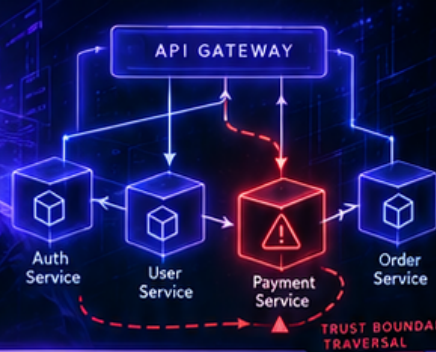


API PENTEST MODULE

API Penetration Testing

REST, GraphQL, WebSocket, gRPC-adjacent stacks — tested like an attacker, not a linter.

API SECURITY
REAL IMPACT



API TRUST
BROKEN AUTH
REAL IMPACT
MODERN SERVICES

BEYOND
SPECIFICATIONS
TOWARDS
REAL SECURITY.



API security testing that understands modern service-to-service trust, not just OWASP API Top 10 checkboxes.

 **PENTESTERRA**

APIP / TARGETS / SCAN Live Scan

Dashboard

API Pentest

Endpoints

GraphQL

WebSocket

Authentication

API Gateway

Attack Chain

Findings

Reports

Integrations

Settings

Target: https://api.acme-corp.com

API Penetration Testing (APIP)

Scanning...

New Scan

Endpoints Discovered
1,842
↑ +312 new

Verified Findings
27
6 critical

Auth Issues
43
JWT / OAuth / SAML

Attack Paths
7
Multi-service chains

Request / Response

Endpoints (1,842)

GraphQL

WebSocket

Auth Analysis

Gateway

Findings (27)

Request

POST https://api.acme-corp.com/v1/users/123/profile

Send

Headers

Params

Body

Auth

```
1 {
2   "userId": "123",
3   "email": "attacker@evil.com",
4   "role": "admin",
5   "isActive": true
6 }
```

Response

200 OK 284 ms

Pretty Raw Headers

```
1 {
2   "id": "123",
3   "name": "John Doe",
4   "email": "attacker@evil.com",
5   "role": "admin",
6   "isActive": true,
7   "updatedAt": "2024-11-15T10:24:31Z"
8 }
```

Finding CRITICAL

Broken Authorization (BOLA) Verified

User can modify another user's profile by changing the userId parameter.

PoC attached

Attack chain available

Affects multiple endpoints

Real business impact

View Finding →

Scan Progress

[INFO] Initializing API attack modules...

[INFO] Parsing OpenAPI/Swagger specifications...

[INFO] Discovering endpoints from JS bundles...

[INFO] Analyzing GraphQL schema (introspection)...

[INFO] Testing authentication mechanisms...

[INFO] Fuzzing parameters for IDOR/BOLA...

[INFO] Testing rate limiting and gateway controls...

[INFO] Launching active exploitation tests...

[SUCCESS] Vulnerability confirmed: BOLA (ID: APIP-2024-017)

[INFO] Continuing scan ...

Endpoint Discovery 100%

GraphQL Analysis 78%

Auth Testing 62%

Exploitation 45%

Attack Chain Analysis 28%

Discovered API Surface View All →

REST 1,230 endpoints (OpenAPI)

GraphQL 86 queries/mutations (Schema)

WebSocket 12 endpoints (Real-time)

gRPC 24 services (Proto files)



What it does

- ✓ **GraphQL:** introspection abuse, injection through nested queries
- ✓ **REST:** IDOR/BOLA, BFLA, mass assignment, versioning abuse, spec exposure (Swagger/OpenAPI auto-parsed for scope)
- ✓ **API gateway:** rate-limiting bypass, auth bypass, microservices & service-mesh trust boundaries
- ✓ **WebSocket:** CSWSH (cross-site WebSocket hijacking)
- ✓ **JWT:** alg:none bypass, JKU/XSU header injection, OAuth/SAML misuse
- ✓ **Auto-discovers** API surface from JS bundles, OpenAPI specs, and traffic — not just a manually-fed endpoint list



Why Pentesterra

- ✓ API context is not isolated from web — one findings model, one attack chain, one triage queue
- ✓ Active exploitation, not just static specification analysis



Modern APIs power modern business. We test them like real attackers do.

— PENTESTERRA

API SECURITY
REAL EXPLOITS
MODERN SERVICES
ATTACK CHAINS
REAL ADVANTAGE

THE API PENTEST WORKFLOW

1

Discover

Find API surface (JS, OpenAPI, traffic)

2

Map

Understand services & trust boundaries

3

Test

Run active exploitation tests

4

Verify

Confirm real impact with PoC

5

Correlate

Link findings into attack chains



OpenAPI & Swagger



GraphQL Introspection



JWT Analysis



API Gateway Testing



Microservices & Service Mesh



Active Exploitation



Attack Chain Correlation



AI / LLM Coverage

Test the APIs your business depends on.

Explore API Pentest →

PEOPLE
PROCESS
TECHNOLOGY
STRONGER
TOGETHER

API Security Tested **Like an Attacker, Not a Linter**

REST, GraphQL, WebSocket, gRPC-adjacent stacks - tested for real exploitability, not just OWASP API Top 10 checkboxes. API security testing that understands modern service-to-service trust.

4

Protocol families covered: REST, GraphQL, WebSocket, gRPC-adjacent

Auto-Discovered

API surface from specs, JS bundles, and live traffic

Active

Real exploitation, not static specification analysis

WHY NOW

Your API surface grew again this quarter, and half of it isn't in the OpenAPI spec anyone actually maintains. An undocumented endpoint with broken authorization is invisible to a spec-based scanner - and it's exactly what an attacker finds first.

WHAT IT COVERS

GraphQL & REST

Introspection abuse, injection through nested queries, and the authorization failures that make an endpoint exploitable, not just documented.

- GraphQL introspection abuse and injection through nested queries
- IDOR/BOLA, BFLA, mass assignment, versioning abuse
- OpenAPI/Swagger auto-parsed for scope, spec exposure checks

Authentication & Trust Boundaries

Modern APIs are built on service-to-service trust - Pentesterra tests where that trust actually breaks.

- JWT tampering: alg:none bypass, JKU/X5U header injection
- OAuth/SAML misuse
- API gateway: rate-limiting bypass, auth bypass, microservices & service-mesh trust boundaries

WebSocket & Real-Time

Real-time channels get the same active-exploitation treatment as REST endpoints, not a pass.

- CSWSH (cross-site WebSocket hijacking)
- Auth and session validation over persistent connections
- Discovered and tested alongside REST and GraphQL in the same run

Auto-Discovered Surface

The API surface is discovered the way an attacker finds it, not just from what you hand over.

- OpenAPI/Swagger specs, JS bundle analysis, and live traffic - not a manually-fed endpoint list
- REST, GraphQL, WebSocket, and gRPC-adjacent services enumerated automatically
- Every discovered endpoint is in scope, not just the documented ones

WHY PENTESTERRA

API context is not isolated from web - one findings model, one attack chain, one triage queue. A broken-authorization finding on an internal API and a phishing-captured credential can chain into the same reported path.

FREQUENTLY ASKED QUESTIONS

Does this replace a linter or a spec-conformance check?

No, and it isn't meant to. A linter checks your OpenAPI spec against a style guide. This actively exploits the endpoints your spec describes (and the ones it doesn't) - broken authorization, injection, JWT tampering - the way an attacker would, with a working proof of concept attached.

How is the API surface discovered?

Automatically, from three sources: OpenAPI/Swagger specifications, JavaScript bundle analysis, and observed live traffic - so undocumented or forgotten endpoints are in scope, not just the ones in your spec file.

Does it test GraphQL and WebSocket, or just REST?

All of them in the same run: REST, GraphQL (introspection abuse and nested-query injection), WebSocket (including CSWSH), and gRPC-adjacent services.

Is API testing connected to the rest of the platform?

Yes. Findings share the same model as web and network pentest results, feed the cross-domain Attack Chain graph, and get the same non-destructive exploit verification as every other module.

Test the APIs your business depends on

Point Pentesterra at your API surface - REST, GraphQL, WebSocket, all in one run.

info@pentesterra.com
pentesterra.com/web-app-pentest