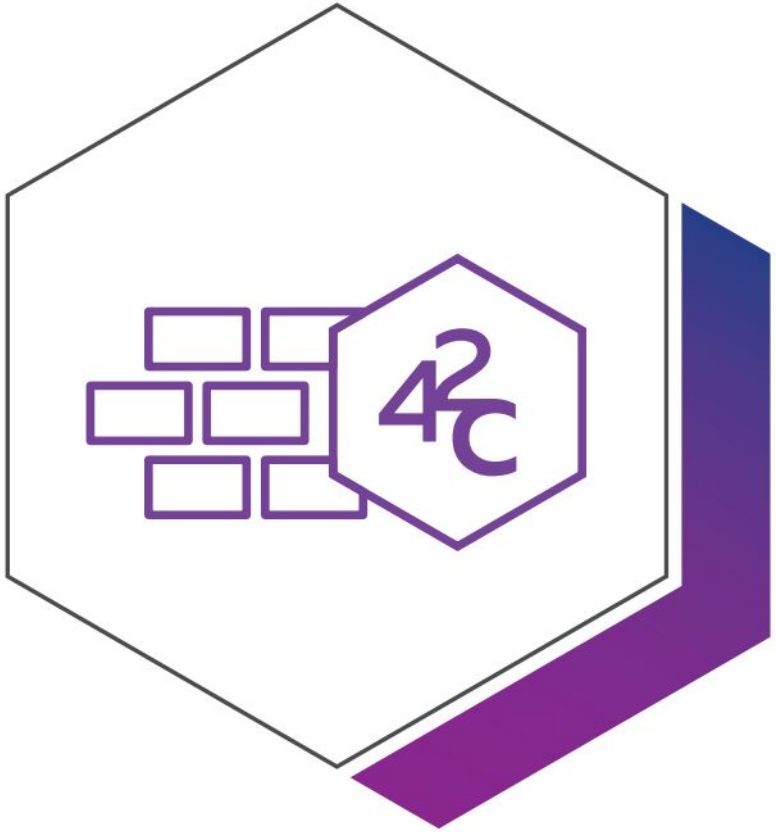
- Attempts to interpret data based on the runtime context i.e., Javascript, HTML
- Attempts to block what shouldn’t be present in each context
- Can easily be subverted with encoding, etc.

- **Attempts to block “known bad”**

# Building guardrails - trust, but verify

- Provide the sec**urity tooling and solutions**
- Provide the **guidance** on usage
- Set the **policies and standards**
- Implement **governance**

- Give the developers the **freedom** to implement their solutions



# Our approach to API security

Continuous API Security

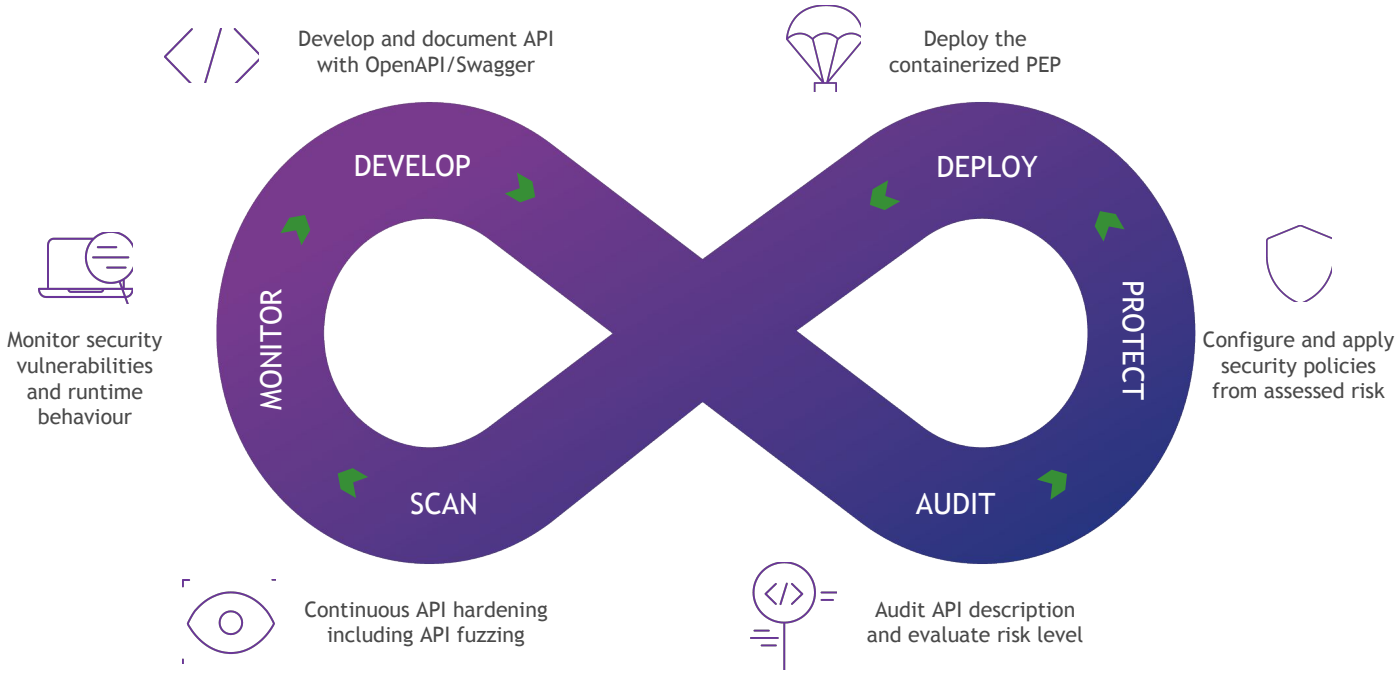
# AUTOMATE & SCALE API SECURITY TO PROTECT YOUR APIs

## SHIFT LEFT

- Growing recognition of need to include security at design time
- *Security as code* for a seamless DevSecOps experience
- *Embed and* **automate security** *in the API development CI/CD pipeline.*

## SHIELD RIGHT

- *Security teams retain control and visibility of the* ***enforcement*** *of API security policies.*
- ***Low-footprint*** *containerized PEP enforces all policies at runtime.*



# THE DEVELOPER FIRST API SECURITY PLATFORM

## SECURITY MANAGEMENT & GOVERNANCE

Visibility & control of security policy enforcement throughout API lifecycle for security teams.

## API AUDIT

Lock down your API's definitions to reduce the attack surface and remove potential security gaps.

## API SCAN

Dynamic runtime testing of your API to ensure compliance with API Contracts.

## API PROTECT

Protect each API with an API micro-firewall to distinguish legitimate traffic from malicious API attacks.

## INTEGRATED ACROSS API LIFECYCLE

Continuous security enforcement across IDE, CI/CD and at runtime.

**BENEFITS:** AUTOMATION - COMPLIANCE - COST SAVINGS - TIME TO MARKET

# LEVERAGE POWER OF THE ECOSYSTEM

Complete and continuous protection for APIs throughout the SDLC.
Secure by design and at runtime.

**SHIFT LEFT**
with 42Crunch AUDIT & SCAN
(Design & Develop)

**SHIELD RIGHT**
with 42Crunch FIREWALL
(Deployment Runtime)


IDE | CI/CD | API MANAGEMENT | CLOUD PLATFORM | MONITORING SIEM

# 42Crunch protections

# #1: Rate limitation

approov

This protection limits how many requests API Firewall accepts from an IP address within a given time window.

```
: x-42c-request-limiter_0.1
```

```
/api/login:
    post:
        x-42c-local-strategy:
            x-42c-strategy:
                protections:
                    - x-42c-request-limiter-based-on-ip_v0.1:
                        window: 15
                        hits: 10
                        burst.enabled: true
                        burst.window: 2
                        burst.hits: 5

                    # ...
```

# #2: Security headers

approov

x-42c-security-headers_0.1

These protections on APIs control security headers either locally to specific paths, operations, responses, or alternatively to all incoming requests or outgoing responses.

```
responses:
    200:
        # ...
        x-42c-local-strategy:
            x-42c-strategy:
                protections:
                    - x-42c-security-headers_0.1:
                        hsts.max_age: 7200
                        csp.policy: default-src: 'self'; upgrade-insecure-requests;
block-all-mixed-content]
                        mode: add-replace
                    # ...
```

# #3: JWT validation

JWT token validation performs a variety of checks on request tokens and blocks invalid requests

```
/api/user/info:
    get:
        x-42c-local-strategy:
            x-42c-strategy:
                protections:
                    - x-42c-jwt-validation_0.1:
                        header.name: x-access-token
                        jwk.envvar: JWK_PUBLIC_RSA_KEY
                        authorized.algorithms: [RS256, RS384]
                    # ...
```

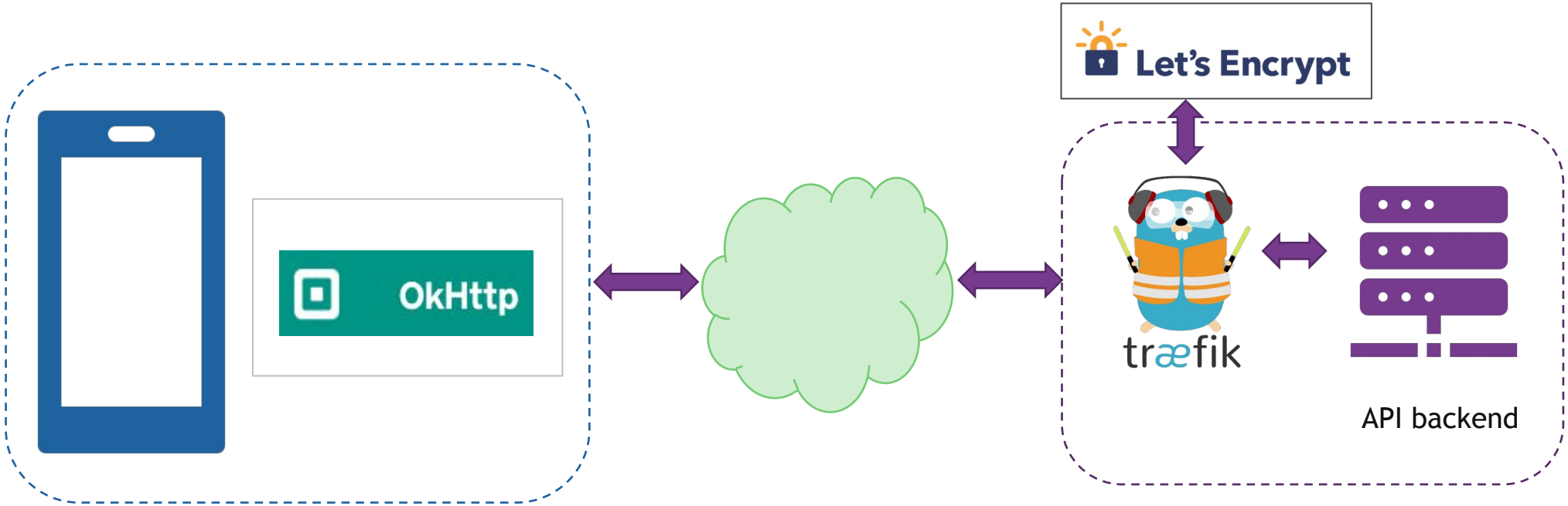
- x-42c-jwt-validation_0.1
- x-42c-jwt-validation-ec_0.1
- x-42c-jwt-validation-hmac_0.1
- x-42c-jwt-validation-rsa_0.1

# Live demo

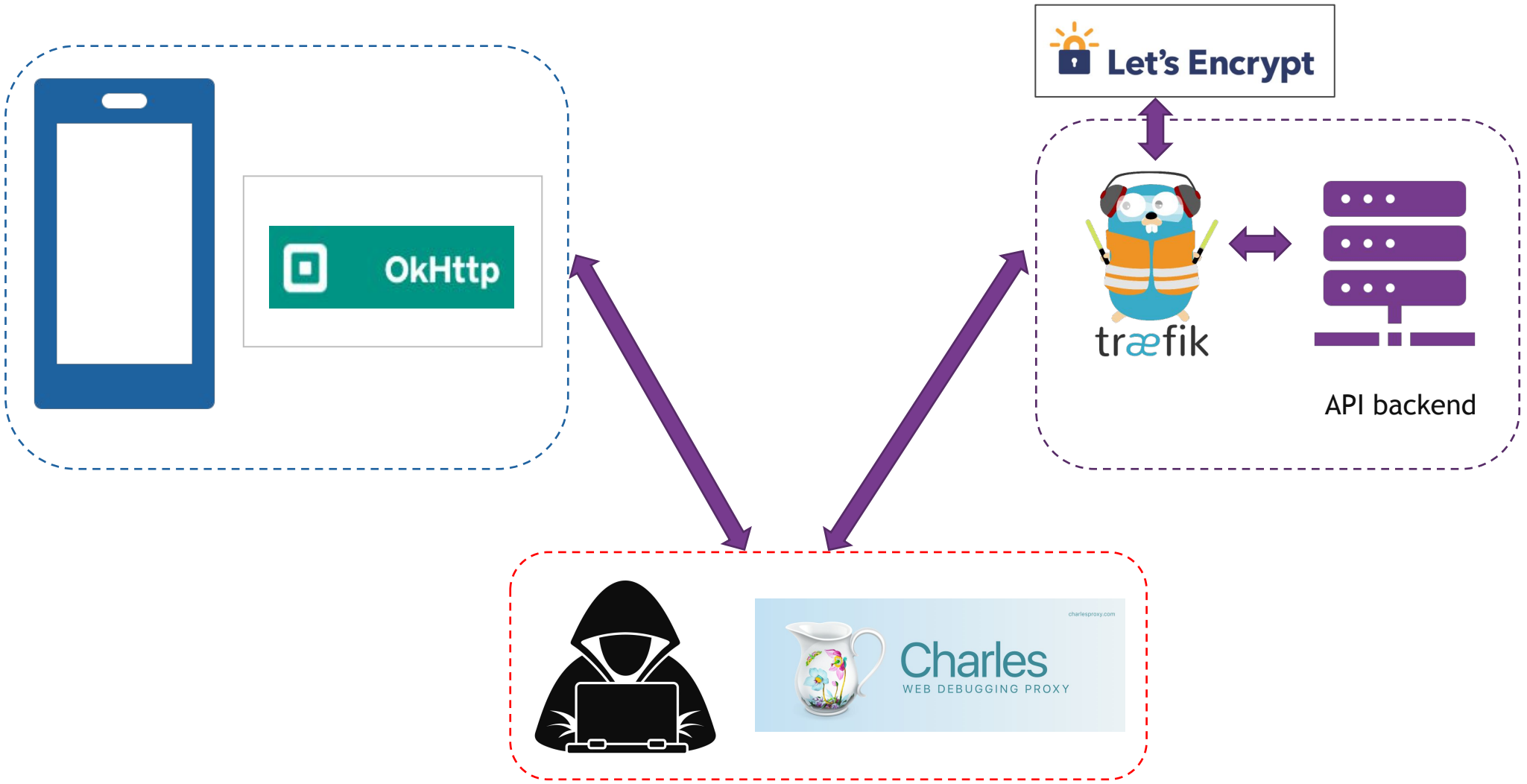
# Demo agenda

- Abusing a mobile app 101
  - Using a proxy
  - Using ‘httpie’
- Protecting against MitM and debugging
- Approov backend protection using 42Crunch
  - With expired tokens
  - Without a token at all
- Additional data protections
  - Excessive data exposure
  - Broken function level authentication

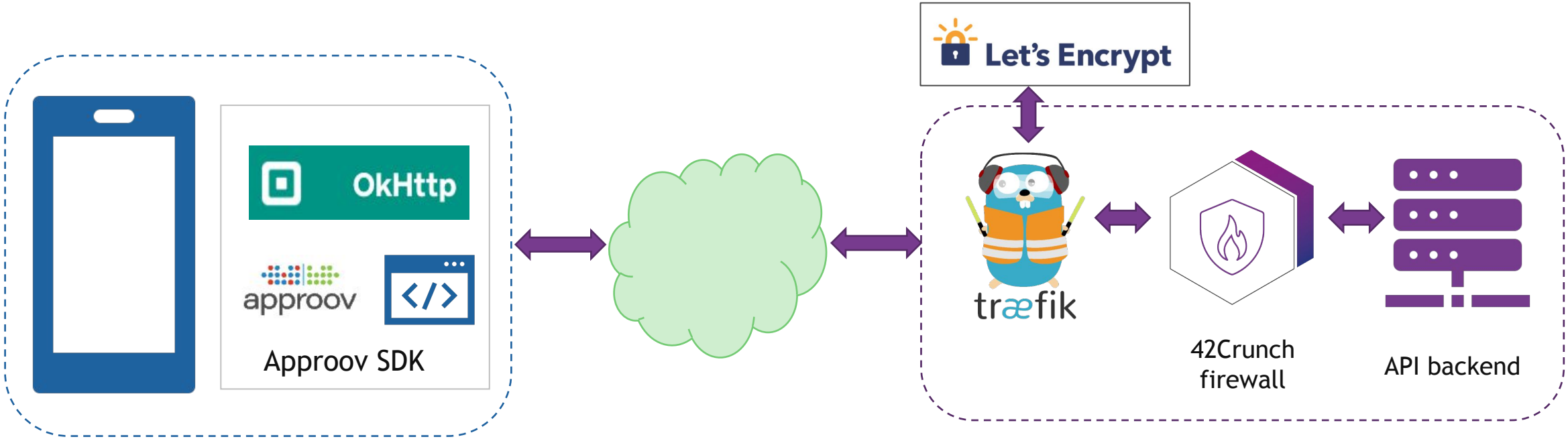
# Red demo environment



# Red demo environment with MitM



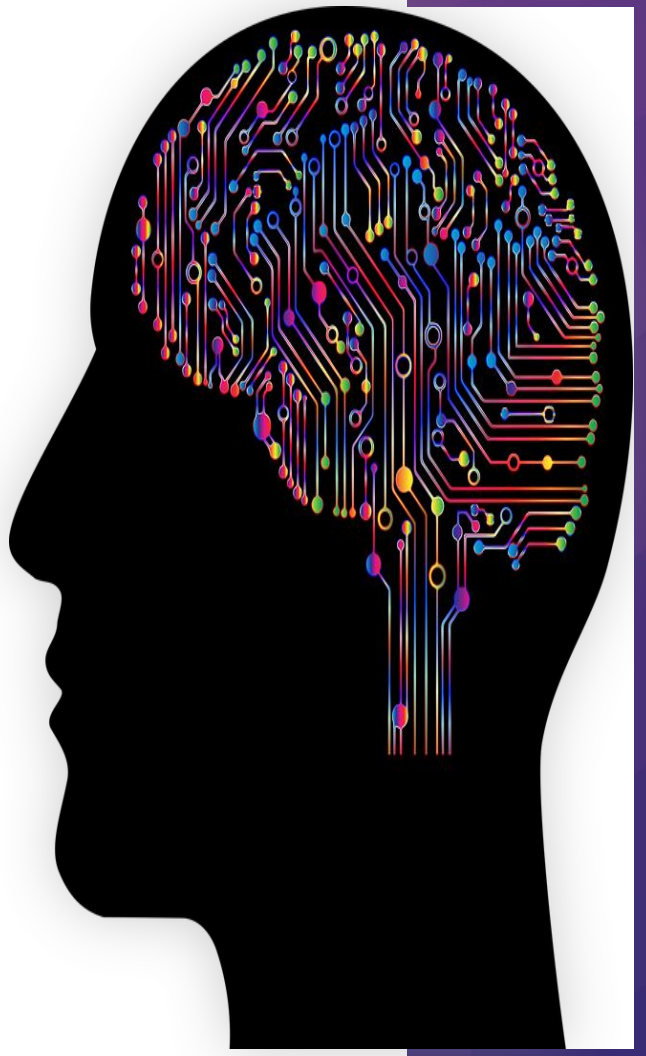
# Green demo environment



# Question Three:

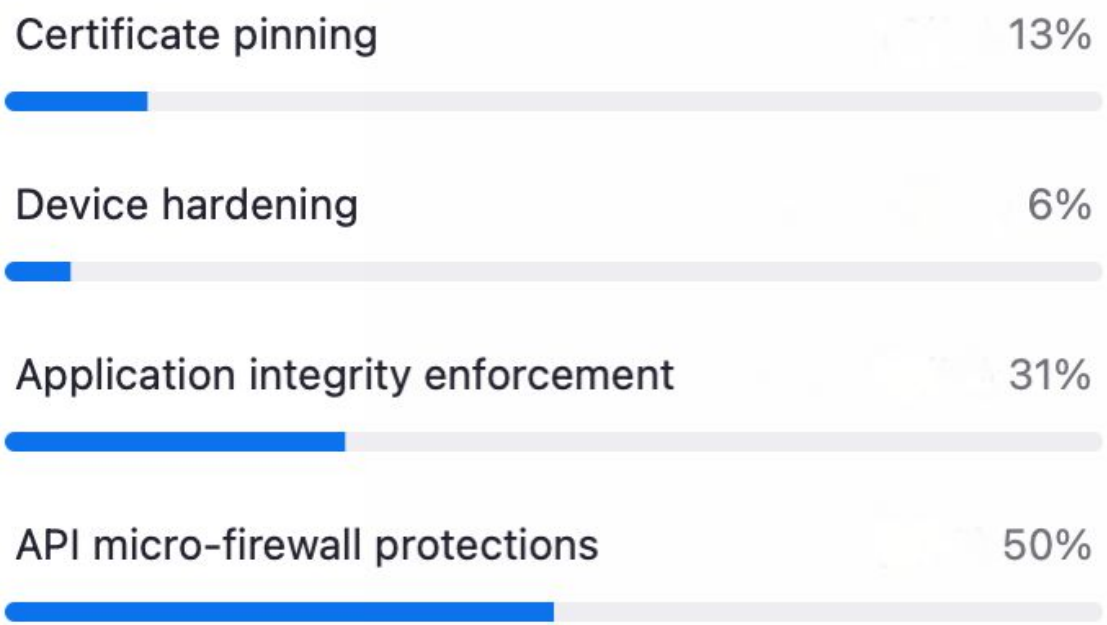
## What *single* protection do you feel would be most beneficial to your environment?

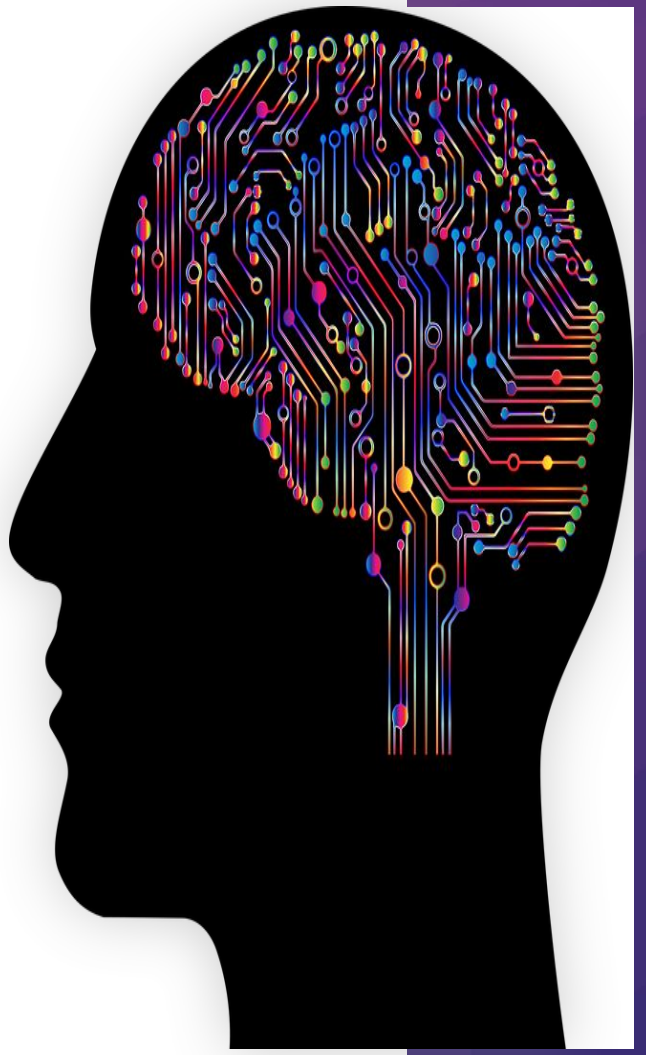
1. Certificate pinning
2. Device hardening
3. Application integrity enforcement
4. API micro-firewall protections



# Question Three:

## What *single* protection do you feel would be most beneficial to your environment?

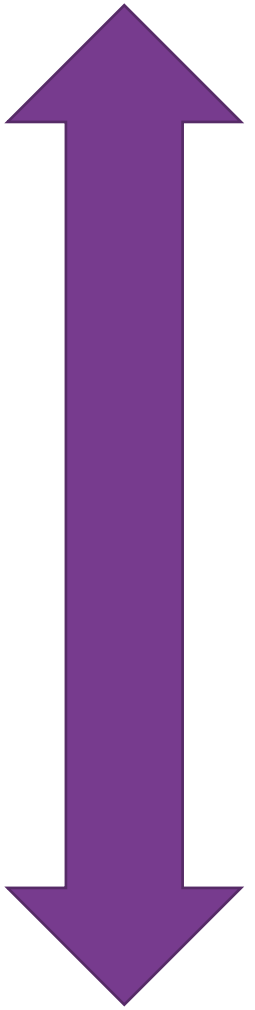


# Use cases / benefits

# Use cases

- A Bad Thing ™ has happened !
- Emerging threats
- Legacy APIs
- Defense in depth
- Changes to policy
- Central governance
- New features
- New development teams

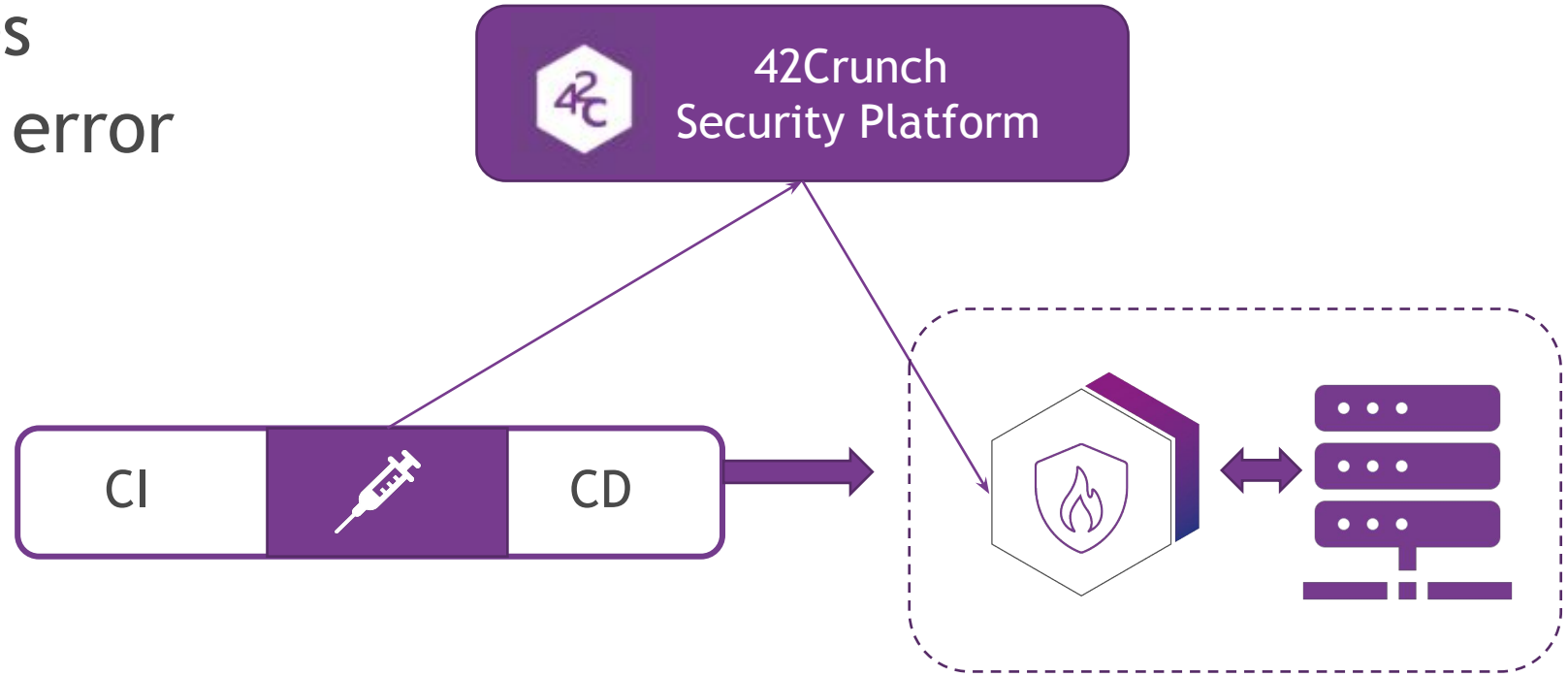


**Reactive**

**Proactive**

# Benefits

- Scale
- Central governance
- Segregation of duties
- Reduction in human error
- Hotfixes

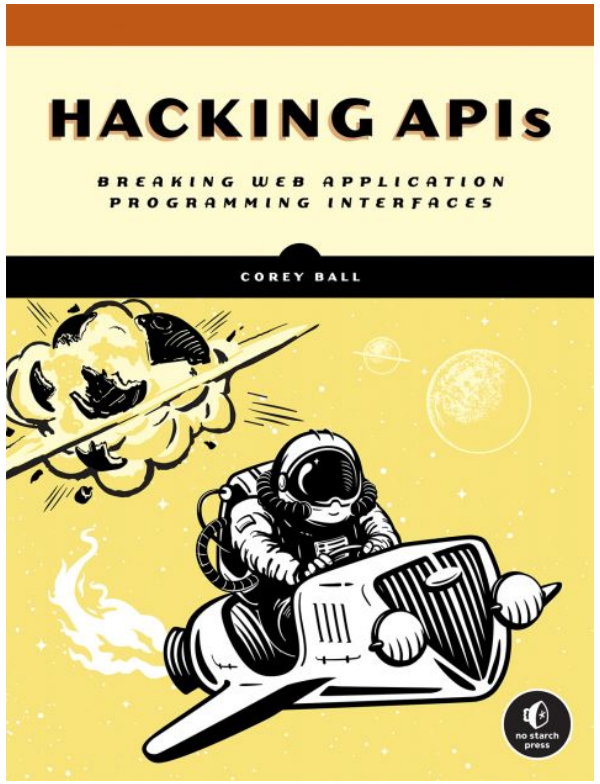


# Learning more

## APISecurity.io

https://apisecurity.io/

## “Hacking APIs”




https://nostarch.com/hacking-apis

## Awesome API Security


https://github.com/arainho/awesome-api-security

Upcoming activity

# Further Information

## OWASP API Security Top 10

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


## Industry Conference Den Haag, Netherlands. April 4-6

API CONFERENCE

## OpenAPI Editor - Free Download

https://42crunch.com/resources-free-tools/

## APIsecurity.io Weekly Newsletter

https://apisecurity.io/

APIsecurity.io