



OWASP

Open Web Application
Security Project

REST API安全测试的思路

基于Swagger的REST API自动化安全测试实践

贾玉彬 Gary

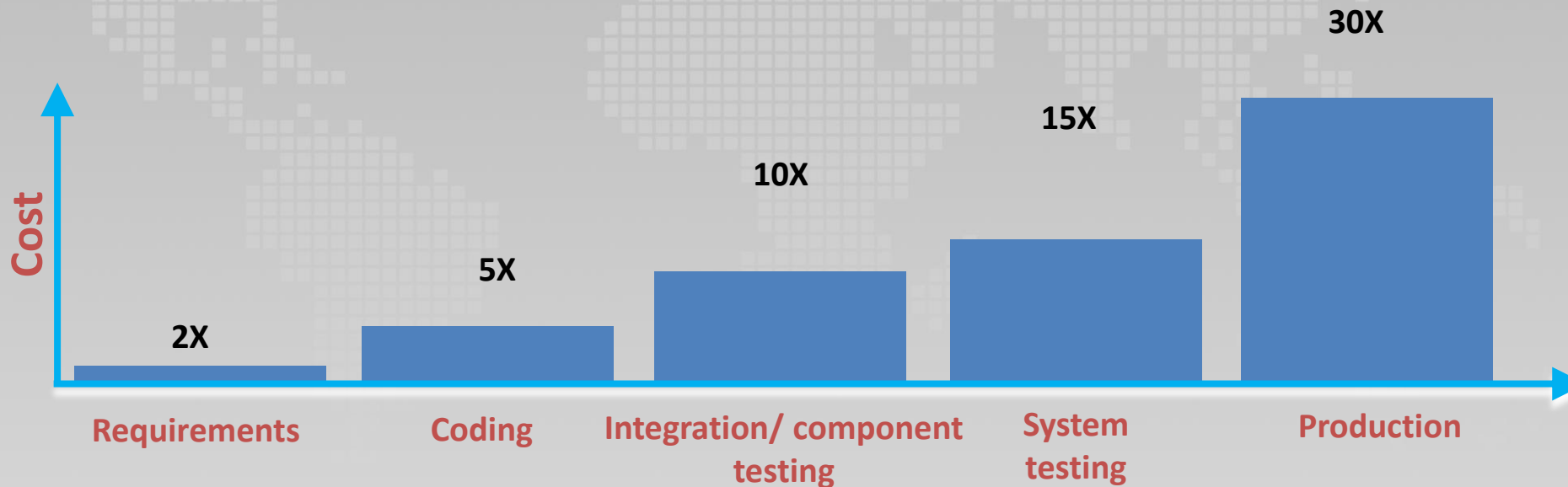
Gary_jia@rapid7.com

大纲

- 由应用安全说起
- 传统REST API安全测试
- 基于Swagger的REST API 的安全测试



在SDLC中尽早发现和修复安全问题



Source: NIST



OWASP
Open Web Application
Security Project

不同的团队不同的目标



DevSecOps

“Everyone is responsible for security” with the goal of safely distributing security decisions at speed and scale

It does not have to be like this:

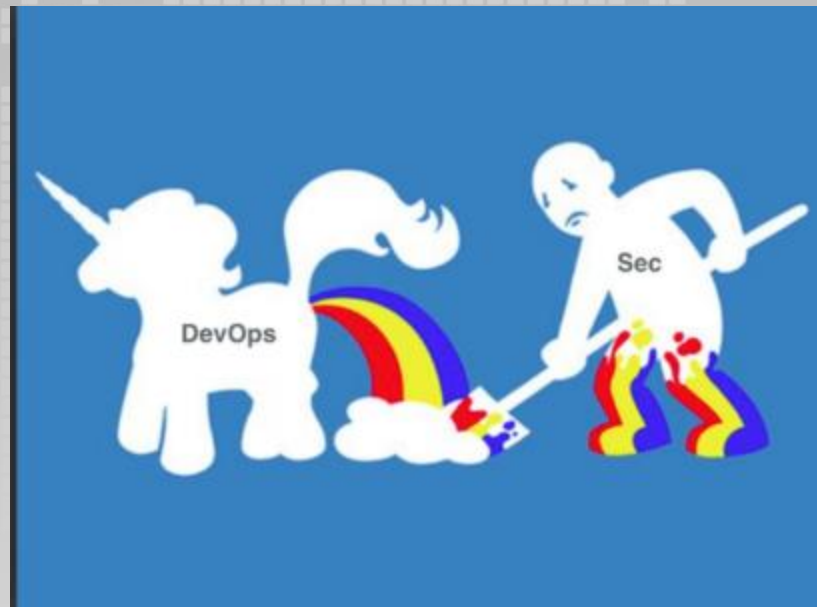


Image : Pete Cheslock at #DevOpsDaysAustin.



OWASP
Open Web Application
Security Project

传统的REST API安全测试



OWASP
Open Web Application
Security Project

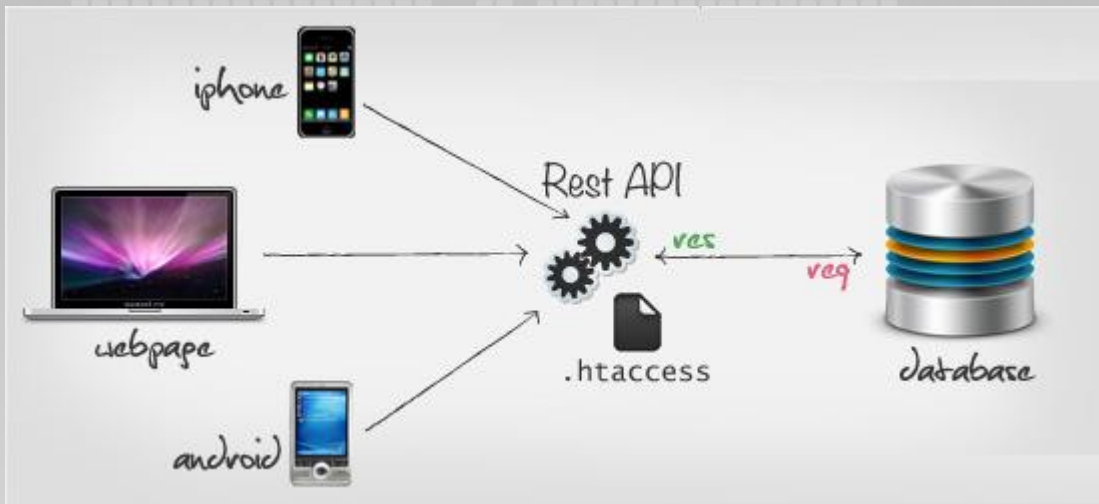
REST API

- REST – REpresentational State Transfer
- REST描述的是客户端和服务端的一种交互形式，REST本身不实用，实用的是如何设计RESTful风格 API
- 传输资源的形式一般是JSON和XML



RESTful结构的好处

RESTful通过一套统一的接口为Web, iOS和Android提供服务



*图片来自互联网



OWASP
Open Web Application
Security Project

REST API安全问题会带来的后果

- 接口被滥用消耗系统资源
- 数据泄露
- 伪造/篡改数据
- 应用被仿制
- 引入其他安全问题



REST API是可被攻击的

- REST API是可以被常用的攻击方法进行攻击的，如注入、XSS、CSRF、XML Entity攻击等等



REST API测试

- 通常没有WEB界面与之进行交互，通过HTTP的各种终端工具或自己编写HTTP客户端脚本与服务端通讯来进行测试
- 通过变化API调用参数组合，变化API功能调用顺序，实现全面和各种复杂组合的调用
- 针对安全测试也存在同样的问题



以移动应用为例的测试过程（手工）

- 安装配置代理
- 配置测试移动设备
- 准备测试用数据
- 手工测试



安装并设置代理、配置移动设备

The image is a collage of screenshots from OWASP ZAP 2.4.3, illustrating the setup and use of the tool for mobile device security testing.

- Local Proxy Configuration:** The 'Options' dialog box is shown with the 'Local Proxy' tab selected. The 'Address (eg localhost, 127.0.0.1)' is set to '192.168.1.109' and the 'Port (eg 8080)' is set to '8081'. The 'Always unzip gzipped content' checkbox is checked.
- Edit Access Point:** The 'Edit access point' dialog box is shown for a T-Mobile US device. The 'Name' is 'T-Mobile US', the 'APN' is 'epc.tmobile.com', the 'Proxy' is '192.168.1.109', the 'Port' is '8081', and the 'Username' is 'none'.
- Network Traffic Logs:** The 'History' tab shows a list of HTTP requests and responses. The table below represents the data shown in the logs:

ID	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. B.	Highest P.	No. Tags
1	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1&per_page=1000 HTTP/1.1	401	Unauthorized	30	85 bytes	Medium	
4	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1	401	Unauthorized	30	85 bytes	Medium	
5	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1	401	Unauthorized	20	85 bytes	Medium	
6	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1	200	OK	47	113 bytes	Medium	
7	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1	200	OK	88	1524 bytes	Medium	
8	06/02/16 18:14	GET	http://192.168.1.109/api/category?page=1	200	OK	1	848 bytes	Medium	
15	06/02/16 18:15	GET	http://192.168.1.109/api/category?page=1	200	OK	1	848 bytes	Medium	
16	06/02/16 18:15	POST	http://192.168.1.109/api/category?page=1	200	OK	49	200 bytes	Medium	

- Mobile App Interface:** A screenshot of a mobile app interface showing a list of products. The products listed are:

- Martha Stewart Crafts Garland, Pink Pom Pom Small, Price: \$9.0
- Martha Stewart Crafts Modern Festive Pennant Garland, 12ft, Price: \$5.6
- Martha Stewart Crafts Pom Poms, Pink, 2 Sizes

At the bottom of the app interface, there are buttons for '< Back', 'Go to cart', and 'Add to cart'.

手工测试REST API（SQL注入为例）

- 测试是否存在注入漏洞
- 例如使用PUT方法更新用户档案
- 请求中包含参数值（first_name）

URL: <http://192.168.202.131/api/user/1>

Method: PUT

Parameter name: first_name

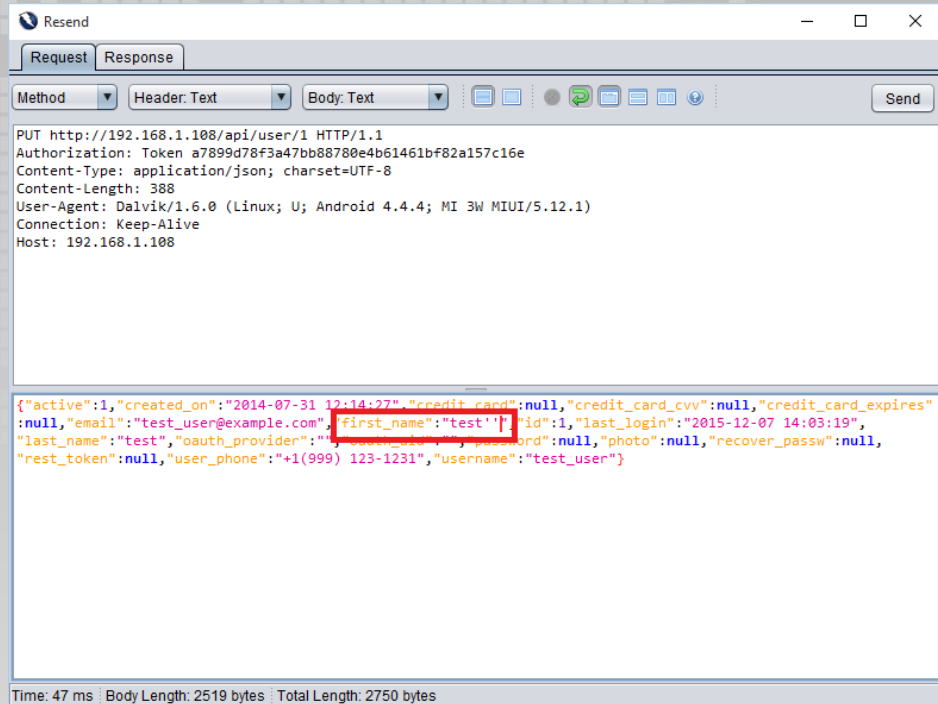
Attack values: test''



手工测试REST API（SQL注入为例）

Request 1

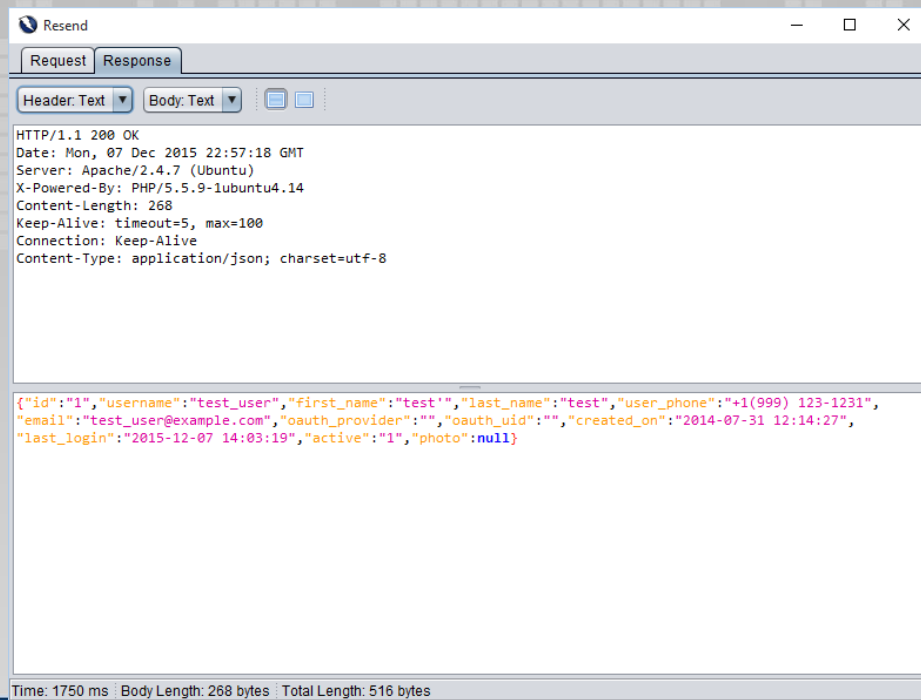
使用test”把两个单引号（'）注入到请求



手工测试REST API（SQL注入为例）

Response 1

用户档案成功提交



手工测试REST API（XSS为例）

- 测试是否存在XSS漏洞
- 例如使用PUT方法更新用户档案
- 请求中包含参数值（first_name）

URL: <http://192.168.202.131/api/user/1>

Method: PUT

Parameter name: first_name

Attack values: <script>alert(1)</script>

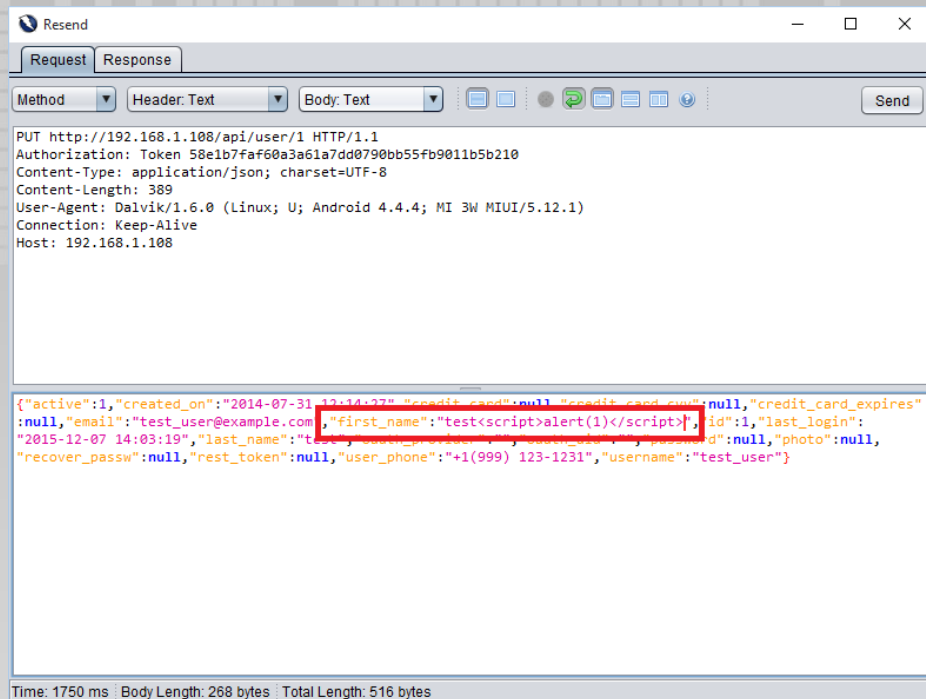


手工测试REST API（XSS为例）

Request

在请求中使用

`<script>alert(1)</script>`把一个脚本标签注入到first_name参数值



手工测试REST API (XSS为例)

Resend

Request Response

Header: Text Body: Text

```
HTTP/1.1 200 OK
Date: Mon, 07 Dec 2015 23:02:28 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.14
Content-Length: 293
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
```

```
{
  "id": "1",
  "username": "test_user",
  "first_name": "test<script>alert(1)</script>",
  "last_name": "test",
  "user_phone": "+1(999) 123-1231",
  "email": "test_user@example.com",
  "oauth_provider": "",
  "oauth_uid": "",
  "created_on": "2014-07-31 12:14:27",
  "last_login": "2015-12-07 14:03:19",
  "active": "1",
  "photo": null
}
```

Time: 1578 ms Body Length: 293 bytes Total Length: 541 bytes

192.168.1.108/account#profile

HACKAZON

All Search products...

The page at 192.168.1.108 says:

1

☐ Prevent this page from creating additional dialogs.

OK

Your account Logout

Search!

My Account

[Home](#) / [My Account](#)

[My Latest Orders](#) [Profile](#)

Order №	Date	Payment Method	Shipping Method	Status
10002797	12/04/2015	creditcard	mail	complete
10000256	12/02/2015	creditcard	mail	complete
10000512	12/02/2015	creditcard	mail	complete
10001024	12/02/2015	creditcard	mail	complete
10001280	12/02/2015	creditcard	mail	complete

[Go to my orders](#)



使用DAST工具测试REST API

- 手工测试REST API是个很具挑战性的工作
- 大部分的DAST工具都需要training mode对REST API进行测试（使用复杂的JSON、XML、GWT结构）
- 方法：导入预先录制的流量文件进行测试



以AppSpider + BurpSuite为例

- BurpSuite使用广泛
- AppSpider可以接受各种录制的流量并且可以识别JSON、XML、GWT、AMF参数和值，不需要使用training mode



DEMO



OWASP
Open Web Application
Security Project

基于SWAGGER的REST API的安全测试



OWASP
Open Web Application
Security Project

基于Swagger的REST API

- Swagger是一个简单又强大的REST API文档生成工具
- 标准的、语言无关的，通过它计算机和人类无需阅读代码、文档或者监测网络流量就能发现并理解Web服务



Swagger REST API

```
{
  "swagger": "2.0",
  "info": {
    "version": "0.0.0",
    "title": "Hackazon modern web app"
  },
  "host": "192.168.1.108:8080",
  "schemes": [
    "http"
  ],
  "paths": {
    "/category": {
      "get": {
        "parameters": [
          {
            "name": "page",
            "in": "query",
            "schema": {
              "type": "string"
            }
          },
          {
            "name": "per_page",
            "in": "query",
            "schema": {
              "type": "string"
            }
          },
          {
            "name": "Authorization",
            "in": "header",
            "required": true,
            "default": "Token token_name",
            "schema": {
              "type": "string"
            }
          }
        ],
        "responses": {
          "200": {
            "description": "Response will be in JSON",
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/ProductKeyDetails"
              }
            }
          }
        },
        "description": "Request the creation of a new ProductKey"
      }
    }
  },
  "definitions": {
    "ProductKeyDetails": {
      "type": "object",
      "properties": {
        "accessToken": {
          "type": "string",
          "required": true
        },
        "emailAddress": {
          "type": "string",
          "required": true
        },
        "licenseType": {
          "type": "string",
          "required": true
        },
        "licenseExpirationTime": {
          "type": "string",
          "required": true
        },
        "accountExpirationTime": {
          "type": "string",
          "required": true
        },
        "customerName": {
          "type": "string",
          "required": true
        },
        "firstName": {
          "type": "string",
          "required": true
        },
        "lastName": {
          "type": "string",
          "required": true
        }
      }
    }
  }
}
```

```
1 swagger: '2.0'
2 info:
3   version: 1.0.0
4   title: AppSpider License and Release API
5   description:
6     ## Processes Licenses and Release Information
7     Used by AppSpider and related services.
8   schemas:
9     - https
10    consumes:
11      - application/json
12    produces:
13      - application/json
14    host: download.appspider.rapid7.com
15    basePath: /
16    paths:
17      /api/createLicense:
18        post:
19          responses:
20            200:
21              description: Response will be in JSON
22              schema:
23                type: array
24                items:
25                  $ref: '#/definitions/ProductKeyDetails'
26          description: Request the creation of a new ProductKey
27          parameters:
28            - name: AccessToken
29              required: true
30              in: header
31              description: Authentication
32              type: string
33            - name: emailAddress
34              required: true
35              in: formData
36              description: User email address
37              type: string
38            - name: licenseType
39              required: true
40              in: formData
41              description: Product name
42              type: string
43              enum:
44                - AppSpiderPro
45                - AppSpiderExpress
46                - AppSpiderEval
```

Paths

/api/createLicense

POST /api/createLicense

Description

Request the creation of a new ProductKey

Parameters

Name	Located in	Description	Required	Schema
AccessToken	header	Authentication	Yes	string
emailAddress	formData	User email address	Yes	string
licenseType	formData	Product name	Yes	string
licenseExpirationTime	formData	Can be any date format you prefer	Yes	string
accountExpirationTime	formData	Can be any date format you prefer. Defaults to licenseExpirationTime	No	string
customerName	formData	Company name	No	string
firstName	formData	First Name	No	string
lastName	formData	Last Name	No	string

Responses

Code	Description	Schema
200	Response will be in JSON	{ ProductKeyDetails { } }

Try this operation



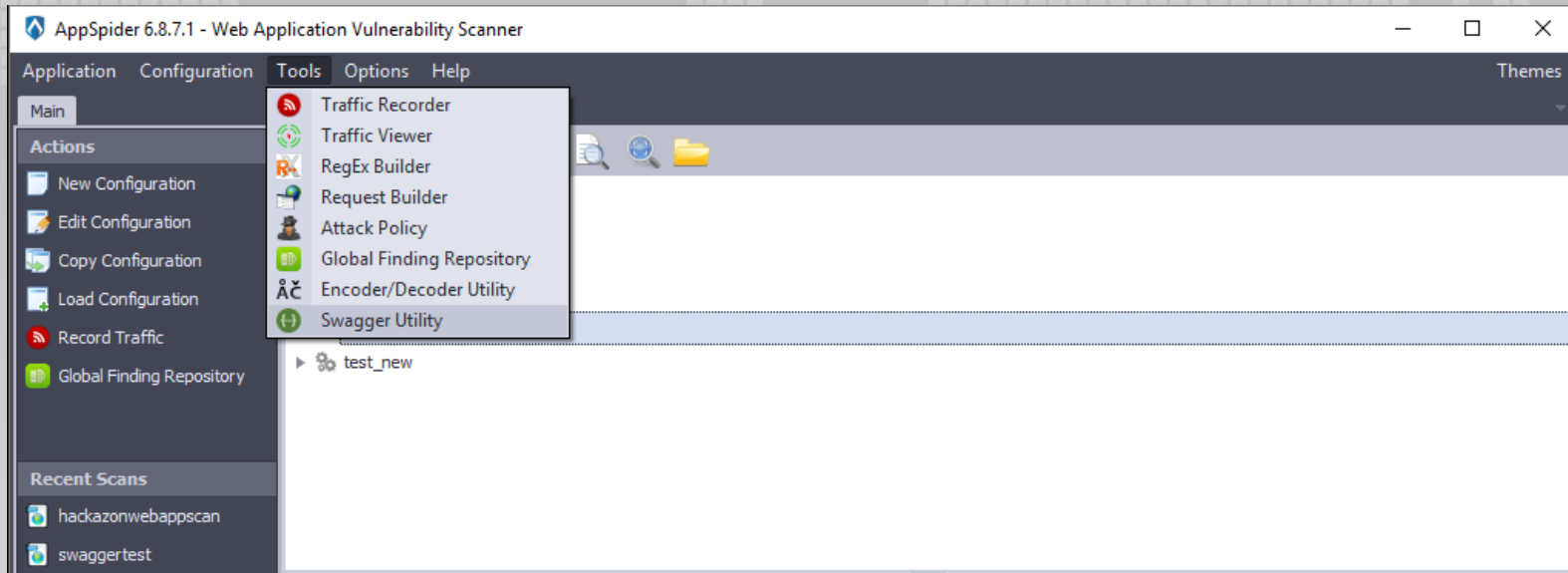
OWASP
Open Web Application
Security Project

DEMO

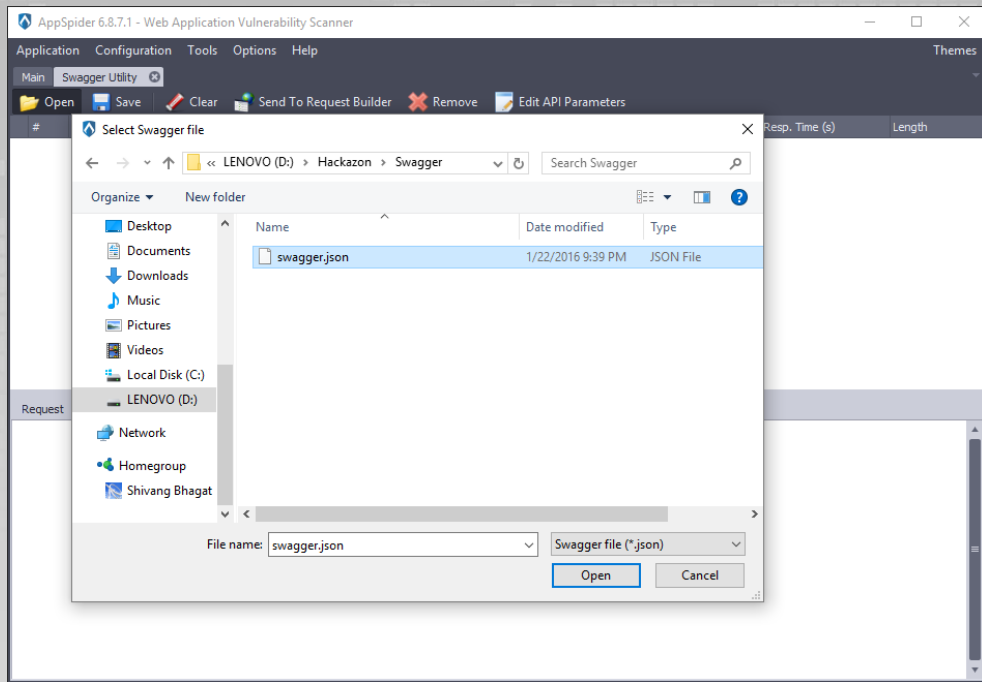


OWASP
Open Web Application
Security Project

运行AppSpider并选择Swagger Utility



导入Swagger JSON文件



API function calls

AppSpider 6.8.7.1 - Web Application Vulnerability Scanner

Main Swagger Utility

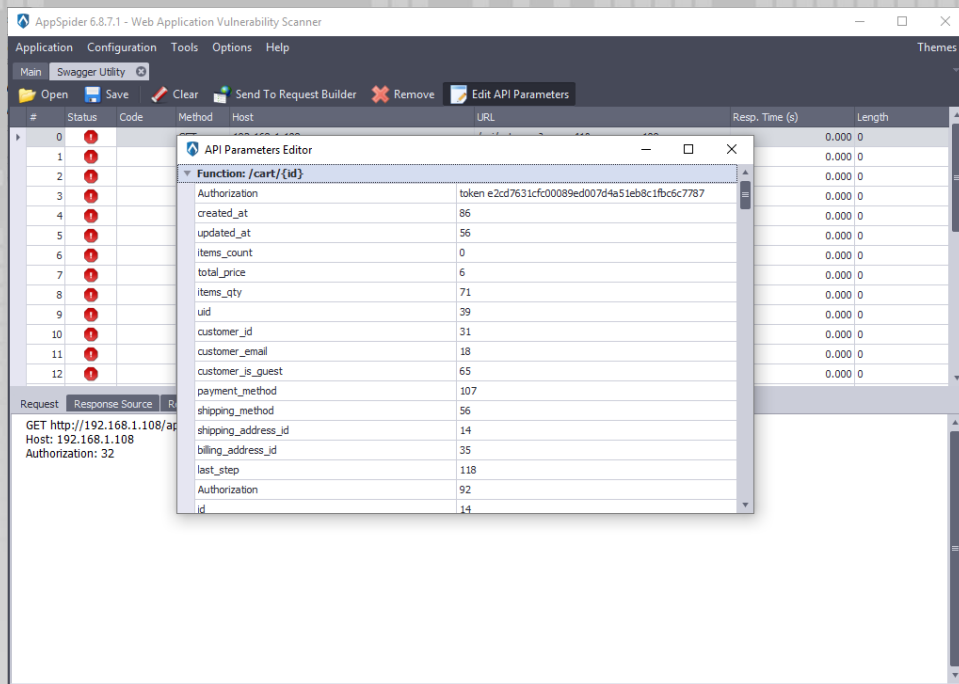
Open Save Clear Send To Request Builder Remove Edit API Parameters

#	Status	Code	Method	Host	URL	Resp. Time (s)	Length
0	1		GET	192.168.1.108	/api/category?page=41&per_page=109	0.000	0
1	1		GET	192.168.1.108	/api/category/126	0.000	0
2	1		GET	192.168.1.108	/api/user/107	0.000	0
3	1		PUT	192.168.1.108	/api/user/110	0.000	0
4	1		GET	192.168.1.108	/api/user/me	0.000	0
5	1		GET	192.168.1.108	/api/product?page=66&categoryID=16	0.000	0
6	1		GET	192.168.1.108	/api/order?page=18	0.000	0
7	1		POST	192.168.1.108	/api/order	0.000	0
8	1		GET	192.168.1.108	/api/order/47	0.000	0
9	1		GET	192.168.1.108	/api/cart/my?uid=70	0.000	0
10	1		PUT	192.168.1.108	/api/cart/%7Bid%7D	0.000	0
11	1		DELETE	192.168.1.108	/api/cart/14	0.000	0
12	1		POST	192.168.1.108	/api/cart/items	0.000	0

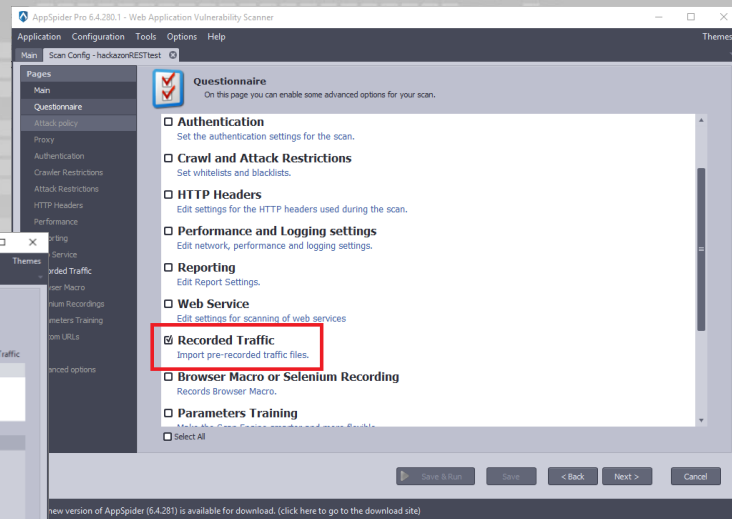
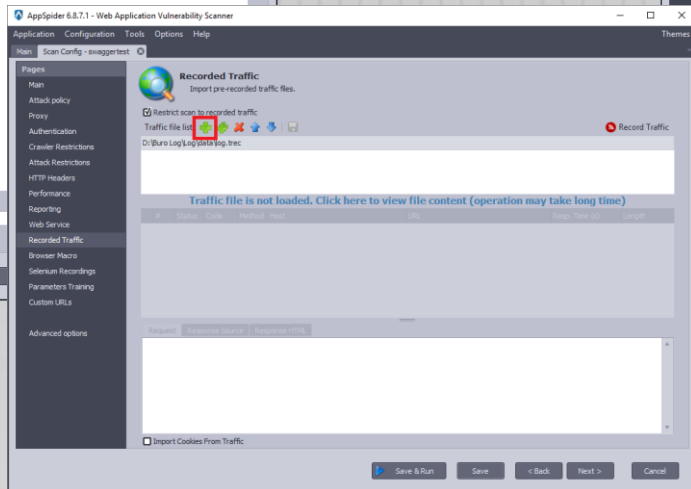
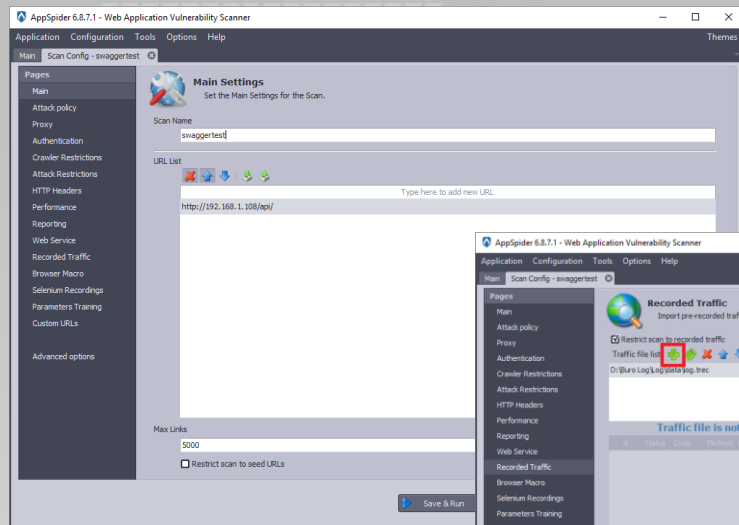
Request Response Source Response HTML

GET http://192.168.1.108/api/category?page=41&per_page=109 HTTP/1.1
Host: 192.168.1.108
Authorization: 32

操作API参数



创建新的扫描配置





Scan Results

Scan Name:

Webscantest-includeAPIs-reactjs

Date:

8/24/2016 11:24:23 PM

Authenticated User:

testuser

Total Links / Attackable Links:

416 / 416

Target URL:

http://webscantest.com

Reports:

Select Report

Summary

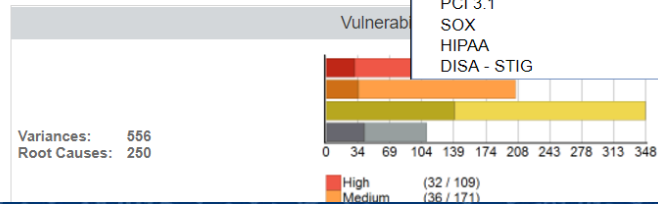
A **partial** scan was performed.

- We crawled **416 links** for which we performed **39,701** requests.
- There are **556 vulnerabilities** detected which can be remediated.
- There are an additional 98 findings such as Best Practices, PCI 3.1, SOX, HIPAA, DISA - STIG.

There were issues affecting the scan:

- More than **5%** of the requests failed in this scan. This is a significant issue as it can cause a loss of remediation labor by 55%.
- There were issues affecting the scan: server and significantly compromises the assessment value of this scan. Try setting a lower number (i.e. half to two-thirds of the current value)
- We detected loss of session **1 time**. While it is true that this generally does not cause loss of session. This suggests your session expiration policies might be a bit too fragile, compromising usability

Vulnerabilities



Security Status - Partial

Vulnerability

Best Practice

Exposure

Select Report

Scan Results Home

Executive Summary

Remediation Summary

Application Threat Modeling

Reflection Report

Vulnerabilities

Remediation Reports:

Application Developer

Application Developer by URL

Server Administrator

Database Administrator

Database Administrator by URL

Best Practices and Compliance Reports:

Best Practices

Privacy

PCI 3.1

SOX

HIPAA

DISA - STIG

Vulnerability Reports

Total Vulnerabilities	250 Root Causes
Application & Database	243 Root Causes
Server Administrator	7 Root Causes



总结

- REST API安全测试很重要
- 尽量使用自动化/半自动化工具进行测试
- 使用Swagger生成REST API文档
- 使用支持Swagger的DAST进行安全测试



谢谢



OWASP
Open Web Application
Security Project