Server Side Request Forgery

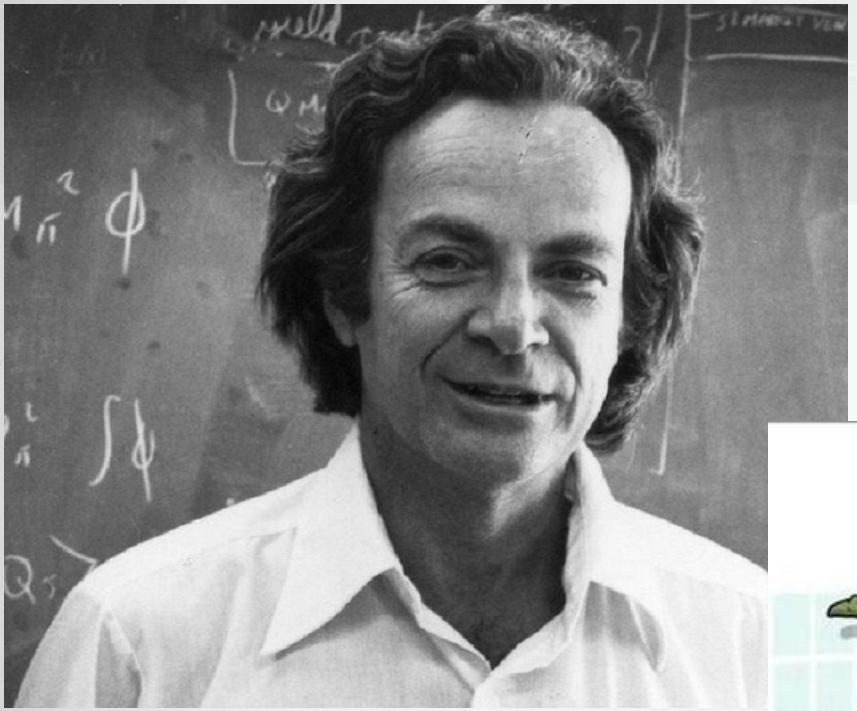
David Calligaris
Director Vulnerability Research
Huawei Technologies GMBH

OWASP Italy Day
Udine, 14th December 2019

Disclaimer

All the content of these slides represent <u>my personal view</u> not that of my employer.

Understanding



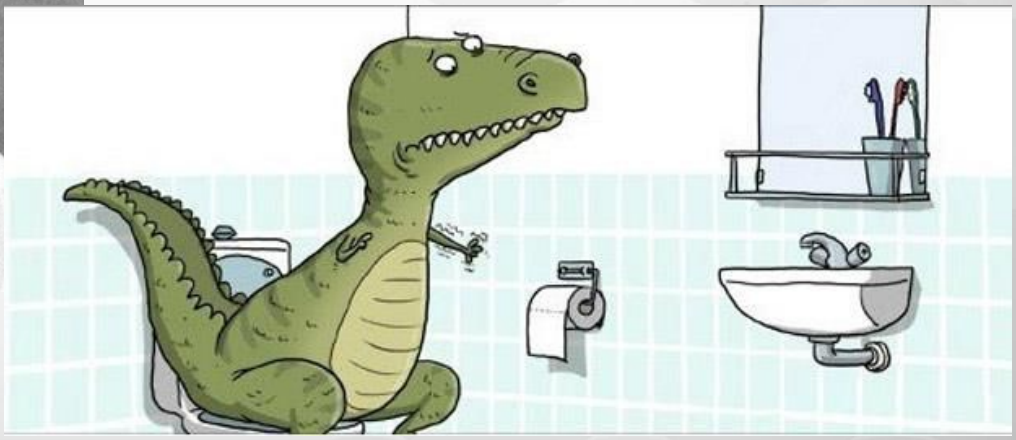


Table of Contents

What is a Vulnerability?

- How we can get food / drinks for free?

Vulnerabilities

Keyword (*)	Year	Number (Today)
XSS	2000 / 2001	14733
SQL Injection	2000 / 2001	8119
CSRF	2002	2534
XXE	2002	549
SSRF	2013	243
IDOR	2018	17

https://cve.mitre.org/cgi-bin/cvekey.cgi

(*) Some names are missing e.g. LFI, RFI, Command Injection etc

Vulnerabilities

Keyword (*)	Year	Number (Today)
XSS	2000 / 2001	14733
SQL Injection	2000 / 2001	8119
CSRF	2002	2534
XXE	2002	549
SSRF	2013	243
IDOR	2018	17

https://cve.mitre.org/cgi-bin/cvekey.cgi

(*) Some names are missing e.g. LFI, RFI, Command Injection etc

SSRF

S erver
S side
R equest
F orgery

In a Server-Side Request Forgery (SSRF) attack, the attacker can abuse functionality on the server to read or update internal resources. The attacker can supply or a modify a URL which the code running on the server will read or submit data to, and by carefully selecting the URLs, the attacker may be able to read server configuration such as AWS metadata, connect to internal services like http enabled databases or perform post requests towards internal services which are not intended to be exposed.

https://www.owasp.org/index.php/Server_Side_Request_Forgery

Why talk about SSRF?

- Microservices Impact
- Cloud Impact
- Quite popular in Bug Bounty Programs
- New research area

> *"SSRF has become the most serious vulnerability facing organizations that use public clouds," Johnson wrote.*

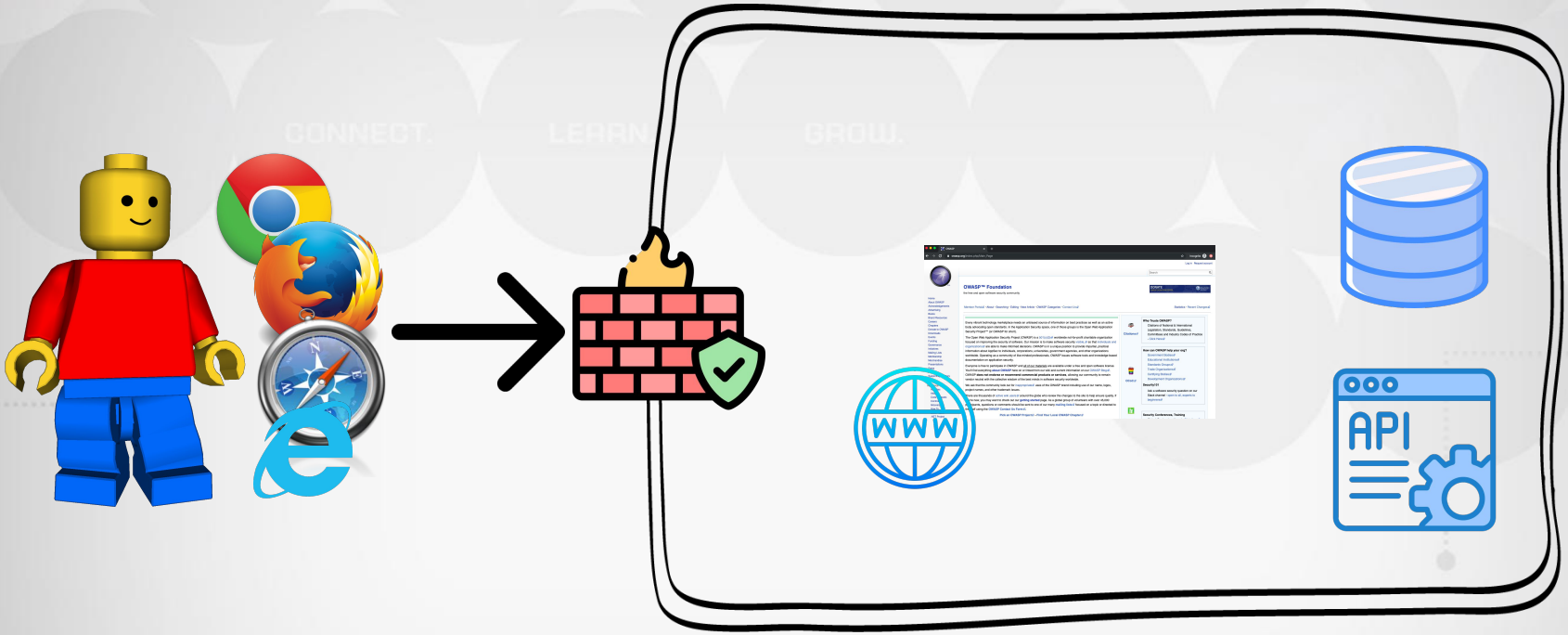
446 Privilege Escalation From user to SYSTEM via unauthenticated command execution
By b0yd to Ubiquiti Inc. | Resolved | Critical | $16,109.00

15 SSRF vulnerability on [redacted] leaks internal IP and various sensitive ir
By alyssa_herrera to U.S. Dept Of Defense | Resolved | Medium

48 Inappropriate URL parsing may cause security risk!
By orange to PHP (IBB) | Resolved | Medium | $1,000.00 | disclosed about 1 month ago

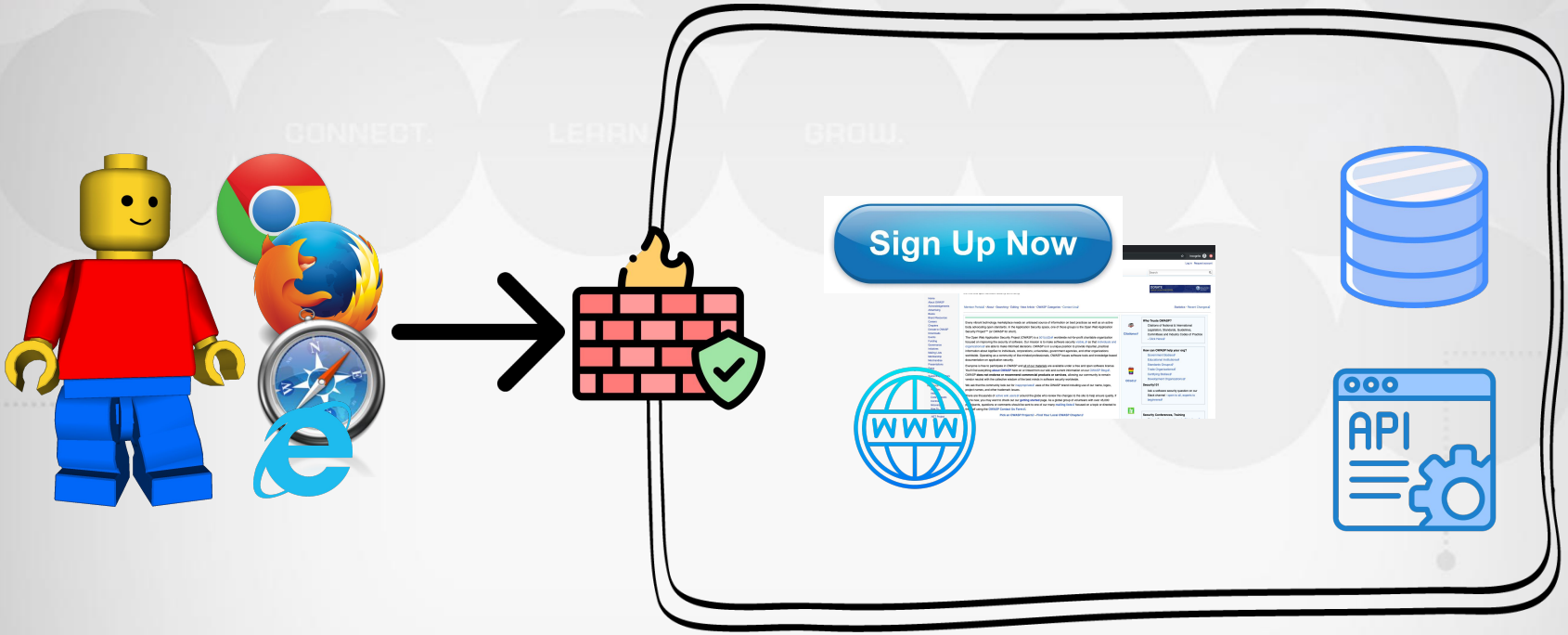
Server-Side Request Forgery (SSRF) | disclosed 11 days ago

SSRF - Example



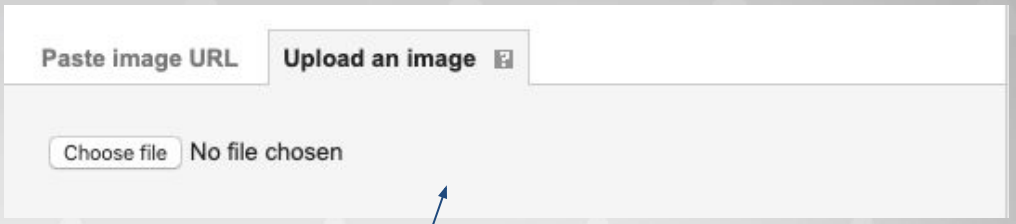
Once upon a time a user surf the web …

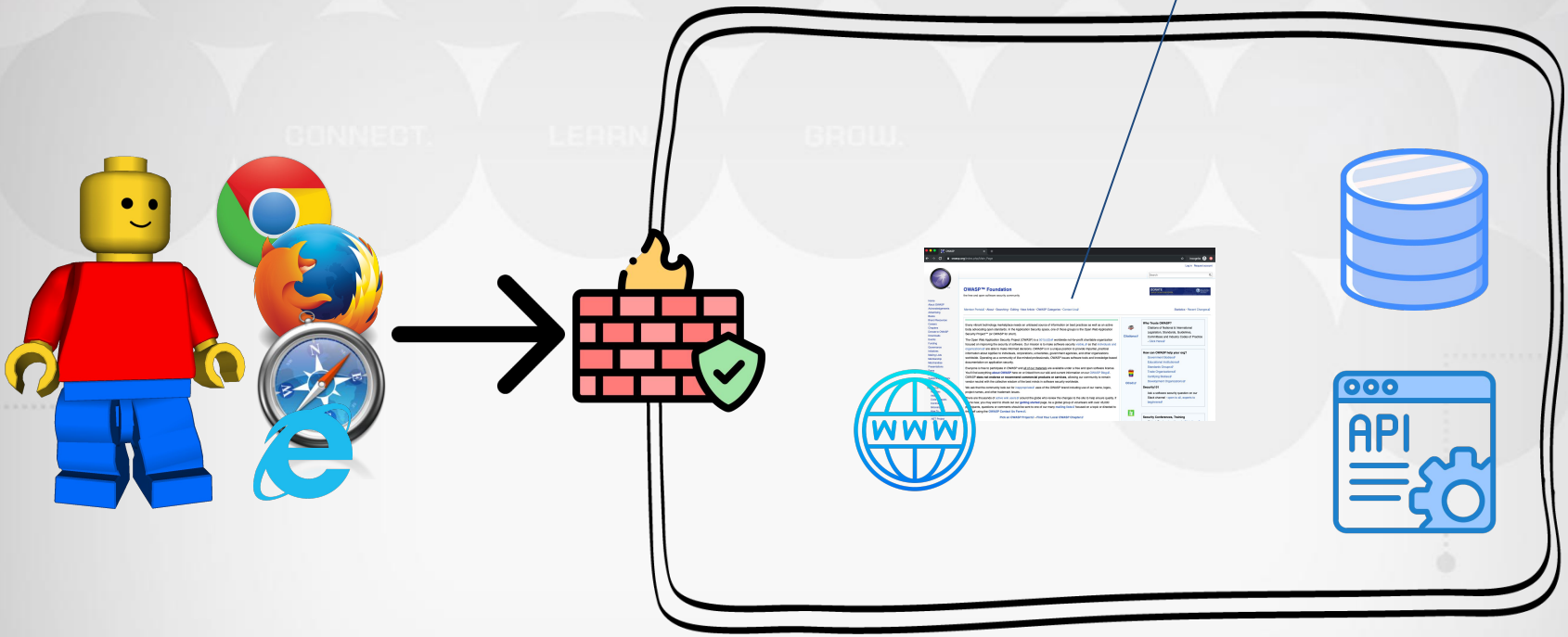
SSRF - Example



On the website the user create a profile ...

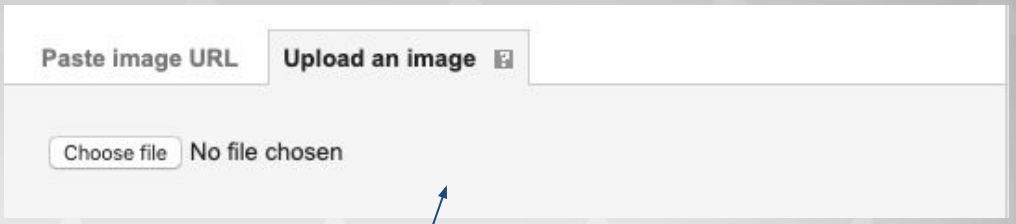
SSRF - Example

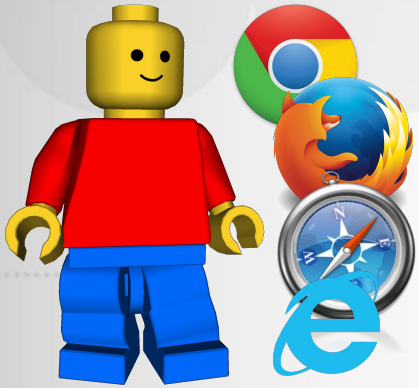


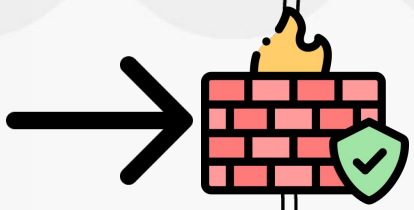


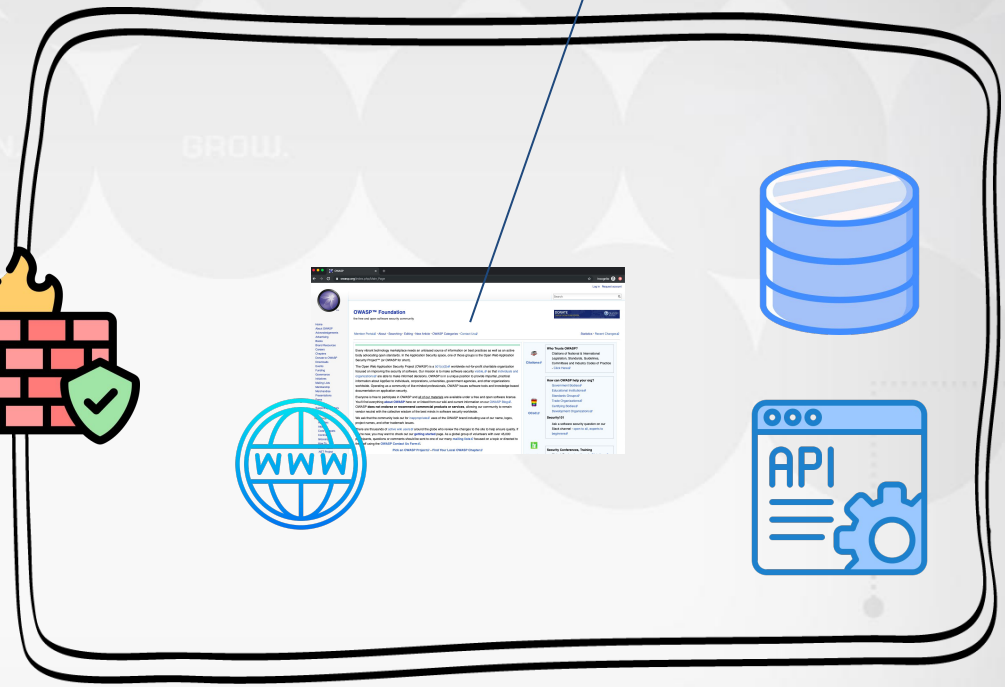
on the registration process the user has the opportunity to upload a profile picture from his computer

SSRF - Example

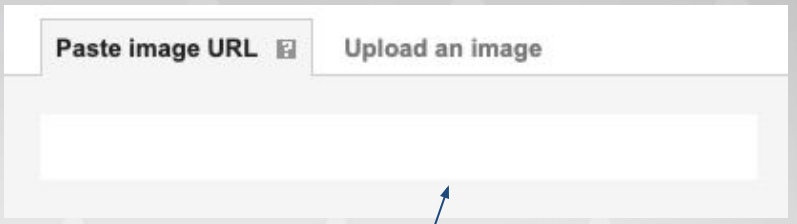


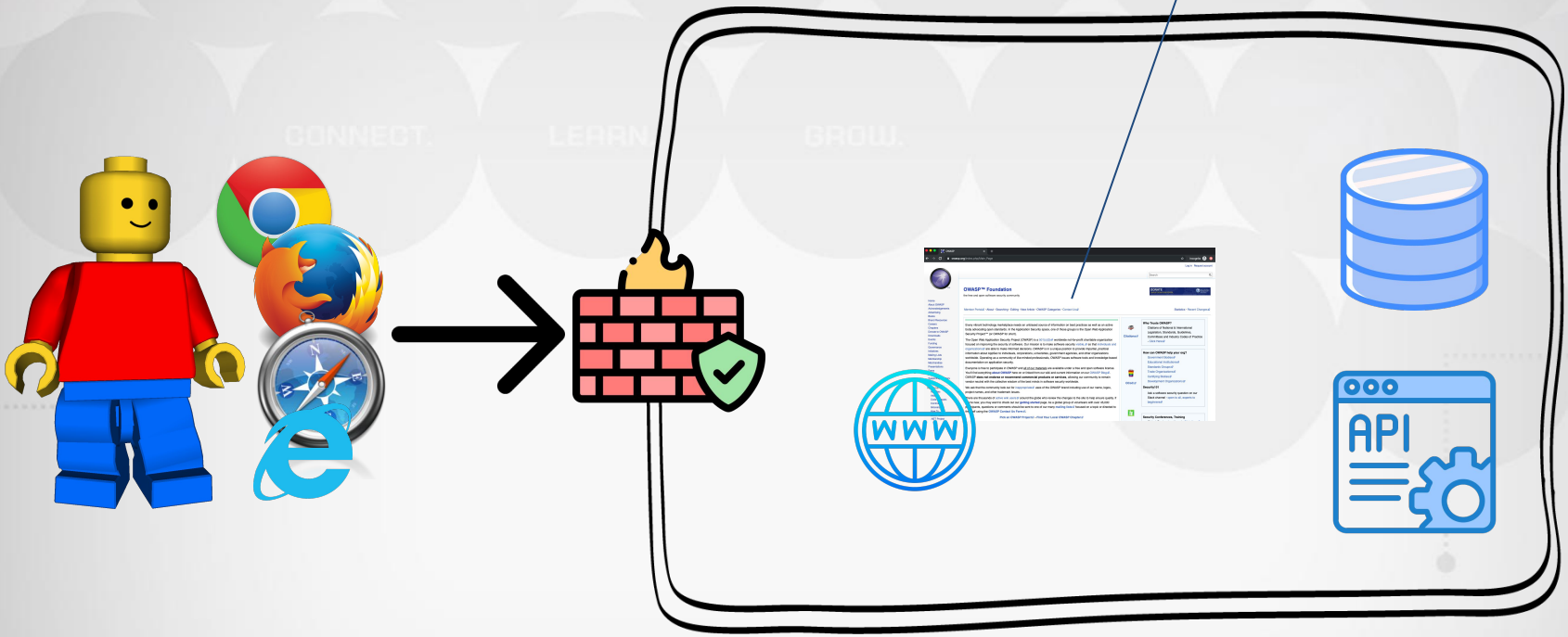






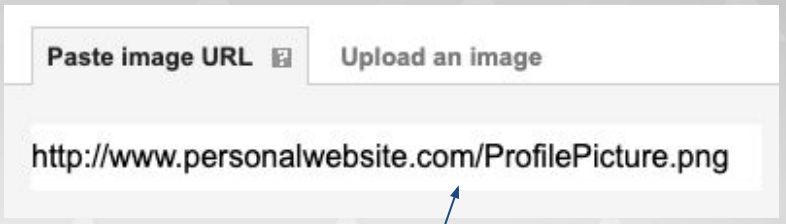
SSRF - Example

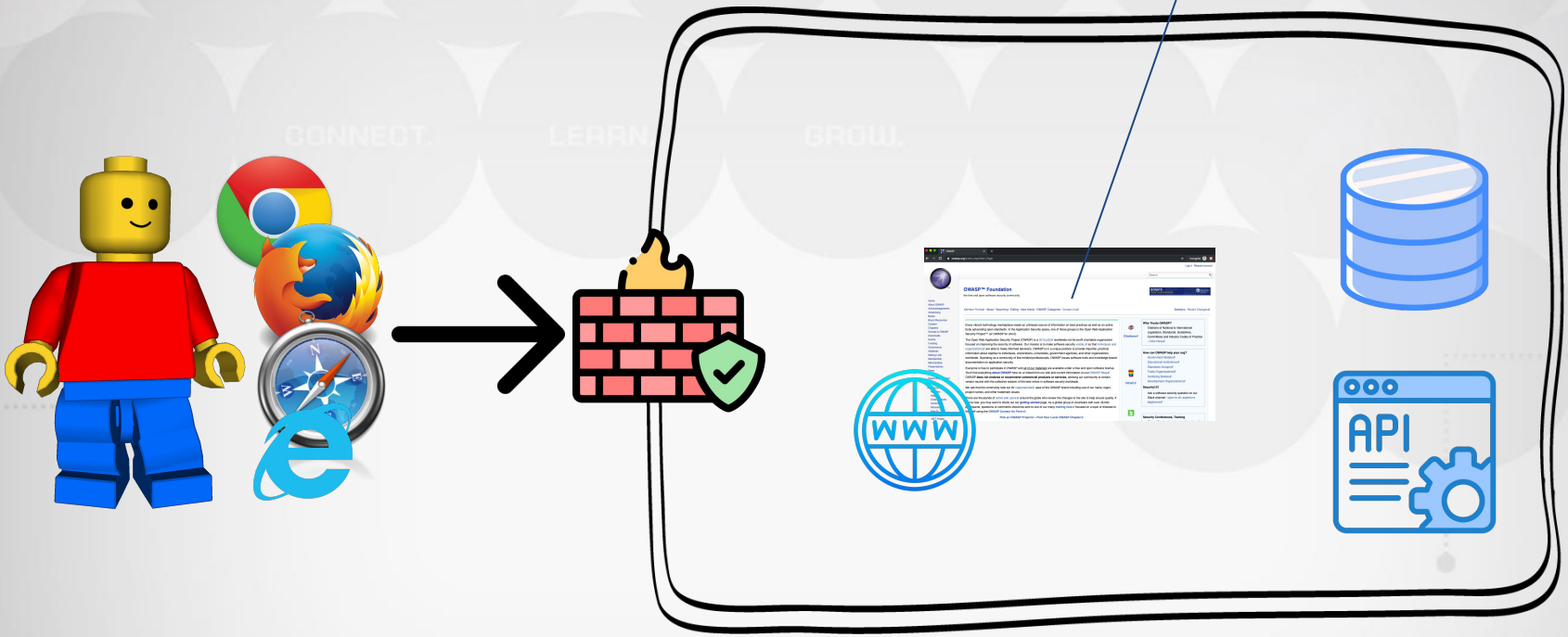




… or from a url …

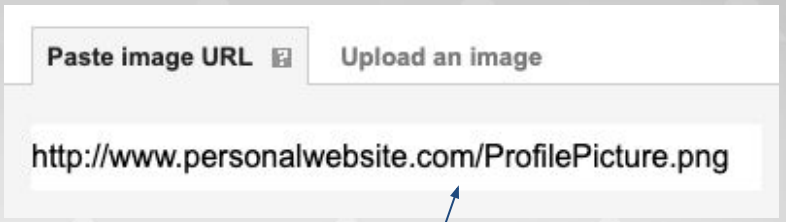
SSRF - Example

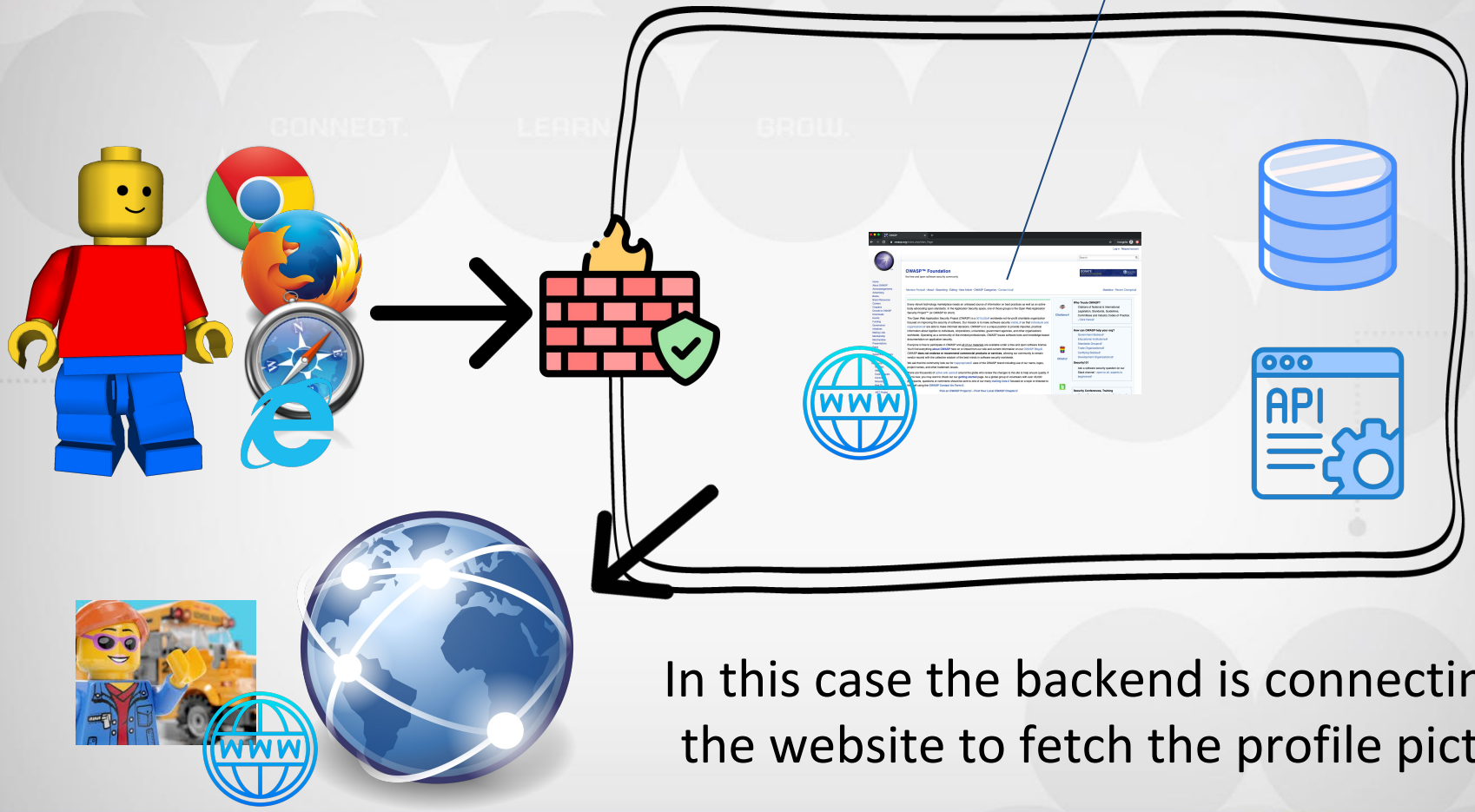




… like this example

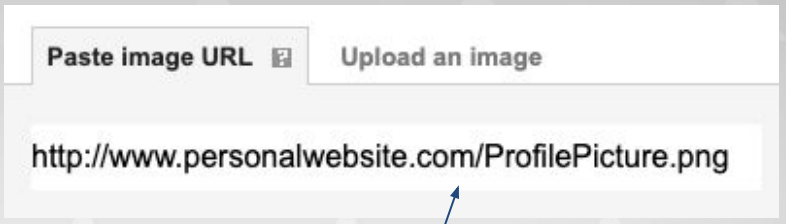
SSRF - Example

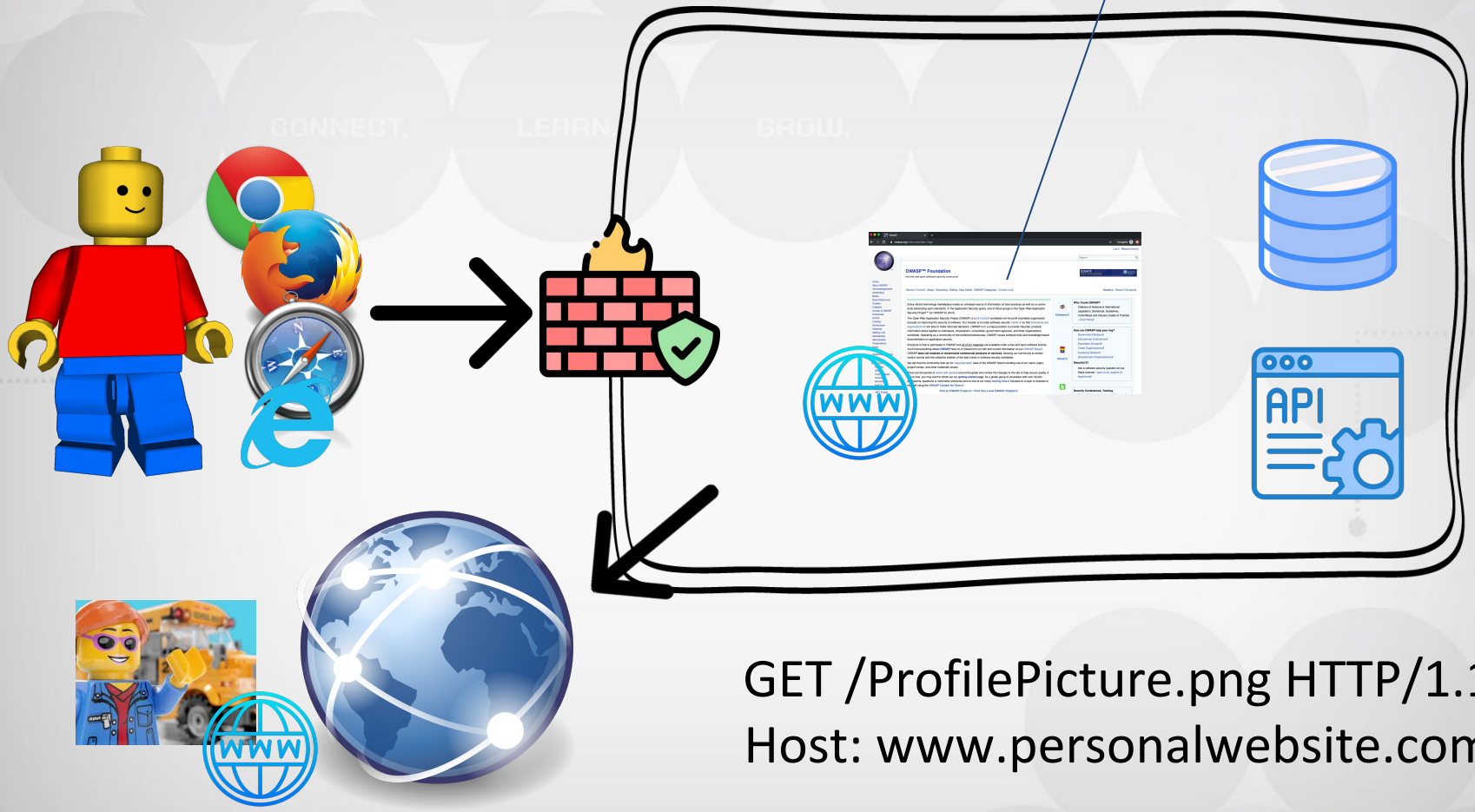




In this case the backend is connecting to the website to fetch the profile picture

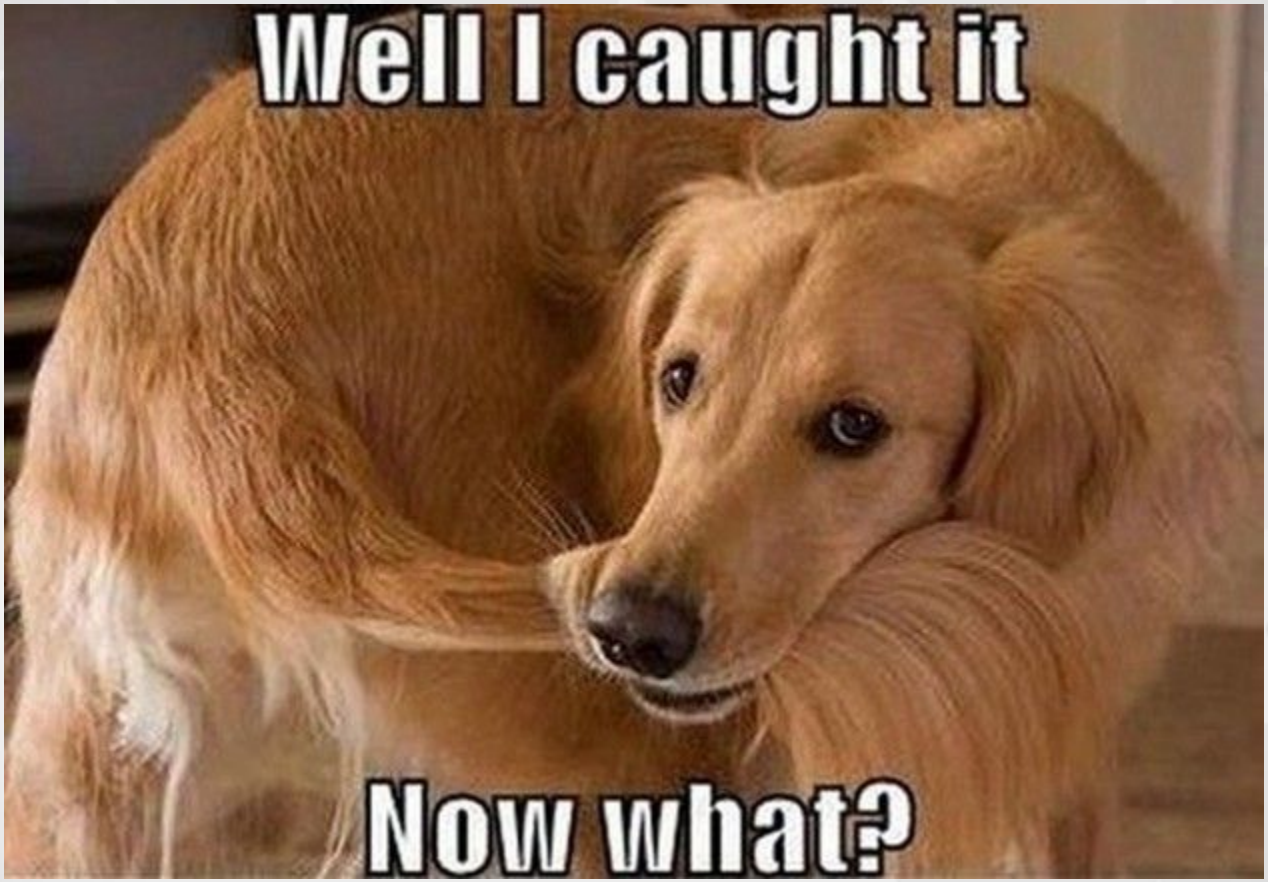
SSRF - Example



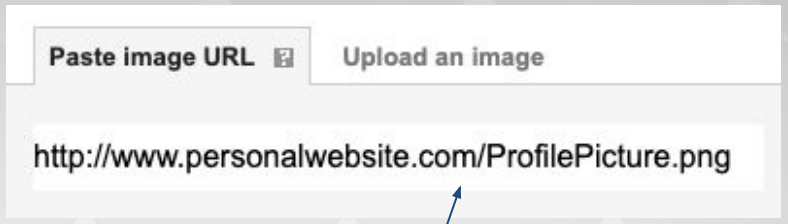


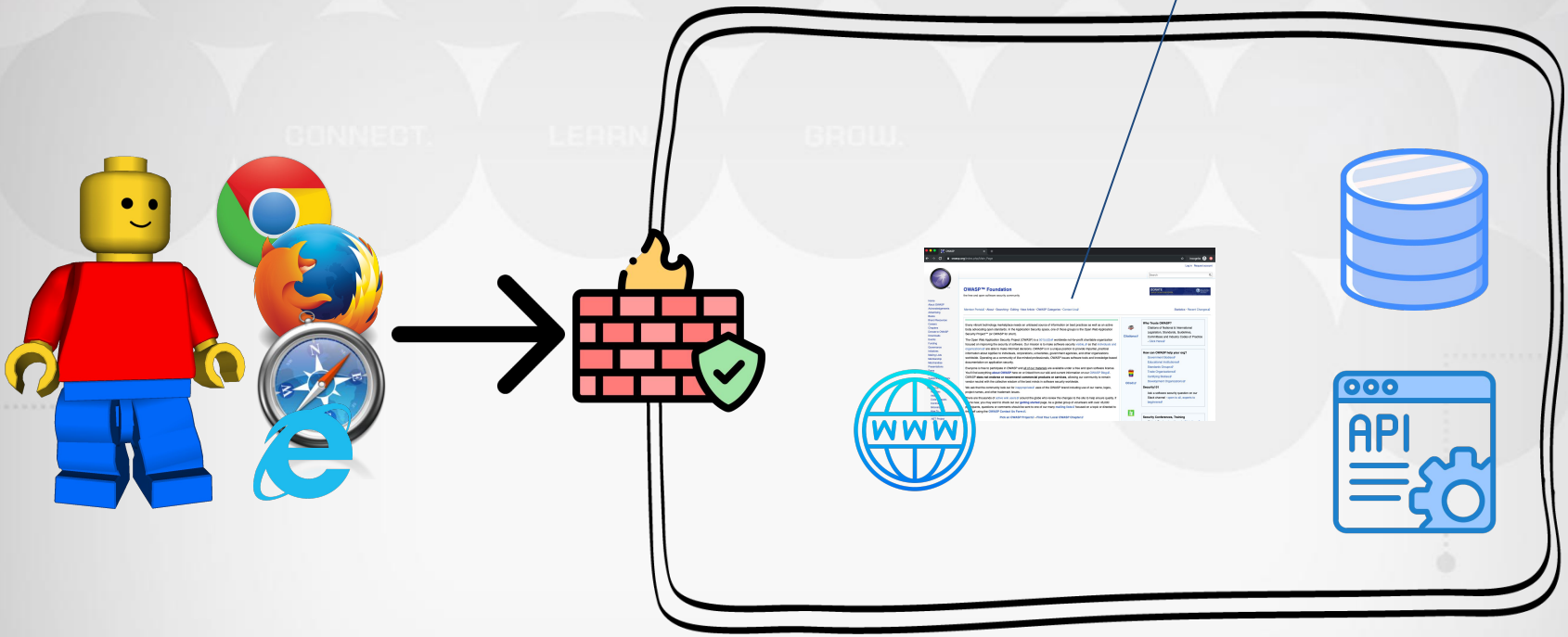
GET /ProfilePicture.png HTTP/1.1
Host: www.personalwebsite.com

SSRF - Example



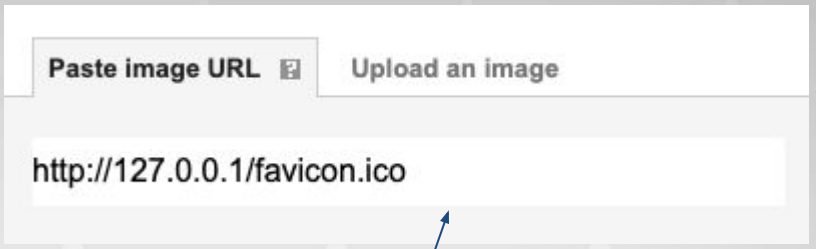
SSRF - Example

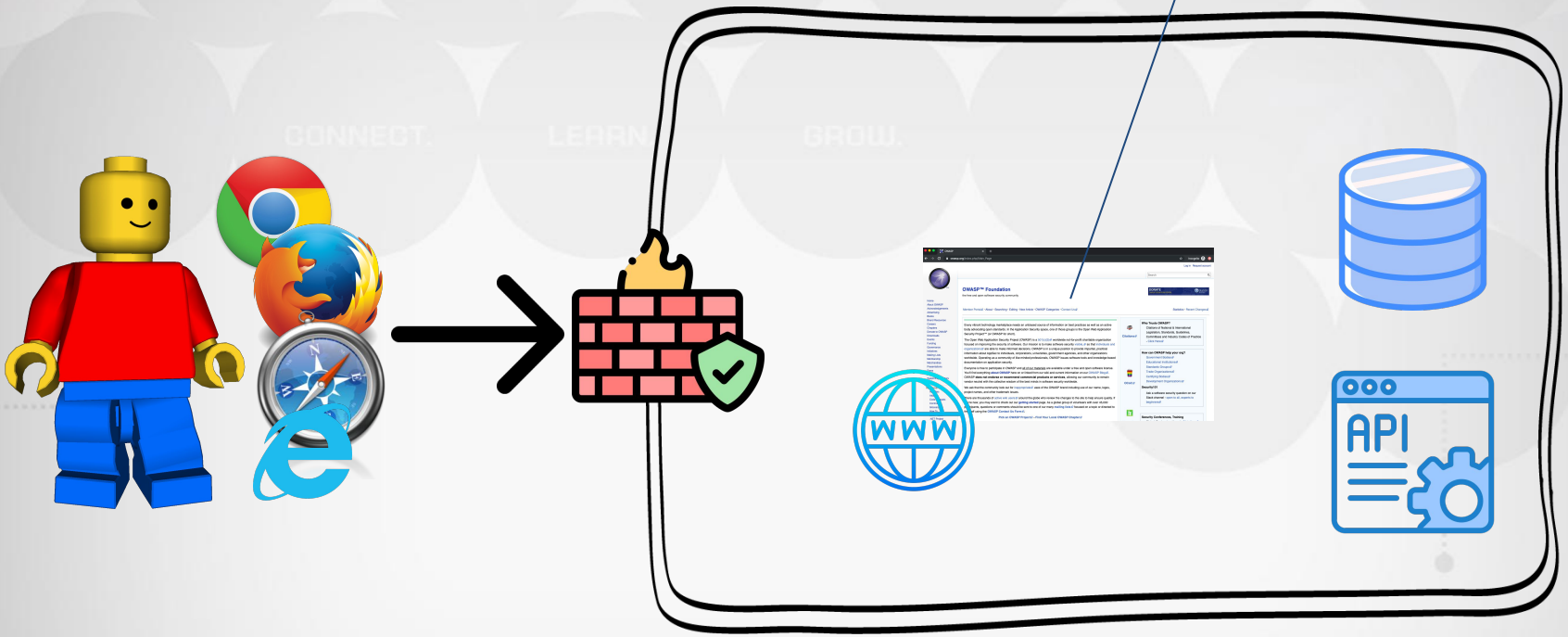




Let’s do a step back … do you remember when we submit this url ?

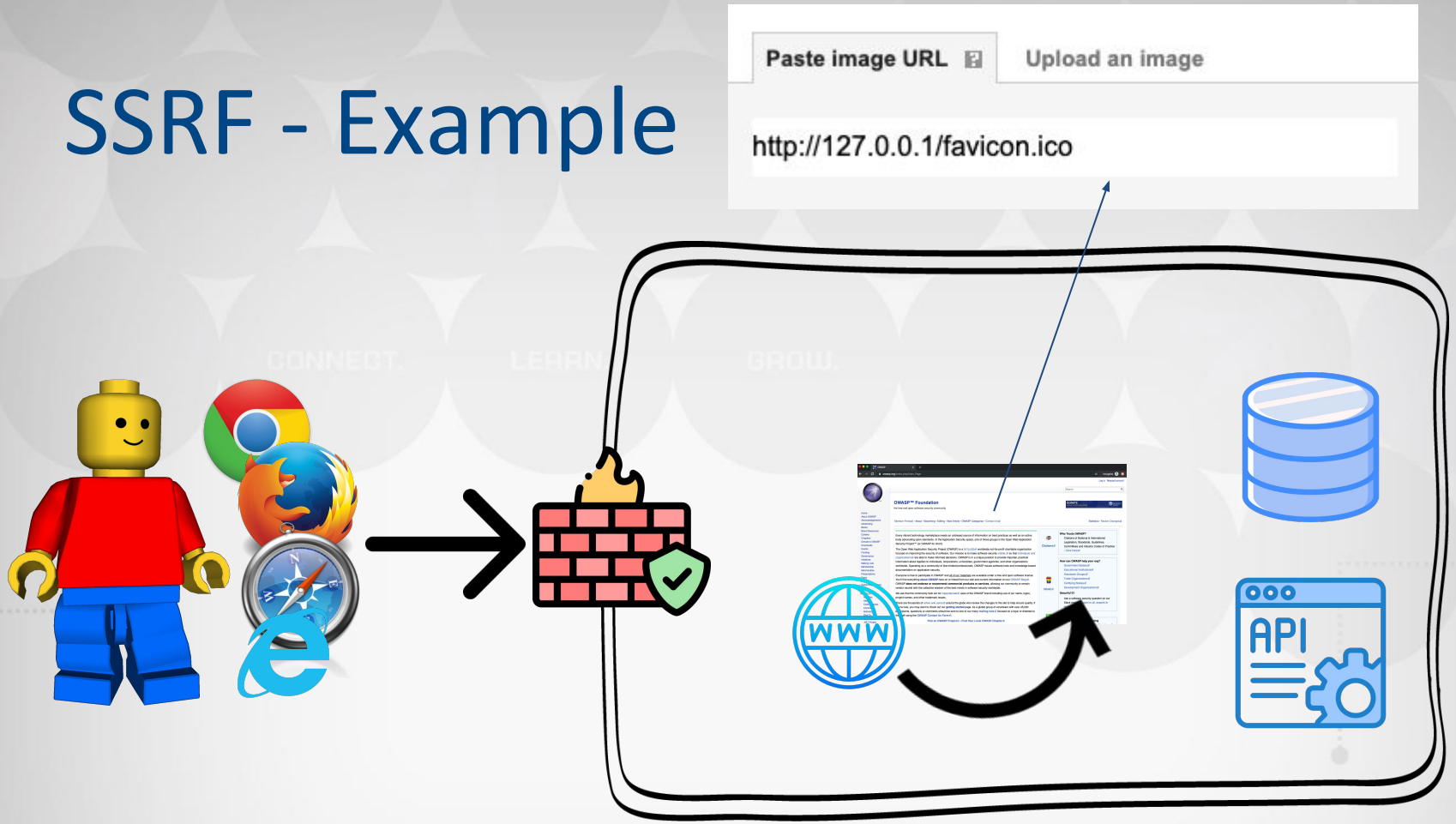
SSRF - Example





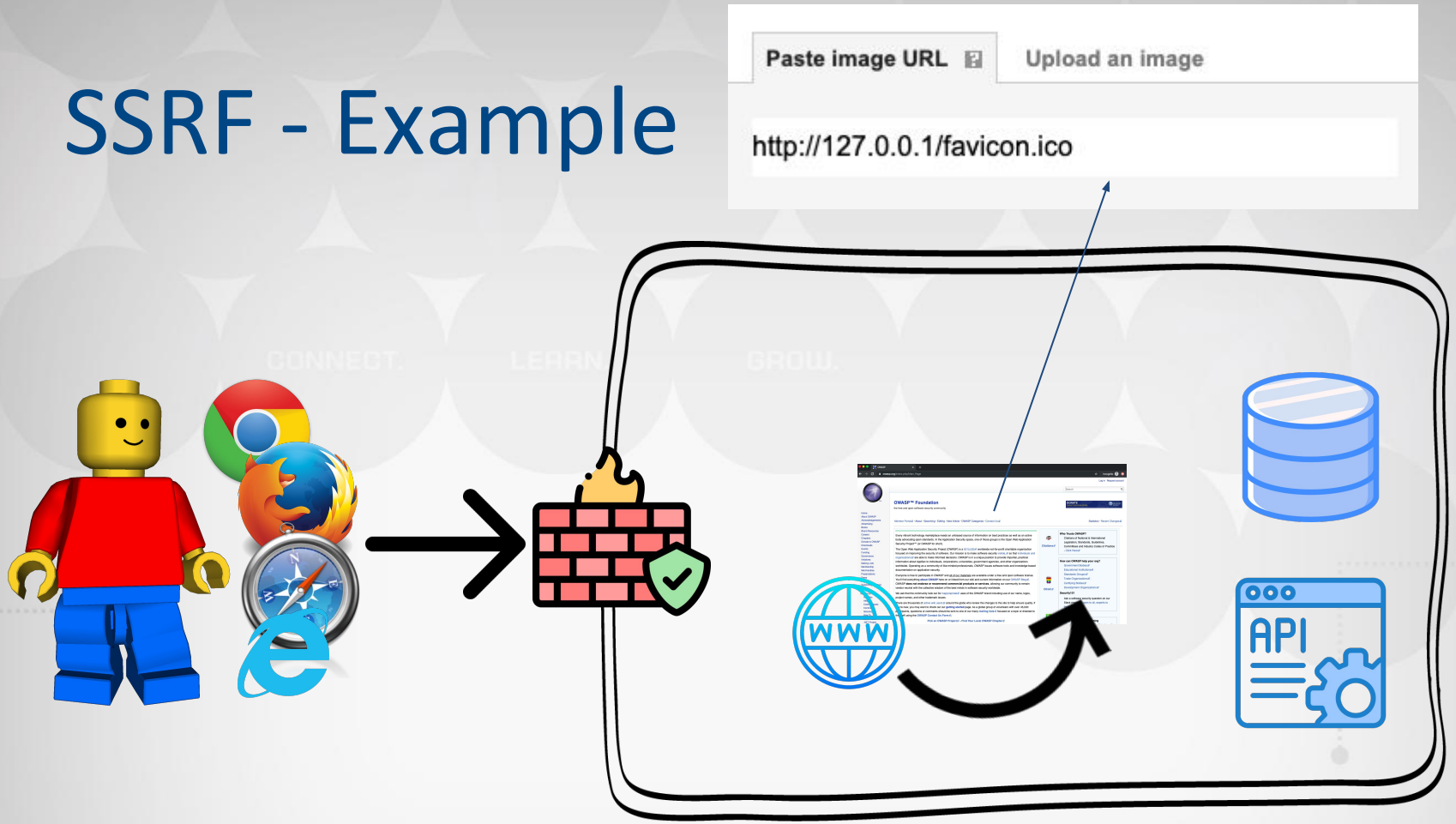
What will happen if we submit this one?

SSRF - Example



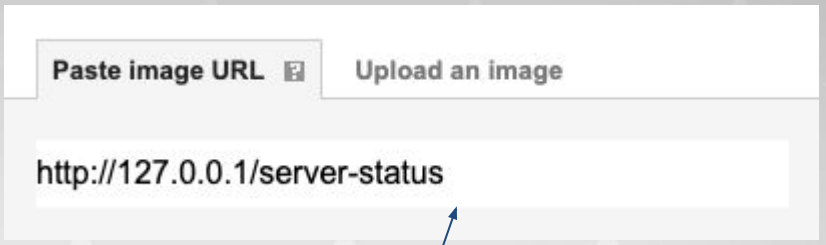
in this case the server connects to itself

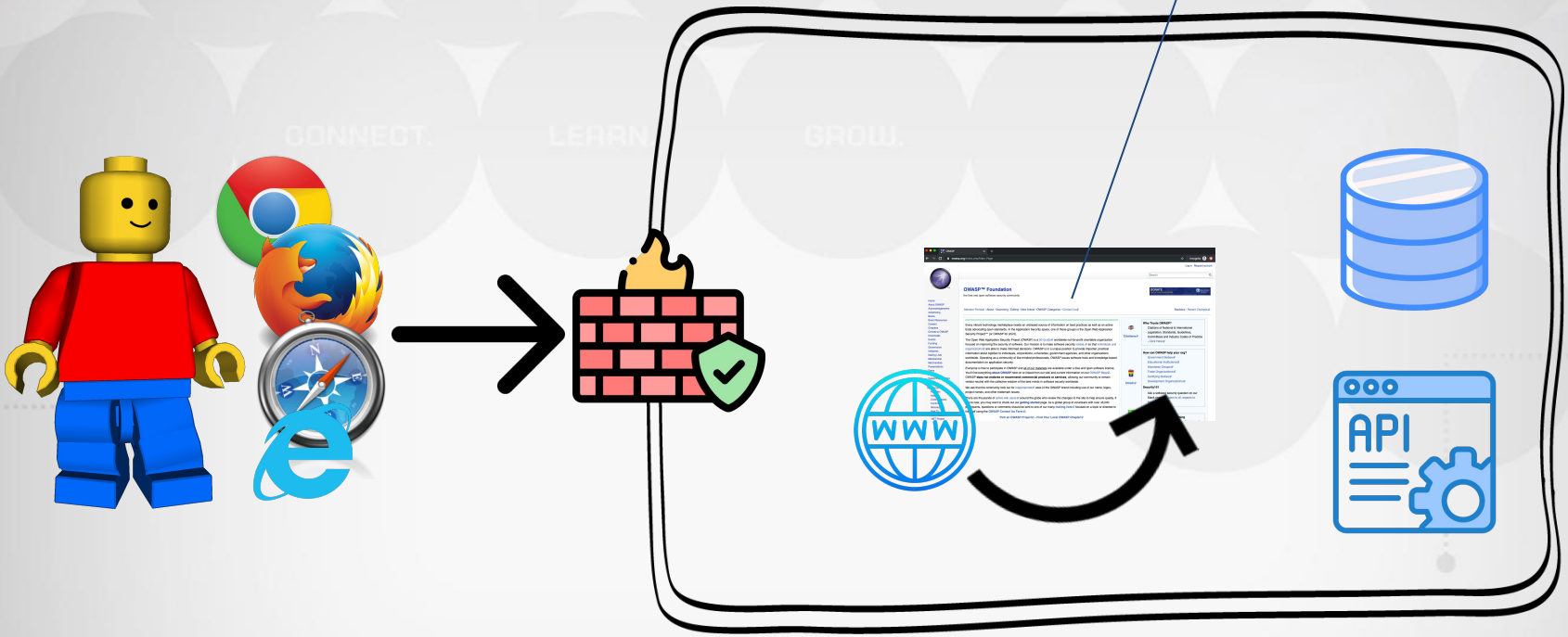
SSRF - Example



GET /favicon.ico HTTP/1.1
Host: 127.0.0.1

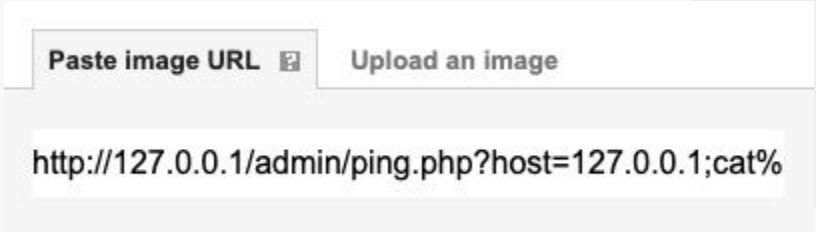
SSRF - Example





GET /server-status/ HTTP/1.1
Host: 127.0.0.1

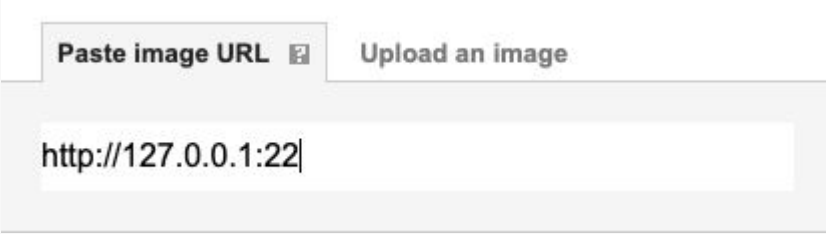
SSRF - Example



http://127.0.0.1/admin/ping.php?host=127.0.0.1;cat%20/etc/passwd

We can try to connect to exploit some vulnerabilities on the web server listening on localhost

SSRF - Example



We can try to connect to different TCP ports ...

SSRF - Example

… and we can get different messages

```
Error: cannot upload image: SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.4
```

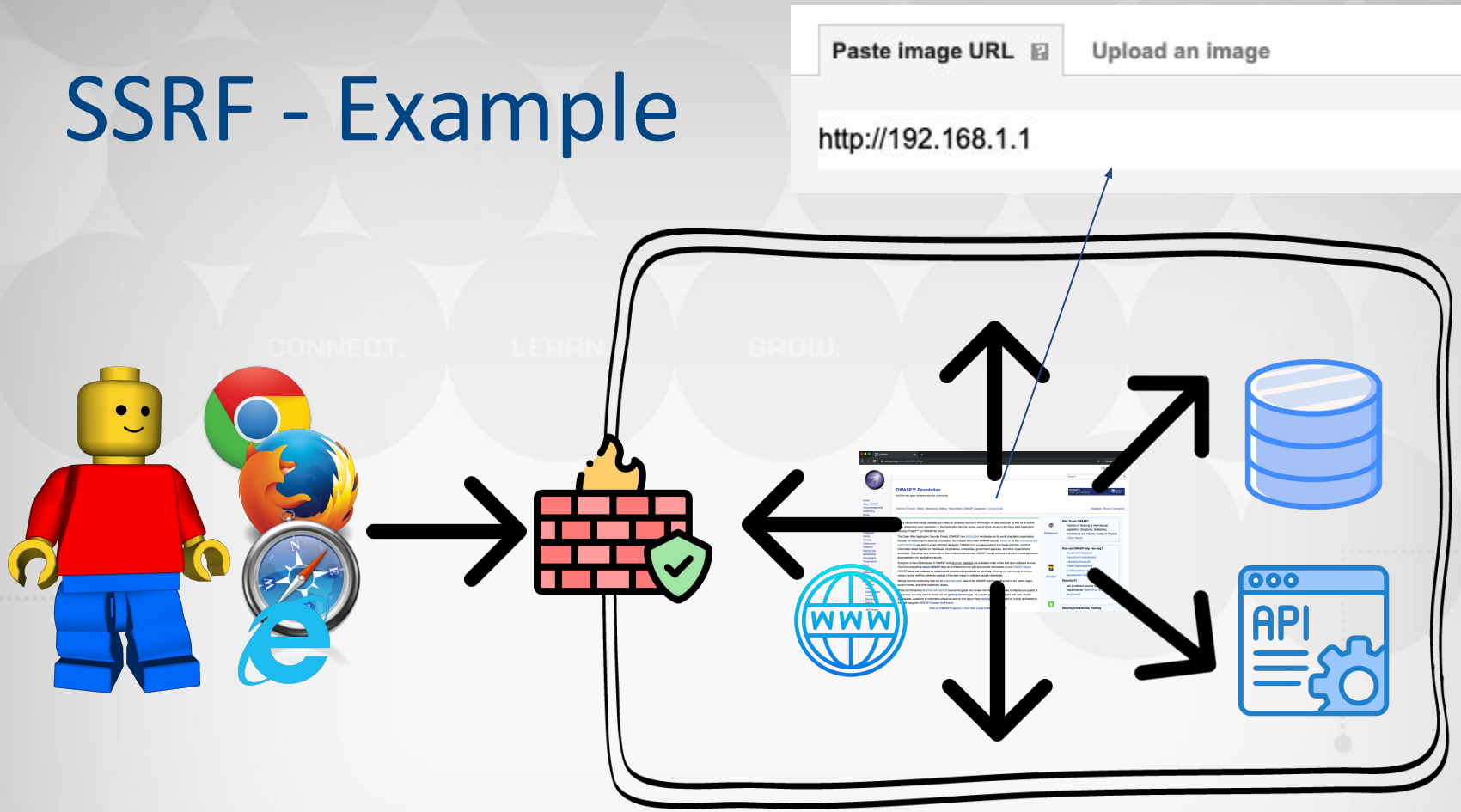
Closed Port

```
Error: cannot upload image: Connection Failed
```

Open Port

This allow us to perform a **Port Scan** on the localhost (127.0.0.1)

SSRF - Example



We can use a similar technique to perform a Port Scan in the local network (LAN)

SSRF - Example

Host Alive & Open Port

```
Error: cannot upload image: http-server-header: Apache/2.2.8
(Ubuntu) DAV/2
```

Host Down or Closed Port

```
Error: cannot upload image: Connection Failed
```

SSRF - Example

RFC1918 name	IP address range	Number of addresses	Largest CIDR block (subnet mask)	Host ID size	Mask bits	*Classful* description[Note 1]
24-bit block	10.0.0.0 – 10.255.255.255	16 777 216	10.0.0.0/8 (255.0.0.0)	24 bits	8 bits	single class A network
20-bit block	172.16.0.0 – 172.31.255.255	1 048 576	172.16.0.0/12 (255.240.0.0)	20 bits	12 bits	16 contiguous class B networks
16-bit block	192.168.0.0 – 192.168.255.255	65 536	192.168.0.0/16 (255.255.0.0)	16 bits	16 bits	256 contiguous class C networks

https://en.wikipedia.org/wiki/Private_network

SSRF - Cloud

What about: http://169.254.169.254?

SSRF - Cloud

docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html

aws

English

Amazon Elastic Compute Cloud

User Guide for Linux Instances

Documentation - This Guide

Search

- What Is Amazon EC2?
- Setting Up
- Getting Started
- Best Practices
- Tutorials
- Amazon Machine Images
- Instances
  - Instance Types
  - Instance Purchasing Options

AWS Documentation » Amazon EC2 » User Guide for Linux Instances » Amazon EC2 Instances » Configuring Your Amazon Linux Instance » Instance Metadata and User Data

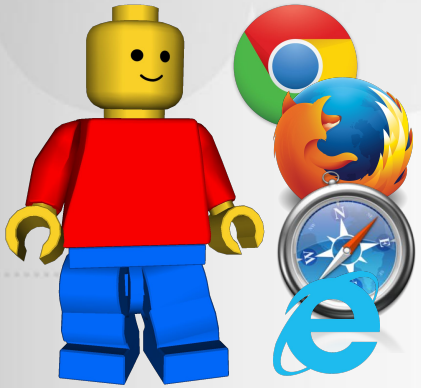
Instance Metadata and User Data

Instance metadata is data about your instance that you can use to configure or manage the running instance. Instance metadata is divided into categories, for example, host name, events, and security groups.

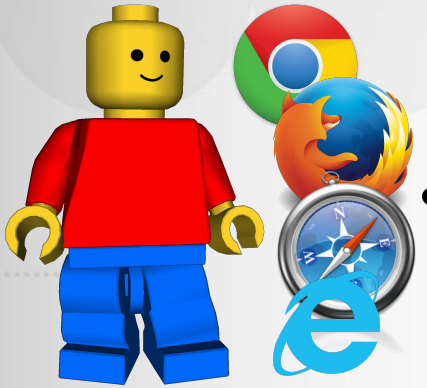
You can also use instance metadata to access *user data* that you specified when launching your instance. For example, you can specify parameters for configuring your instance, or include a simple script. You can build generic AMIs and use user data to modify the configuration files supplied at launch time. For example, if you run web servers for various small businesses, they can all use the same generic AMI and retrieve their content from the Amazon S3 bucket that you specify in the user data at launch. To add a new customer at any time, create a bucket for the customer, add their content, and launch your AMI with the unique bucket name provided to your code in the user data. If you launch more than one instance at the same time, the user data is available to all instances in that reservation. Each instance that is part of the same reservation has a unique `ami-launch-index` number, allowing you to write code that controls what to do. For example, the first host might elect itself as an initial master node in a cluster. For a detailed AMI launch example, see Example: AMI Launch Index Value.

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html

SSRF - Cloud



SSRF - Cloud



SSRF - Cloud

169.254.169.254/latest/meta-data/

```
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
identity-credentials/
instance-action
instance-id
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
```

SSRF - Cloud

- Interesting URLS:
  - http://169.254.169.254/latest/meta-data/iam/security-credentials
  - http://169.254.169.254/latest/dynamic/instance-identity/document
  - http://169.254.169.254/latest/meta-data/iam/security-credentials/ISRM-WAF-Role

openstack®

Google Cloud

amazon web services

SSRF - Cloud

Capital One hacked, over 100 million customers affected

Greg Kumparak @grg

blog.appsecco.com/an-ssrf-privileged-aws-keys-and-the-capital-one-breach-4c3c2cded3af

APPSECCO
THE APPLICATION SECURITY COMPANY

APPLICATION SECURITY | PENTESTING | CLOUD SECURITY | DEVOPS SECURITY | CULTURE | APPSECCO

An SSRF, privileged AWS keys and the Capital One breach

hnical side of how the Capital One breach
nd what you can do as a user of cloud
ing to you.

KrebsonSecurity
In-depth security news and investigation

ADVERTISING/SPEAKING | ABOUT THE AUTHOR

02 **What We Can Learn from the Capital One Hack**
AUG 19

Advertisement

SSRF - Mitigations

- We can have two approaches :
  - **Whitelist**: **Block Everythin**g and **Allow** some particular host / ip
  - **Blacklist**: **Allow Everything** and **Block** some host / ip

Due to application requirements (fetch external data) normally SSRF Mitigations use Black List approach

SSRF - Mitigations

- Block all the internal reserved IP:
  - 127.0.0.1
  - 192.168.0.0/16
  - 172.16.0.0/12
  - 10.0.0.0/8
  - 169.254.169.254

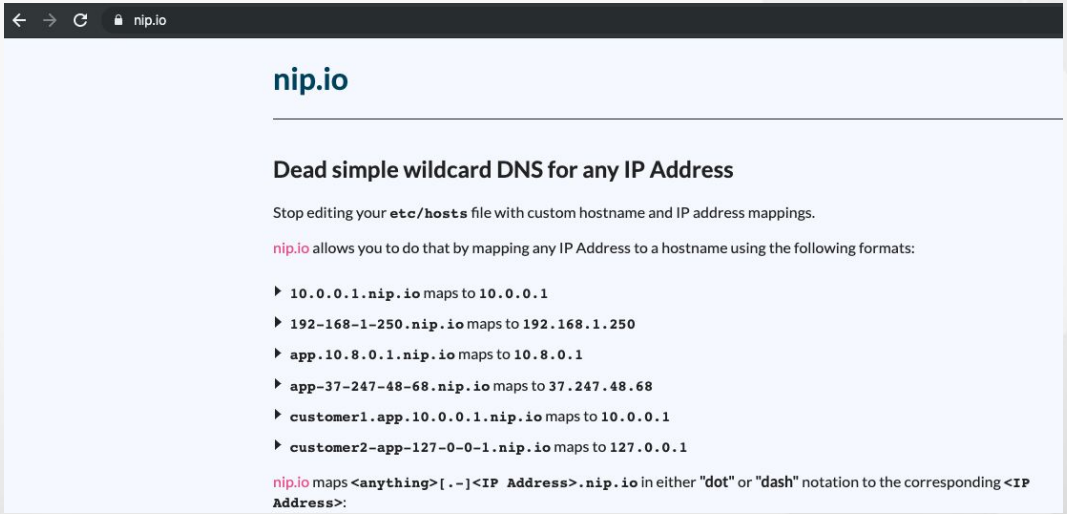
```
Error. Requests to this address are not allowed. Please try again.
```

SSRF - Bypass

- 127.0.0.1 is a /8 Network
  - 127.127.127.127
  - 127.0.0.2
  - 127.255.255.254
  - etc

SSRF - Bypass

- Use hostnames that resolves as internal IP:
  - [whatever].ip.ad.dr.es.nip.io -> ip.ad.dr.es
- e.g:
  - owaspudine.127.0.0.1.nip.io -> 127.0.0.1



SSRF - Bypass

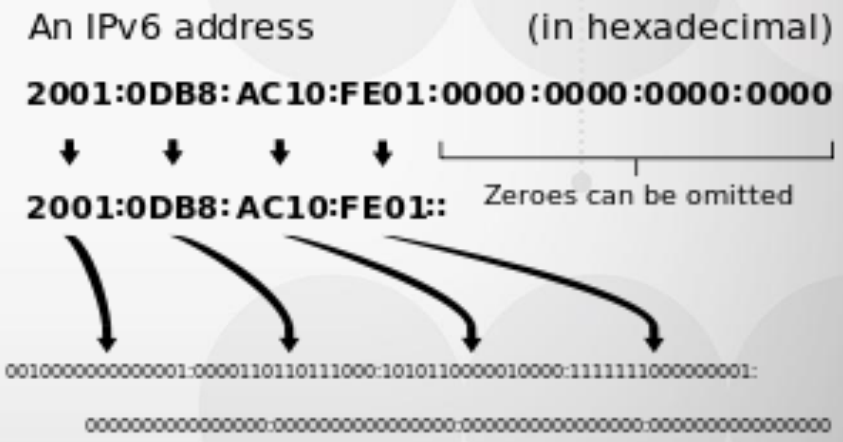
```
sh-3.2$ nslookup owaspdayudine.127.0.0.1.nip.io
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
Name:   owaspdayudine.127.0.0.1.nip.io
Address: 127.0.0.1

sh-3.2$
```

SSRF - Bypass

- Use IPV6
  - 0:0:0:0:0:0:0:1 -> 127.0.0.1
  - ::1 -> 127.0.0.1



SSRF - Bypass

- Use encoding:
  - Decimal 2130706433 -> 127.0.01
  - Hex 0x7f.0x0.0x0.0x1 -> 127.0.0.1
  - Octal 0177.0.0.001 -> 127.0.0.1

SSRF - Bypass

- If the server side client is configured to follow redirect we can abuse this functionality by configuring the web server or creating a simple page

```
<?php header("location: http://127.0.0.1"); ?>
```

HTTP 301

From Wikipedia, the free encyclopedia

The HTTP respons
that the response i
redirect is consider

HTTP 302

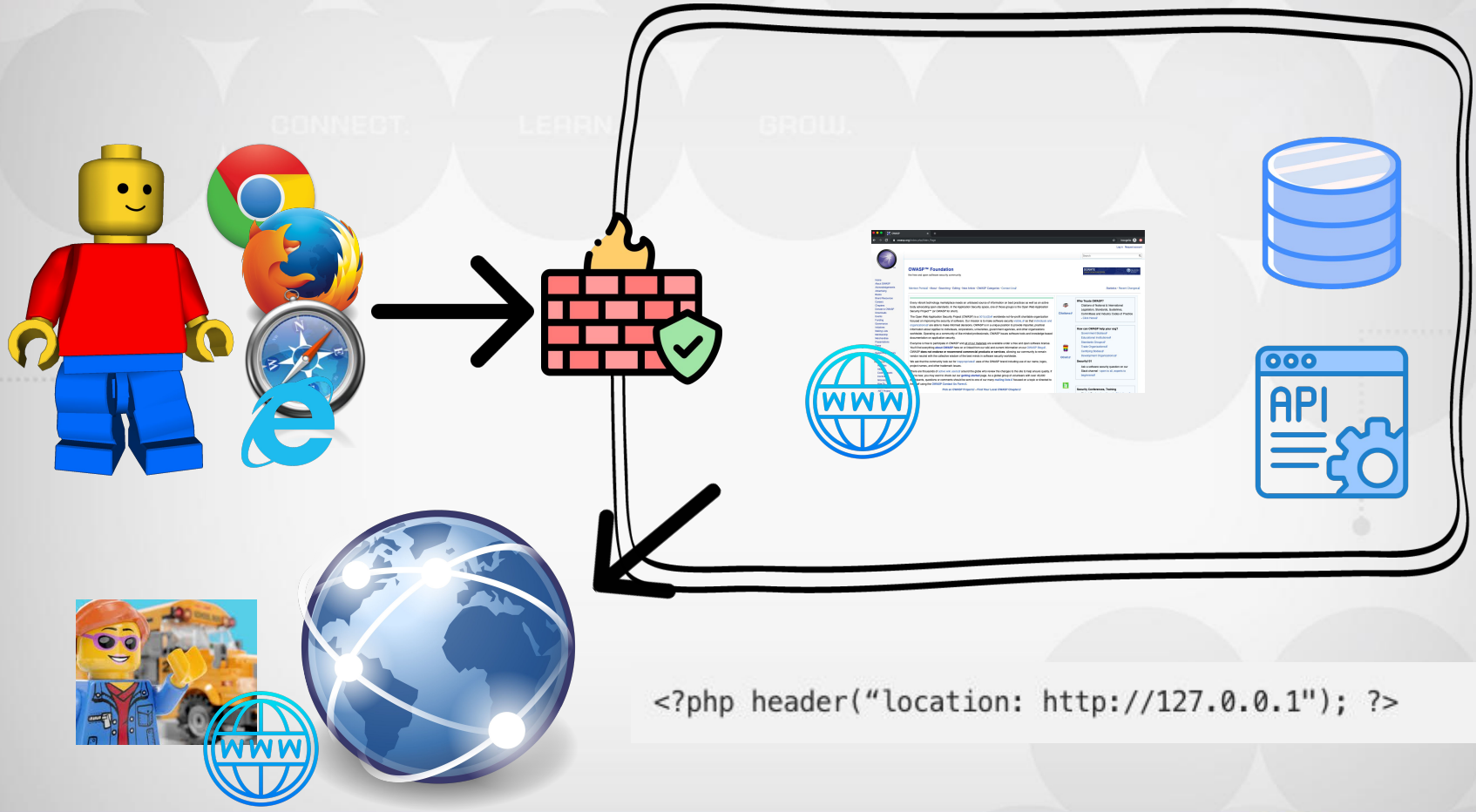
From Wikipedia, the free encyclopedia

The HTTP response status code **302 Found** is a common way of performing URL redirection. The HTTP/1.0 specification (RFC 1945) initially defined this code, and gave it the description phrase "Moved Temporarily" rather than "Found".

SSRF - Example

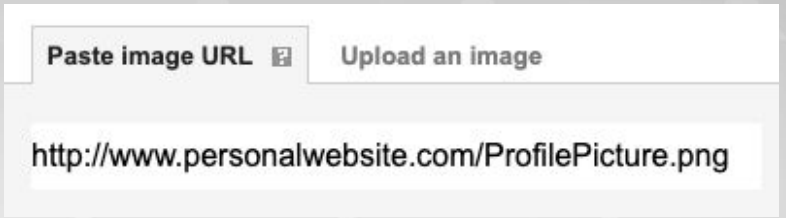
Paste image URL | Upload an image

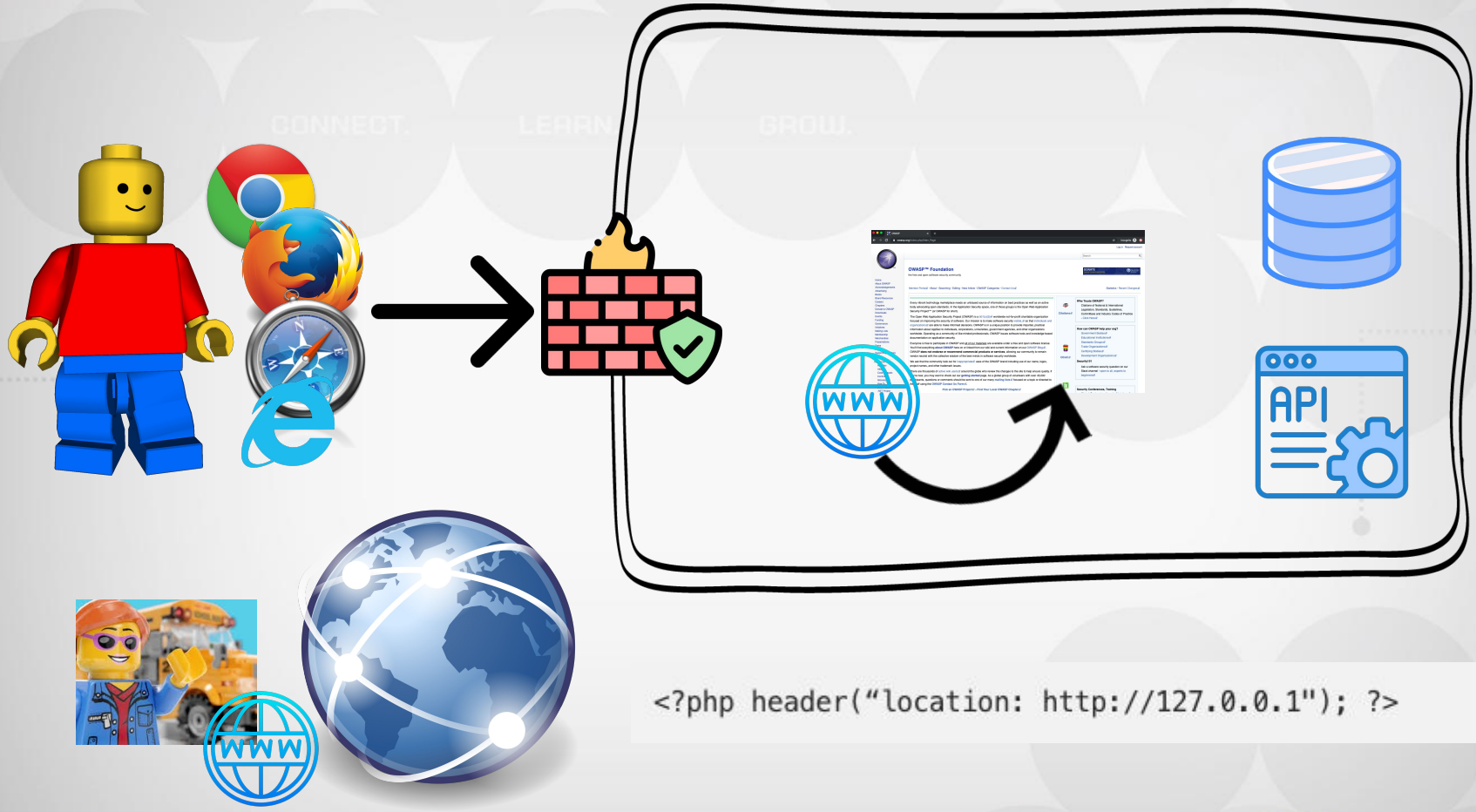
http://www.personalwebsite.com/ProfilePicture.png




```
<?php header("location: http://127.0.0.1"); ?>
```

SSRF - Example






```
<?php header("location: http://127.0.0.1"); ?>
```

SSRF - Bypass

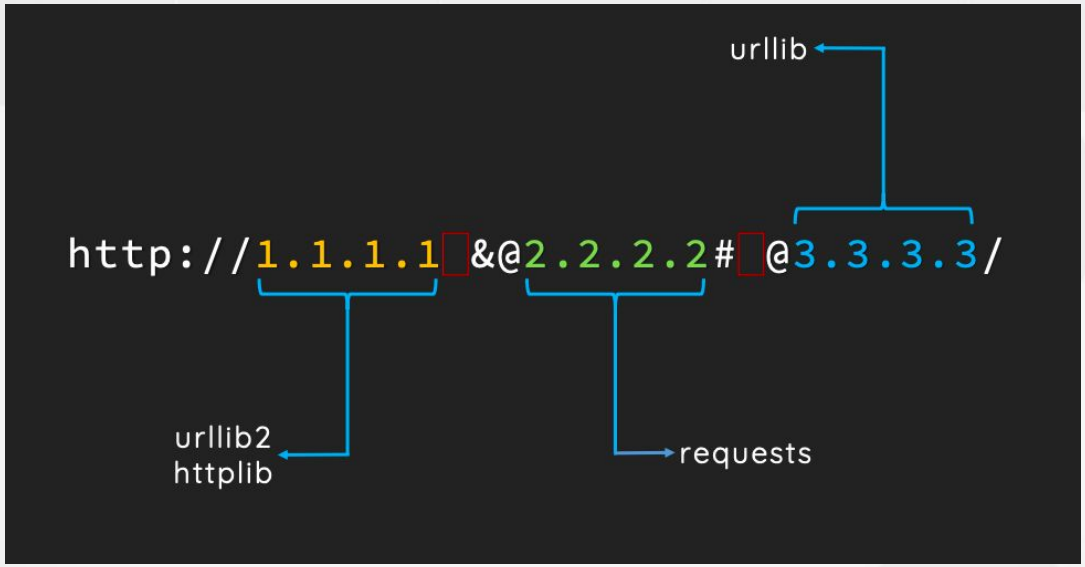
- Protocol Smuggling

```
http://1.1.1.1 &@2.2.2.2# @3.3.3.3/
```

https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf

SSRF - Bypass

- Protocol Smuggling



https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf

SSRF - Test

```
def check_hostname(hostname):
    """
    Resolve the hostname and check the IP address.
    Returns True if the IP address is included in these ranges:
        - 127.0.0.1/8
        - 192.168.0.0/16
        - 10.0.0.0/8
        - 169.254.169.254
    Otherwise returns False.
    """
    ...
    ...

if check_hostname(hostname):
    print("The ip you submitted is not valid")
    return False

requests.get('http://%s' % hostname)
...
```

SSRF - Test

```
def check_hostname(hostname):
    """
    Resolve the hostname and check the IP address.
    Returns True if the IP address is included in these ranges:
        - 127.0.0.1/8
        - 192.168.0.0/16
        - 10.0.0.0/8
        - 169.254.169.254
    Otherwise returns False.
    """
    ...
    ...

if check_hostname(hostname):
    print("The ip you submitted is not valid")
    return False

requests.get('http://%s' % hostname)
...
```

The first time the resolution is requested the IP address is resolved as a random public IP address with a Time To Live (TTL) of few milliseconds

SSRF - Test

```
def check_hostname(hostname):
    """
    Resolve the hostname and check the IP address.
    Returns True if the IP address is included in these ranges:
        - 127.0.0.1/8
        - 192.168.0.0/16
        - 10.0.0.0/8
        - 169.254.169.254
    Otherwise returns False.
    """
    ...
    ...

if check_hostname(hostname):
    print("The ip you submitted is not valid")
    return False

requests.get('http://%s' % hostname)
...
```

When this method is called the TTL is already expired and a new DNS resolution request is made that returns an internal IP address like 127.0.0.1

SSRF - New Mitigations

- Amazon recently (November 2019) introduced a new protection in the IMDSv2:
  - You need to generate a token with a PUT request before performing further request to the server

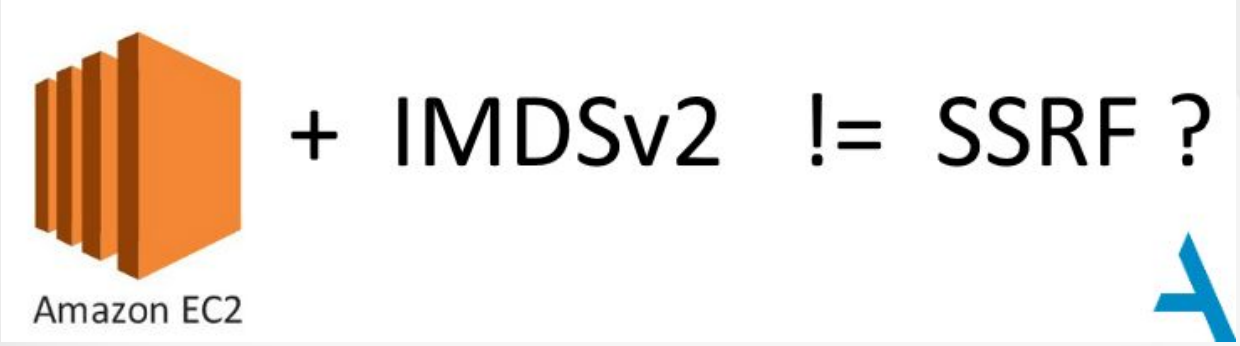
```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds
: 21600"`

curl http://169.254.169.254/latest/meta-data/profile -H "X-aws-ec2-metadata-token: $TOKEN"
```

https://aws.amazon.com/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/

SSRF - New Mitigations

- Do we need to stop to look for SSRF on cloud ?
  - Legacy Server
  - Not enabled by default



https://blog.appsecco.com/server-side-request-forgery-ssrf-and-aws-ec2-instances-after-instance-meta-data-service-version-38fc1ba1a28a

SSRF - More

hackerone.com/hacktivity?querystring=SSRF&filter=type:all&order_direction=DESC&order_field=latest_disclosable_activity_at

Hacktivity | Directory | Inbox | Hacker Dashboard

Hacktivity

See the latest hacker activity on HackerOne

SSRF

Sort

- Popular
- New

Type

- All
- Bug Bounty
- Published
- Disclosed

150 results for "**SSRF**".

5 — Server-Side Request Forgery (SSRF)
By t-pwn to U.S. Dept Of Defense | Resolved | Medium

15 — SSRF vulnerability on [redacted] leaks internal IP and various sensitive inform
By alyssa_herrera to U.S. Dept Of Defense | Resolved | Medium

3 — Incorrect logic in MySQL & MariaDB protocol leads to remote SSRF/Remote file read
By squashbroom to The Internet | Resolved

HackerOne Hacktivity

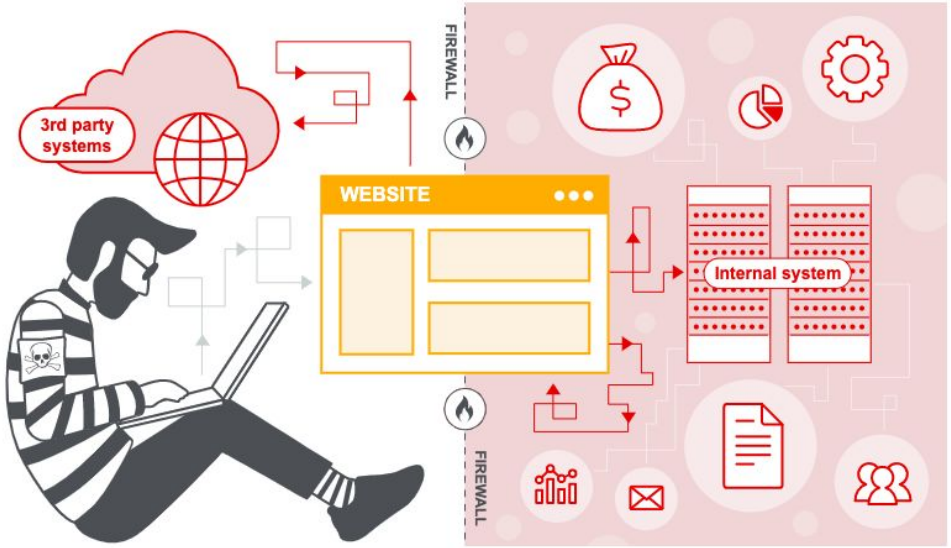
https://hackerone.com/hacktivity?querystring=SSRF&filter=type:all&order_direction=DESC&order_field=latest_disclosable_activity_at

SSRF - More

What is SSRF?

Server-side request forgery (also known as SSRF) is a web security vulnerability that allows an attacker to induce the server-side application to make HTTP requests to an arbitrary domain of the attacker's choosing.

In typical SSRF examples, the attacker might cause the server to make a connection back to itself, or to other web-based services within the organization's infrastructure, or to external third-party systems.



PortSwigger Lab

https://portswigger.net/web-security/ssrf

SSRF - References

- Server Side Request Forgery
  - https://www.owasp.org/index.php/Server_Side_Request_Forgery
- SSRF Payloads
  - https://medium.com/@pravinponnusamy/ssrf-payloads-f09b2a86a8b4
- Payload All The Things
  - https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Request%20Forgery
- Intro to SSRF
  - https://medium.com/swlh/intro-to-ssrf-beb35857771f
- Amazon AWS IMDSv2
  - https://aws.amazon.com/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/
- Server-side request forgery (SSRF)
  - https://portswigger.net/web-security/ssrf

Q&A

Thank You!