Security Testing For RESTful Applications

Eyal Fingold & Ofer Shezaf, HP Enterprise Security Products

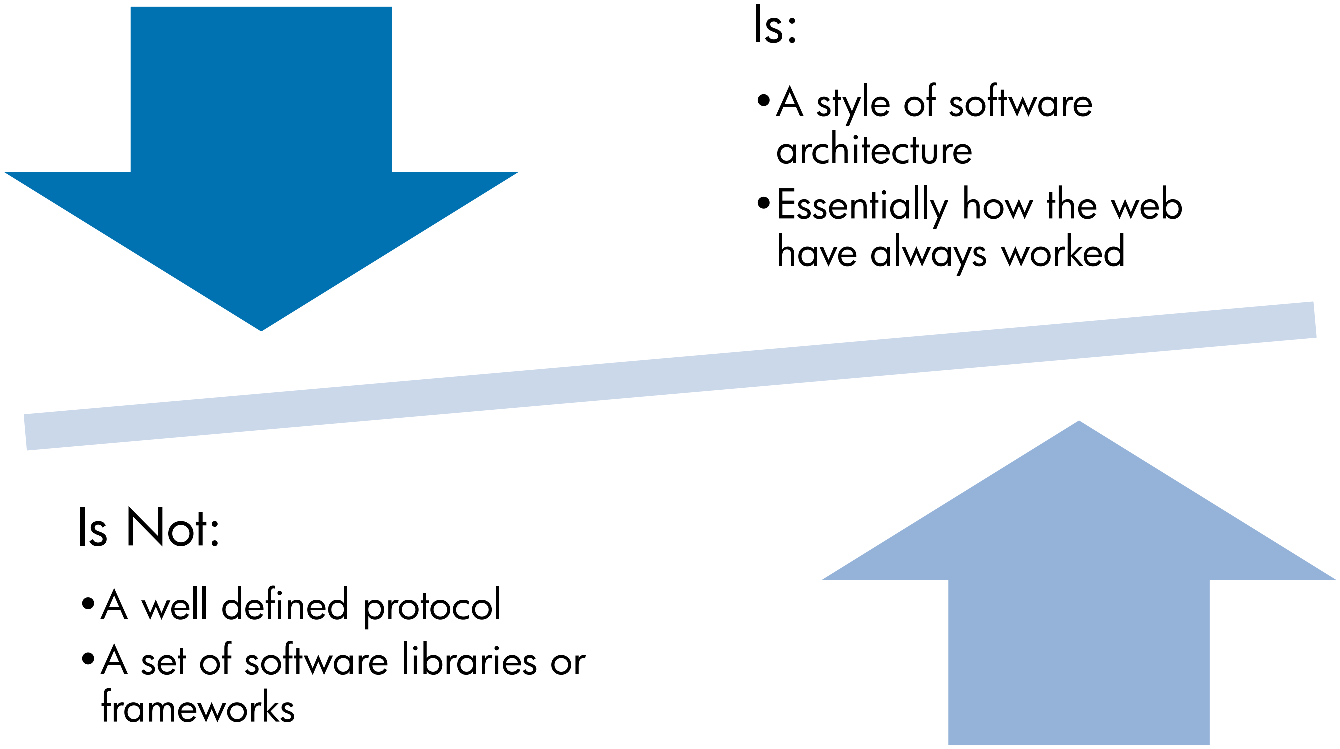
Agenda

- What are RESTful services (REST)?
- Security Issues in REST
- Challenges in security testing for REST

What is REST?

So What REST?

Representational State Transfer (REST) is a style of software architecture for distributed systems such as the World Wide Web (but not just Web)

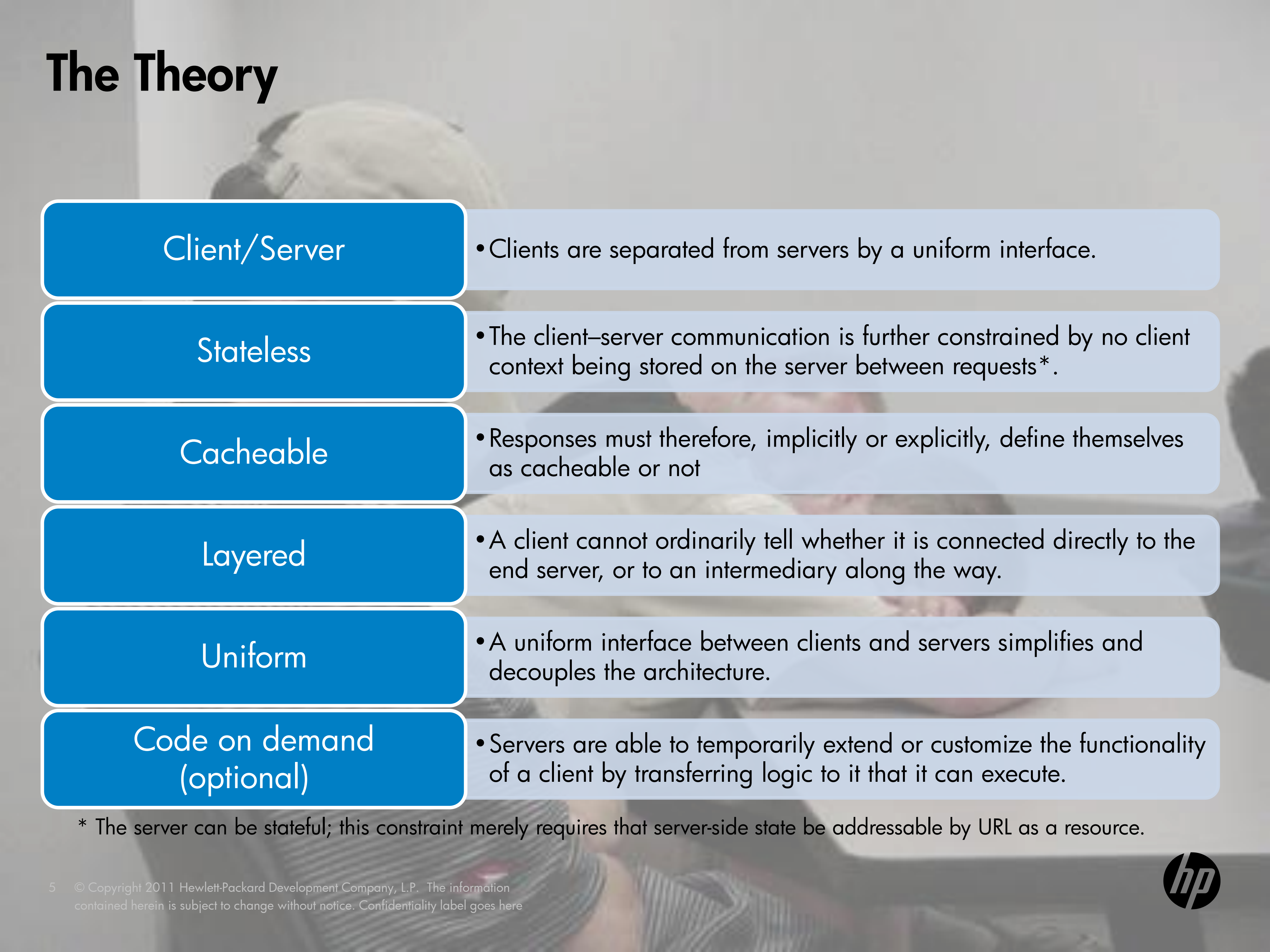


Is:

- A style of software architecture
- Essentially how the web have always worked

Is Not:

- A well defined protocol
- A set of software libraries or frameworks



The Theory

Client/Server	• Clients are separated from servers by a uniform interface.
Stateless	• The client–server communication is further constrained by no client context being stored on the server between requests*.
Cacheable	• Responses must therefore, implicitly or explicitly, define themselves as cacheable or not
Layered	• A client cannot ordinarily tell whether it is connected directly to the end server, or to an intermediary along the way.
Uniform	• A uniform interface between clients and servers simplifies and decouples the architecture.
Code on demand (optional)	• Servers are able to temporarily extend or customize the functionality of a client by transferring logic to it that it can execute.

* The server can be stateful; this constraint merely requires that server-side state be addressable by URL as a resource.

So What RESTful services?

is a simple web service implemented using HTTP and the principles of REST.

It is a collection of resources, with three defined aspects:

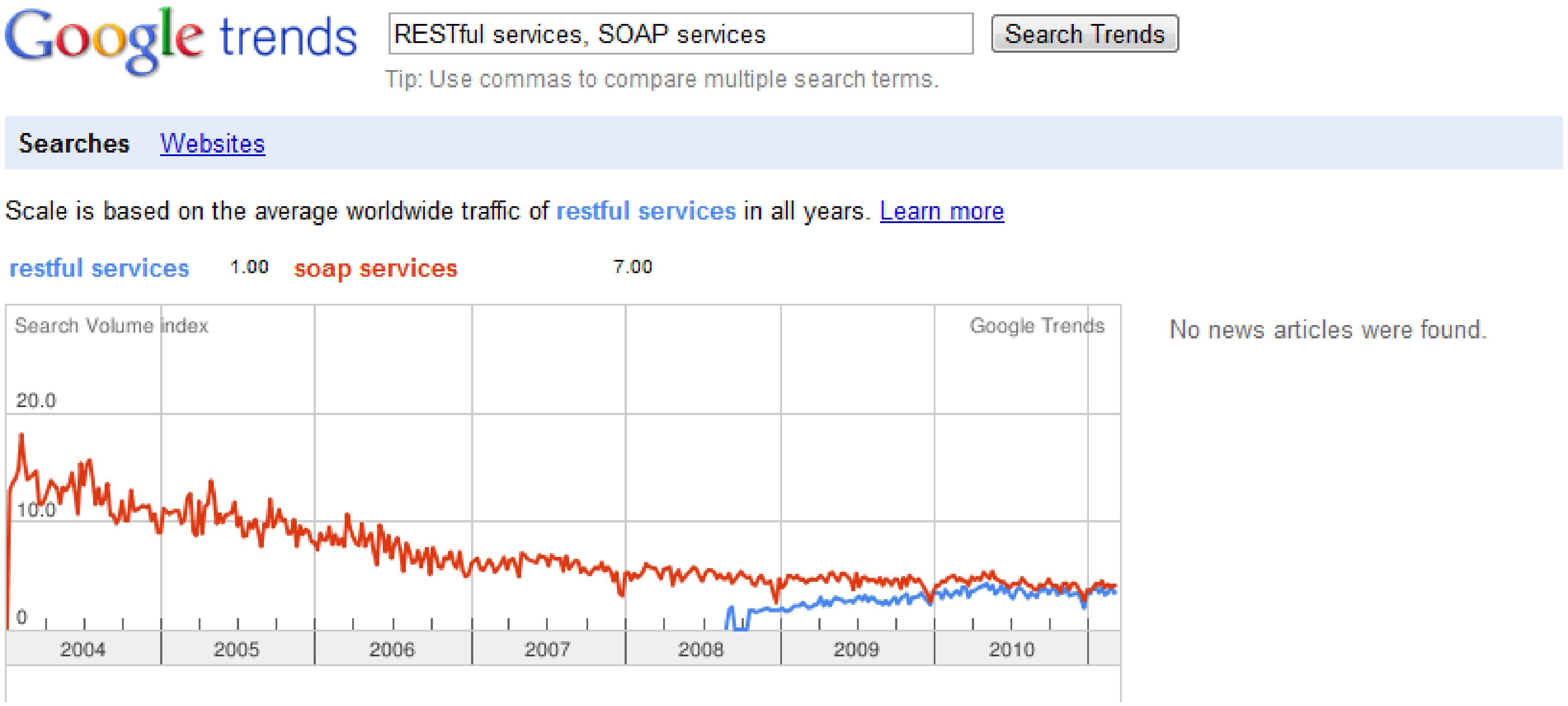
- **URI** for the web service, such as http://example.com/resources/

- The Internet media type of the **data** supported by the web service. This is often JSON, XML or YAML but can be any other valid Internet media type.

- The set of **operations** supported by the web service using HTTP methods (e.g., POST, GET, PUT or DELETE, HEAD etc…).

It's Up and Coming!

But what is it?


Google trends
RESTful services, SOAP services
Search Trends
Tip: Use commas to compare multiple search terms.
Searches
Websites

Scale is based on the average worldwide traffic of restful services in all years. Learn more

No news articles were found.

The Pitch for REST

We are tired of SOAP and WSDL

Would you like something cleaner than SOAP? Something less impenetrable than WSDL? Something less confusingly intertwingled than the various WS-* bafflegab standards? … Say, just what is this Web Services jazz anyhow?

Let's just get return to basics

It's all No Problem. It's all Easy as Pi. REST isn't some obscure thing that nobody supports; it's the way the Web already works, just formalized a bit and with some do's and don'ts.

(John Cowan)

Who Uses REST?

Google

ebay

Google buzz

OAUTH

Dropbox
www.getdropbox.com

YouTube

amazon.com

Microsoft SharePoint 2010

RESTful services frameworks

More than 35 frameworks covering most platforms:

- Ruby
- Java
- .Net (C#, VB)
- PHP
- Perl
- Python
- C++
- etc…

& Mobile, Mobile, Mobile…

In Practice

HTML 1.1 is essentially a RESTful protocol

SOAP Request example:

```
GET /StockPrice HTTP/1.1
Host: example.org
Content-Type: application/soap+xml;
charset=utf-8 Content-Length: nnn
<?xml version="1.0"?>
<env:Envelope
xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:s="http://www.example.org/stock-service">
        <env:Body>
                <s:GetStockQuote>
                        <s:TickerSymbol>HPQ
</s:TickerSymbol>
                </s:GetStockQuote>
        </env:Body>
</env:Envelope>
```

The same request, the REST way:

```
GET /StockPrice/HPQ  HTTP/1.1
Host: example.org
Accept: text/xml
Accept-Charset: utf-8
```

However…

It often doesn't look like your typical Web (1 or 2) application

```
PUT /destinationObject HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
x-amz-copy-source: /source_bucket/sourceObject
x-amz-metadata-directive: metadata_directive
x-amz-copy-source-if-match: etag
x-amz-copy-source-if-none-match: etag
x-amz-copy-source-if-unmodified-since: time_stamp
x-amz-copy-source-if-modified-since: time_stamp
<request metadata>
Authorization: signatureValue
Date: date
```

Parameters in Headers

None Standard Parameters/Method

```
PUT /ObjectName?acl HTTP/1.1
Host: BucketName.s3.amazonaws.com
Date: date
Authorization: signatureValue

<AccessControlPolicy>
  <Owner>
    <ID>ID</ID>
    <DisplayName>EmailAddress</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org
        <ID>ID</ID>
        <DisplayName>EmailAddress</DisplayN
      </Grantee>
      <Permission>Permission</Permission>
```

None Standard AAA

REST Security

REST Security Overview

- No standard security mechanism similar to SOAP Web Services (WS-*)
- Most session management methods are not REST oriented:
  - REST is supposed to be stateless.
  - However often standard Web practices are used.
- (Over)relying on:
  - SSL
  - HTTP Authentication (Basic!, Digest or custom headers)

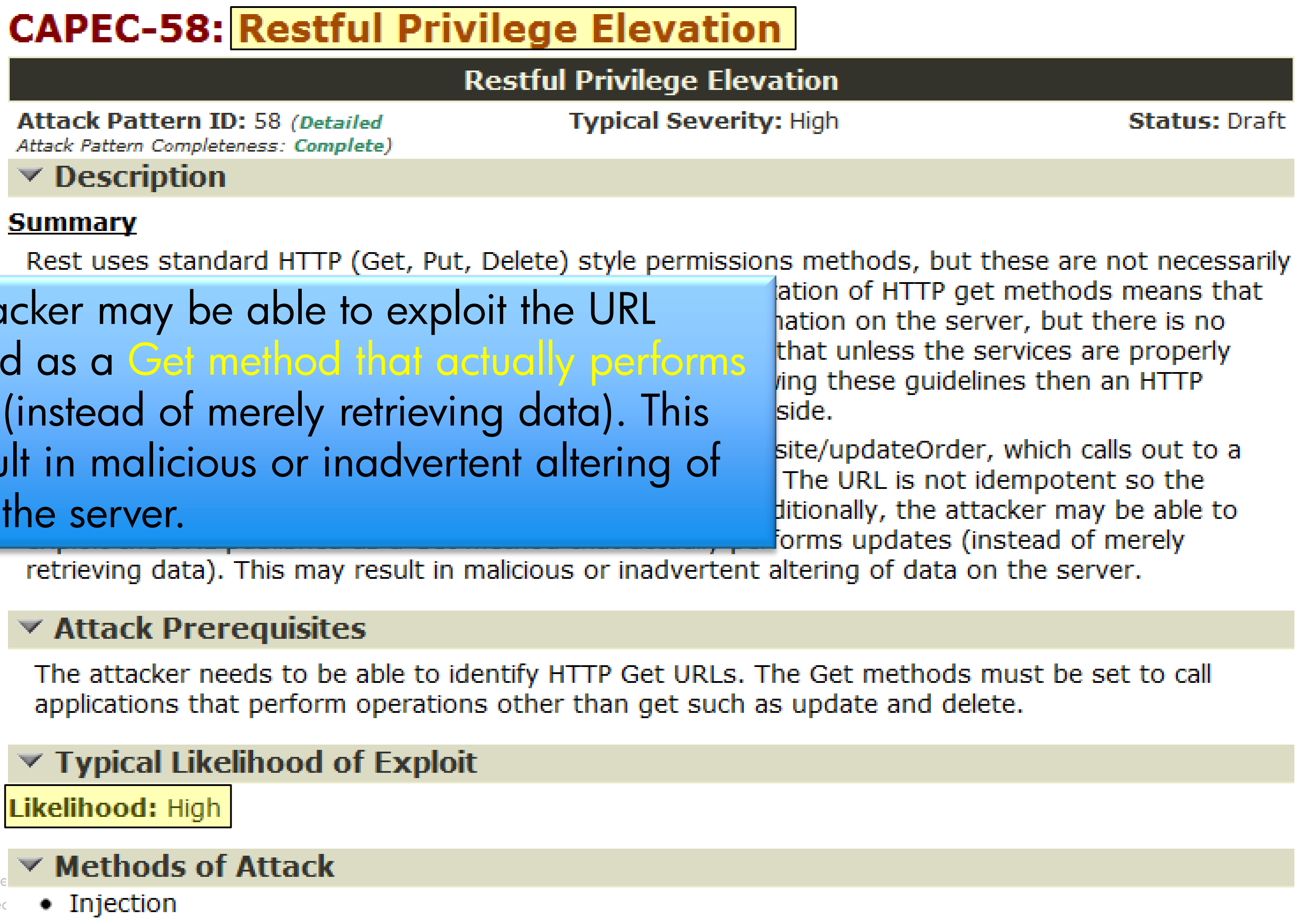
- SSO
  - Web app calling REST services

Think like a developer..
"WS-* has tons of intricate security standards. REST does not have these. We are going with REST"
Woo-hoo!

Think like a security administrator..
"WS-* has tons of intricate security standards. REST does not have these. We are going with REST"
Oh no!

Are There Any RESTful specific Vulnerabilities?

Well, it seems the most common attack vector is a REST one…



CAPEC-58: Restful Privilege Elevation

Restful Privilege Elevation

Attack Pattern ID: 58 *(Detailed Attack Pattern Completeness: Complete)* **Typical Severity:** High **Status:** Draft

Description

Summary

Rest uses standard HTTP (Get, Put, Delete) style permissions methods, but these are not necessarily
ation of HTTP get methods means that
nation on the server, but there is no
that unless the services are properly
ving these guidelines then an HTTP
side.

site/updateOrder, which calls out to a
The URL is not idempotent so the
ditionally, the attacker may be able to
forms updates (instead of merely
retrieving data). This may result in malicious or inadvertent altering of data on the server.

Attack Prerequisites

The attacker needs to be able to identify HTTP Get URLs. The Get methods must be set to call applications that perform operations other than get such as update and delete.

Typical Likelihood of Exploit

Likelihood: High

Methods of Attack

- Injection

..the attacker may be able to exploit the URL published as a Get method that actually performs updates (instead of merely retrieving data). This may result in malicious or inadvertent altering of data on the server.

More Seriously

Design pattern related vulnerabilities

- Restful Privilege Elevation
- Utilizing REST's Trust in the System Resource to Register Man in the Middle
- Session ID in the URL ☹

Related to commonly use implementation method

- JSON hijacking array vulnerability

Somewhat linked to REST

- XSRF

Any Other Web Application Vulnerability

- It is just a web application after all

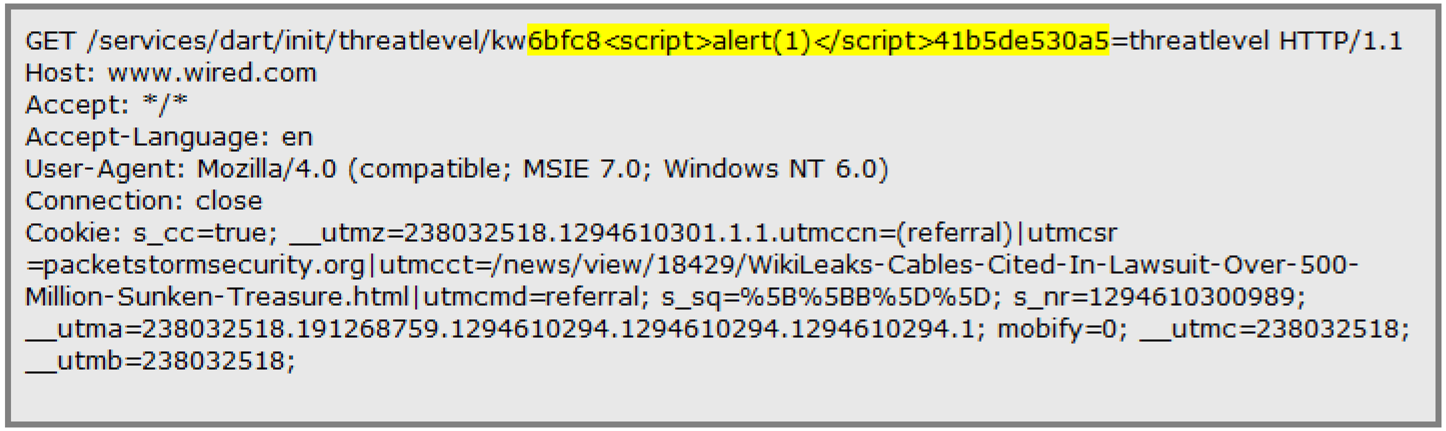
Nothing to Call Home About

Testing Challenges

Parameters Embedded in URLs

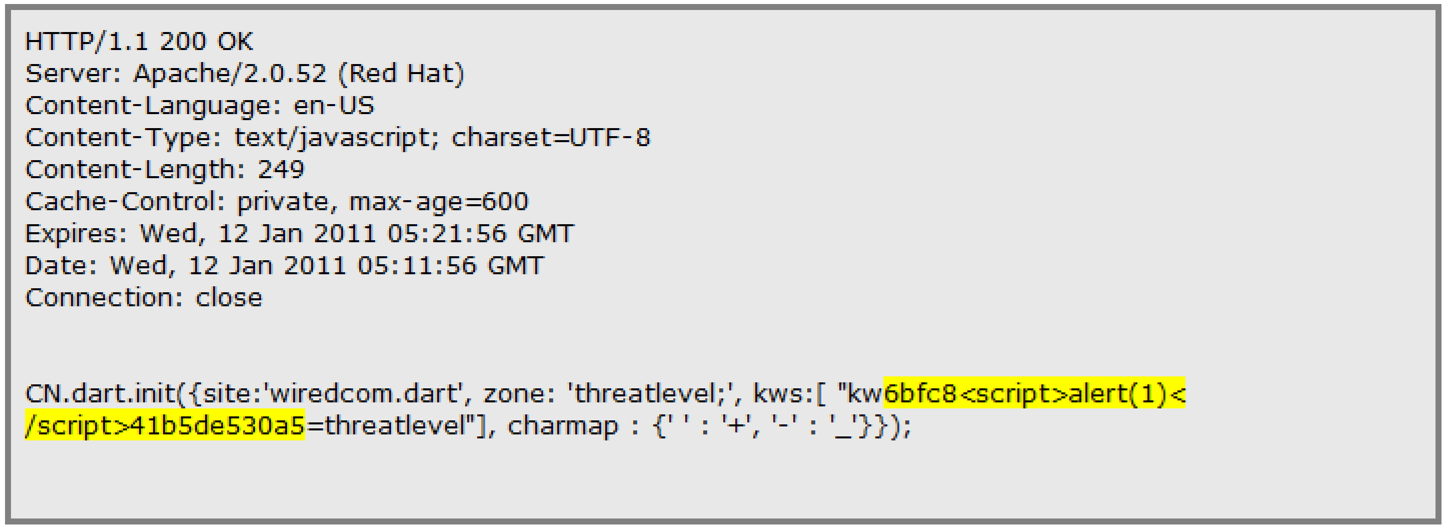
Susceptible to Injection and Manipulation

Request



```
GET /services/dart/init/threatlevel/kw6bfc8<script>alert(1)</script>41b5de530a5=threatlevel HTTP/1.1
Host: www.wired.com
Accept: */*
Accept-Language: en
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)
Connection: close
Cookie: s_cc=true; __utmz=238032518.1294610301.1.1.utmccn=(referral)|utmcsr
=packetstormsecurity.org|utmcct=/news/view/18429/WikiLeaks-Cables-Cited-In-Lawsuit-Over-500-
Million-Sunken-Treasure.html|utmcmd=referral; s_sq=%5B%5BB%5D%5D; s_nr=1294610300989;
__utma=238032518.191268759.1294610294.1294610294.1294610294.1; mobify=0; __utmc=238032518;
__utmb=238032518;
```

Response

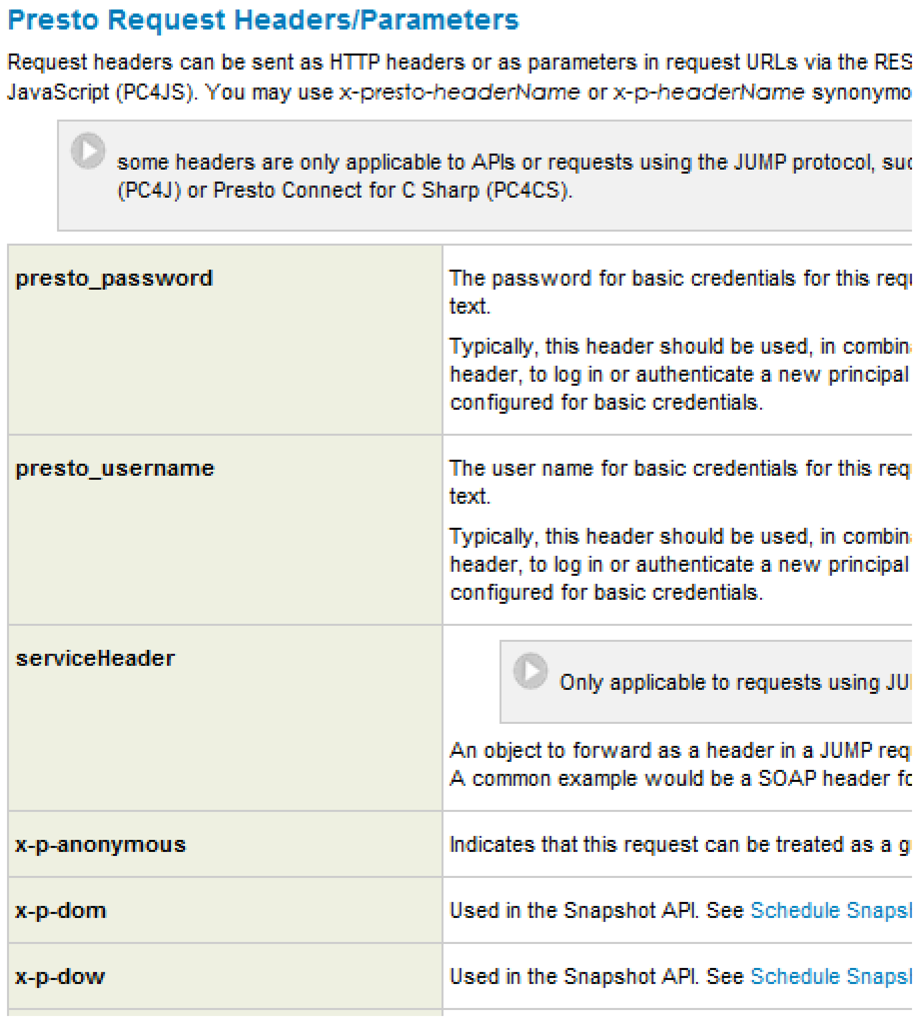


```
HTTP/1.1 200 OK
Server: Apache/2.0.52 (Red Hat)
Content-Language: en-US
Content-Type: text/javascript; charset=UTF-8
Content-Length: 249
Cache-Control: private, max-age=600
Expires: Wed, 12 Jan 2011 05:21:56 GMT
Date: Wed, 12 Jan 2011 05:11:56 GMT
Connection: close


CN.dart.init({site:'wiredcom.dart', zone: 'threatlevel;', kws:[ "kw6bfc8<script>alert(1)<
/script>41b5de530a5=threatlevel"], charmap : {' ' : '+', '-' : '_'}});
```

And Other Strange Locations

- Parameters in request headers
- Matrix parameters
- JSON/XML as a structured value to other parameters



Presto Request Headers/Parameters

Request headers can be sent as HTTP headers or as parameters in request URLs via the RES
JavaScript (PC4JS). You may use *x-presto-headerName* or *x-p-headerName* synonymo

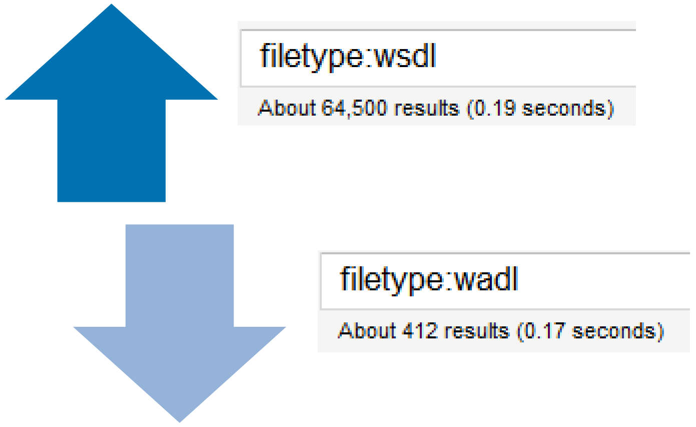
some headers are only applicable to APIs or requests using the JUMP protocol, suc
(PC4J) or Presto Connect for C Sharp (PC4CS).

presto_password	The password for basic credentials for this requ text. Typically, this header should be used, in combin header, to log in or authenticate a new principal configured for basic credentials.
presto_username	The user name for basic credentials for this req text. Typically, this header should be used, in combin header, to log in or authenticate a new principal configured for basic credentials.
serviceHeader	Only applicable to requests using JU An object to forward as a header in a JUMP req A common example would be a SOAP header fo
x-p-anonymous	Indicates that this request can be treated as a g
x-p-dom	Used in the Snapshot API. See Schedule Snaps
x-p-dow	Used in the Snapshot API. See Schedule Snaps

The Attack Surface Issue

REST APIs are Challenging to Map

- Larger than actually used in application:
  - URIs, Methods, Parameters
- Poorly documented:
  - WADL is only a proposed standard and hardly ever used.
- Many different ways to express parameters.
- Especially difficult for automated pen-testing.


filetype:wsdl

About 64,500 results (0.19 seconds)

filetype:wadl

About 412 results (0.17 seconds)

Solutions

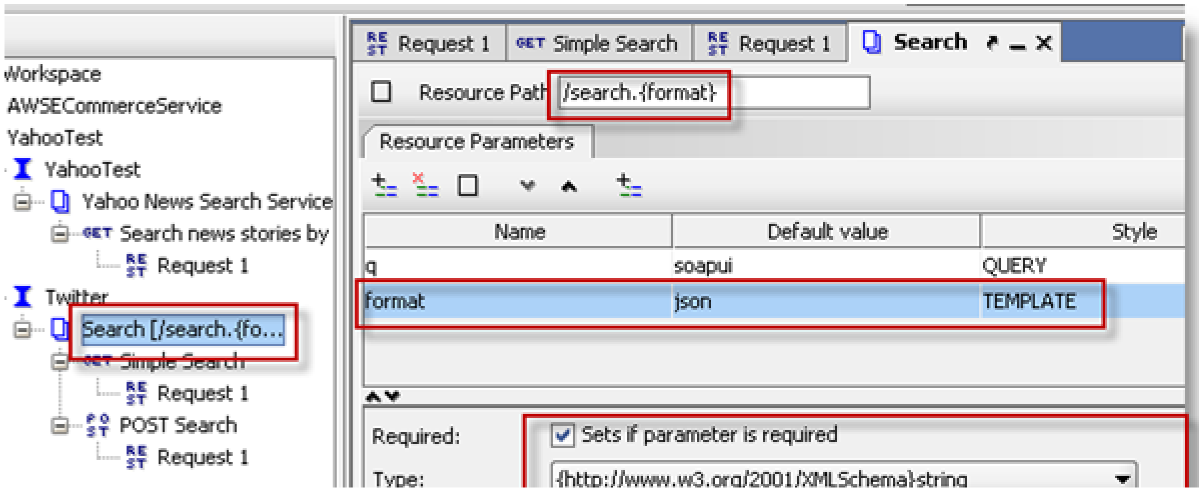
- Manual Definition of the Attack Surface
- Analyze Documentation & Configuration
- Automated Discovery of Rules

Manual Definition of the Attack Surface

Two use cases:

- Define the entire API – complete but difficult. Possible, especially as part of a rigorous QA *(SoapUI example on right).*

- Define templates for identifying and handling REST during crawl.
  – Critical for JS frameworks.



Workspace
AWSECommerceService
YahooTest
YahooTest
Yahoo News Search Service
GET Search news stories by
Request 1
Twitter
Search [/search.{fo...
GET Simple Search
Request 1
POST Search
Request 1

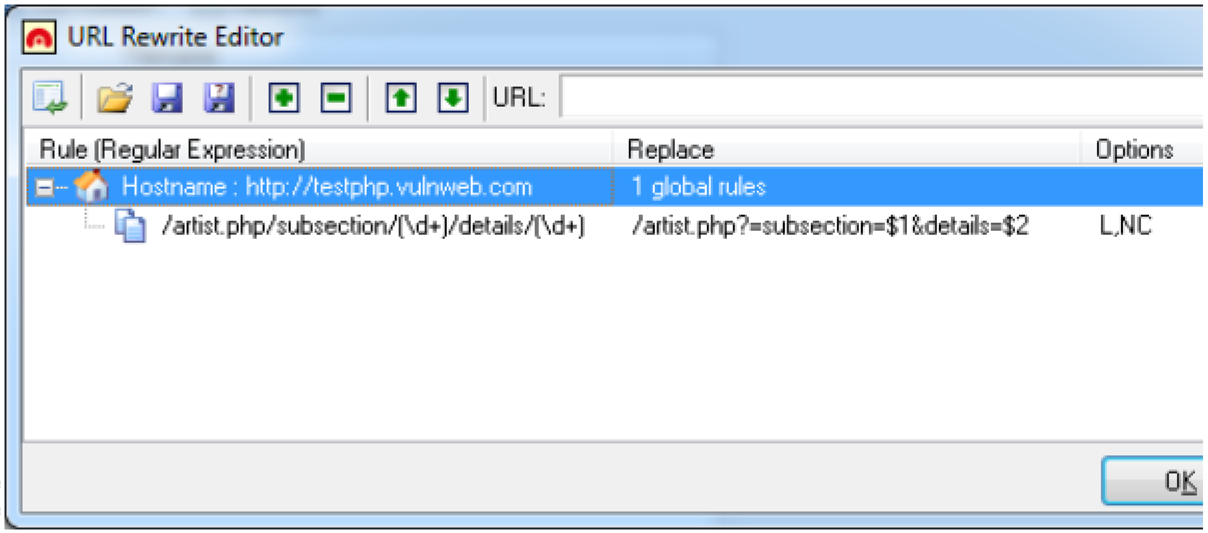
Request 1 | GET Simple Search | Request 1 | Search

Resource Path /search.{format}

Resource Parameters

Name	Default value	Style
q	soapui	QUERY
format	json	TEMPLATE

Required: ✓ Sets if parameter is required
Type: {http://www.w3.org/2001/XMLSchema}string



URL Rewrite Editor

URL:

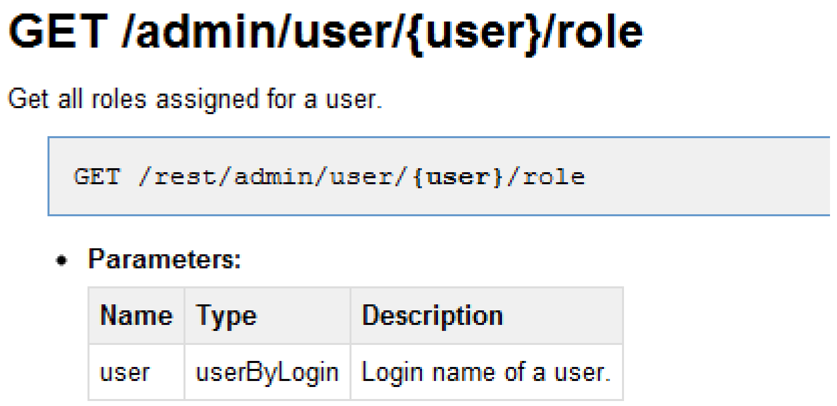
Rule (Regular Expression)	Replace	Options
Hostname : http://testphp.vulnweb.com	1 global rules	
/artist.php/subsection/(\d+)/details/(\d+)	/artist.php?=subsection=$1&details=$2	L,NC

OK

Analyze Documentation & Configuration

Informal documentation:

- Highly unstructured
- Requires heuristic, training and trial and error.



GET /admin/user/{user}/role

Get all roles assigned for a user.

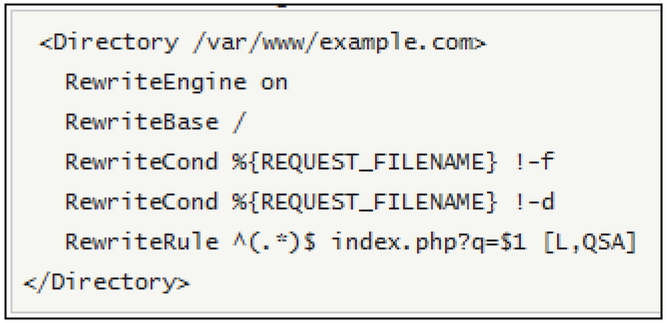
```
GET /rest/admin/user/{user}/role
```

- **Parameters:**

Name	Type	Description
user	userByLogin	Login name of a user.

Web Servers and applications configuration:

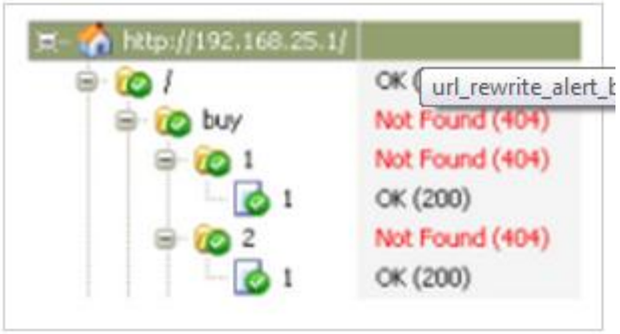
- Easier to use but a limited solution.



```
 <Directory /var/www/example.com>
   RewriteEngine on
   RewriteBase /
   RewriteCond %{REQUEST_FILENAME} !-f
   RewriteCond %{REQUEST_FILENAME} !-d
   RewriteRule ^(.*)$ index.php?q=$1 [L,QSA]
</Directory>
```

Automated Discovery of Rules

- Irregular 404 codes
  - Including site specific ones.
- Pattern analysis:
  - Matrix parameters
  - JSON or XML as values to parameters
- Irregular headers
- And….
  - Need to wait till year end…..



```
http://192.168.25.1/
  /                 OK (  url_rewrite_alert_
    buy             Not Found (404)
      1             Not Found (404)
        1           OK (200)
      2             Not Found (404)
        1           OK (200)
```

Thank You!