Source: Bing Chat AI

LABORATORY 2. OSINT AND SECURE BROWSING

DEPARTMENT OF COMPUTER SCIENCE. UNIVERSITY OF OVIEDO

Computer Security | 2023 – 2024 (V4.0 *"S-81 Isaac Peral"*)

Content

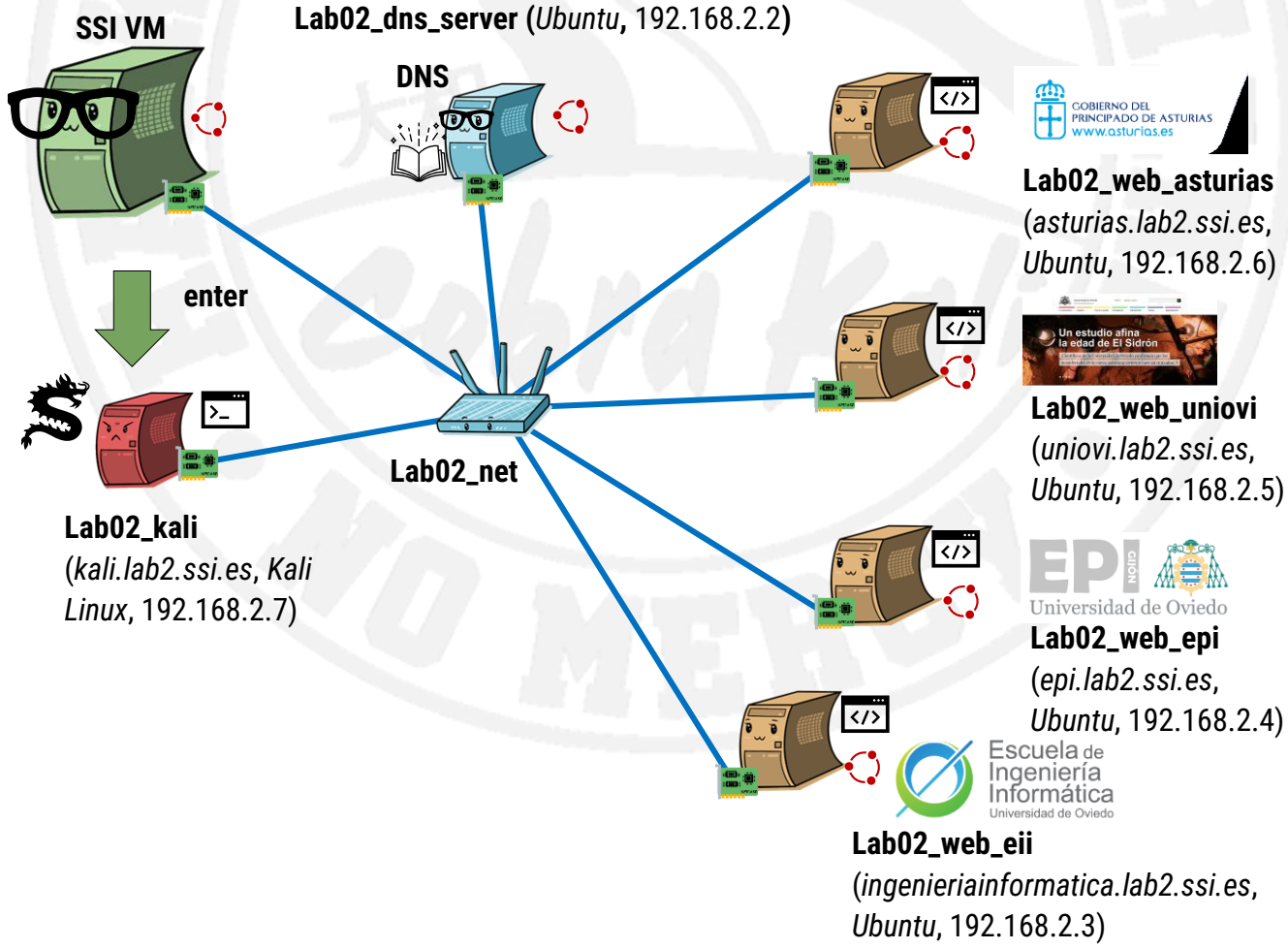
The Infrastructure of this laboratory

NOTE: Before beginning, please make sure that your course virtual machine has a series of packages already installed, so the labs can work correctly. This requires you to **run the following command**: `sudo apt install gnome-keyring gnupg2 pass`

First, ensure that you have the `ssi_labs` folder structure we provided to you inside your VM. This lab (`lab_02`) consists of an infrastructure composed by several containers. Not all activities require using all of them. Please see the following diagram to fully understand the infrastructure layout:

- **An internal LAN**: (*lab02_net*, `192.168.2.0/24`)
- **4 different web servers**
- **A DNS**: providing names for each web server in the domain `lab2.ssi.es`.
- **A Kali machine** (`lab2_kali`) with a couple of tools installed **for enumeration purposes only**. These are described through this document. This also means you do not have to install them if you do not want. This *Kali* machine is the starting point of the lab.

IMPORTANT: **Make sure you have read and understood the instructions of the "*SSI Automated Infrastructures*" document to understand how to work with this type of automated lab infrastructures.**



Block 1: Exploring web contents

Exercise L2B1_ROBOTS: robots.txt file

Activity description / Practical application

You need to know if a website is revealing "secret" routes through this public file

Expected results

To complete this activity, you need to pick any web you like and analyze its `robots.txt` file, determining if you can find something interesting or compromising security and why.

Additional information required to do it

Part of the **Robot Exclusion Standard**, this file is used to tell search engines that follow this standard to not to index the URLs corresponding to certain entries found under the keyword `Disallow` in a file that is always named `robots.txt`. These files are always accessible using this URL pattern: `<My URL>/robots.txt`.

Some developers use it so that *Google* or other search engines do not index "sensitive" content. The problem is that this file is public and accessible so engines can read its contents, and if engines can read it, you can too! 😊. You can find more information here: https://www.synopsys.com/blogs/software-security/robots-txt/

Exercise L2B1_METADATOS: Document metadata

Activity description / Practical application

You need to know if documents, images, etc. of a website are revealing too much information about their owners or authors

Expected results

This activity will be completed when you pick any local or downloaded file of the accepted types and analyze it with the web or the `exiftool` program to see if you find something interesting. You can answer these questions:

- *What would you do if you found IPs?*
- *What would you do if you found people's names?*
- *What would you do if you found app names? What do you think that data could reveal?*

Additional information required to do it

Metadata is data stored attached to a file, **but not as part of its contents, but normally in their file headers**. Metadata can be attached to a file in clear text or ciphered. Lots of people ignore that files may have

metadata. Even worse, **lots of people also do not know that lots of programs automatically put metadata in the files they generate without their consent or being aware of it**.

🔍 ***The importance of this (meta)data may vary from trivial to truly compromising information. Nowadays, lots of services automatically remove metadata from files you upload (social networks do), but you should not trust them to do it 100% of the time (bugs!). They may keep the original metadata from their own use (marketing), even if they remove it from the files displayed to the public. Therefore, you should remove metadata from the files you upload using appropriate programs just in case*** 😊**.**

FOCA is a *Windows* program that extracts and analyzes metadata from different file types, that also implements searching in domains, DNS servers, URLs, and published documents. FOCA can be downloaded from here: https://github.com/ElevenPaths/FOCA. There is a free open-source version and a commercial version. Unfortunately, it requires to set up a *SQL Database* (*SQL Express* or *SQL Server*) that takes time and effort to install.

This is not part of this course objectives so, instead of this tool, we are going to use a simpler online one from the same vendor that analyzes the metadata of individual documents and other file types: ***Metadata2Go*** (https://www.metadata2go.com/view-metadata)..

This web page accepts any file of a list of accepted file types and extracts its metadata to see if there is something useful or suspicious in them. If the webpage does not allow the type of file, we can use a local tool called `exiftool` that we will see in laboratory 4 with other applications. In this laboratory we will use it simply to **read the complete metadata of a file** that we have downloaded from the web (it supports zips, images of many classes, *Office* documents, PDF ...) using its options: https://manpages.ubuntu.com/manpages/trusty/en/man1/exiftool.1p.html. Please install it if not already available in the VM.

Block 2: Searching among exposed and past content

Exercise L2B2_GHACK: Google Hacking

Activity description / Practical application

You can use the Google Search engine to find compromising information about any target on Internet

Expected results

This activity will be completed when you choose a website you like and perform a couple of search operations with the *Google Hacking Database* to familiarize with the process, analyzing if you find something suspicious.

NOTE: *Google* will most probably ask you to verify your identity with a captcha after using 2 or 3 of these queries, to test that you are not a bot.

Additional information required to do it

Google Hacking is the ability to do certain manual search operations in the *Google Search Engine*, normally combined with the `site:` operator. This also applies to other search engines, although each engine uses their own operators. These search operations aim to detect vulnerabilities, configuration problems, or information that may lead to an attack.

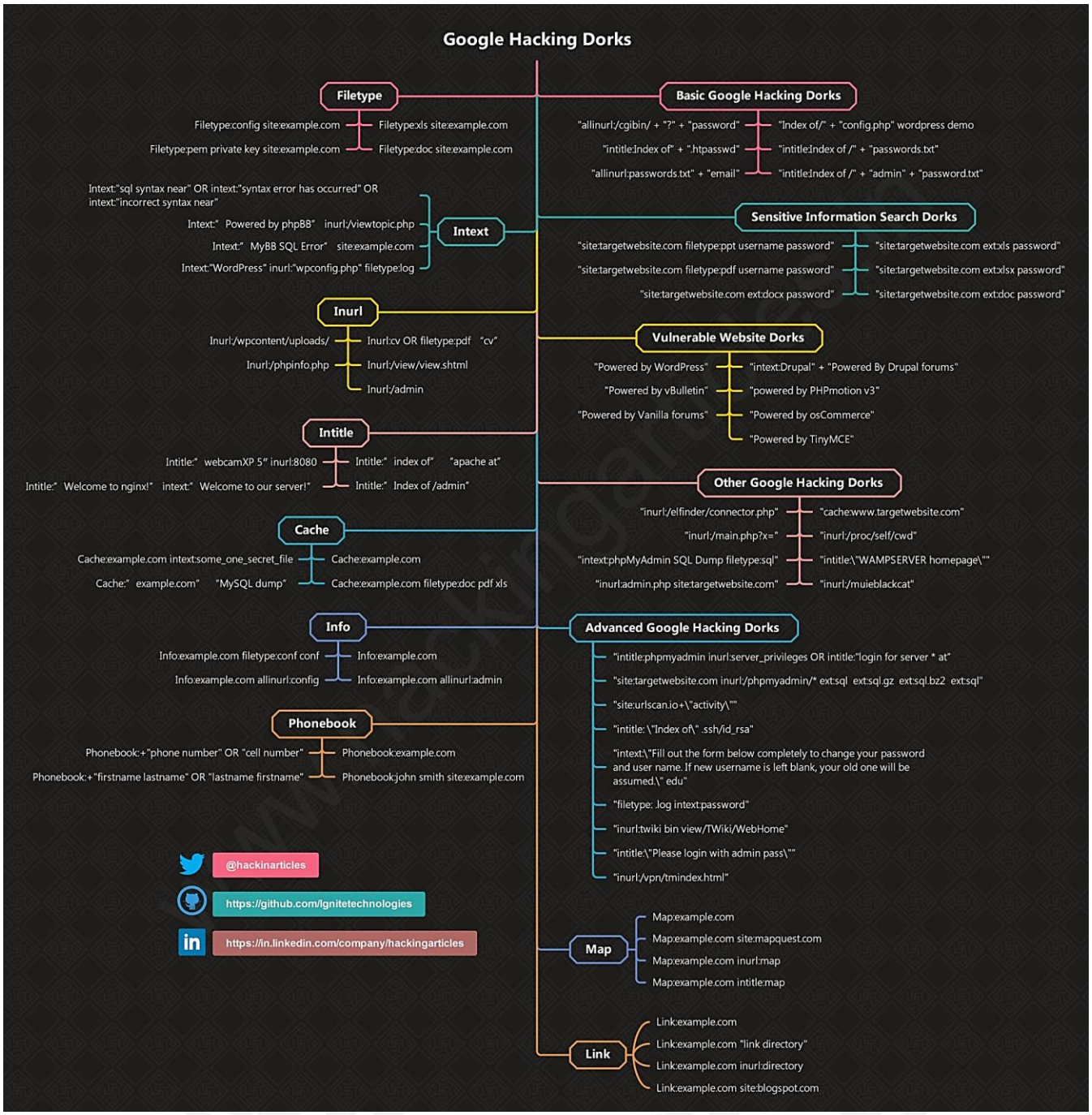
There are lots of predefined search queries that cover different types of attacks / information gathering procedures in the ***Google Hacking Database*** **(GHDB)**: https://www.exploit-db.com/google-hacking-database. The procedure to use this database is:

- Choose a search category and a query
- Perform the query over a concrete target (using the `site:` search operator, https://support.google.com/websearch/answer/2466433?hl=es) and see if the result matches the expected one, according to the search you chose.

You can find an image with the steps that this procedure requires **in the first seminar** of the course. We have the following categories of predefined searches

Footholds	Files Containing Juicy Info
Files Containing Usernames	Files Containing Passwords
Sensitive Directories	Sensitive Online Shopping Info
Web Server Detection	Network or Vulnerability Data
Vulnerable Files	Pages Containing Login Portals
Vulnerable Servers	Various Online Devices
Error Messages	Advisories and Vulnerabilities

You can also find URLs containing more examples **in the first seminar** and you can use this cheatsheet to help you implementing this exercise.



Exercise L2B2_SHODAN: Shodan

Activity description / Practical application

You can use *the* Shodan machine search engine to know information about any public endpoint on Internet

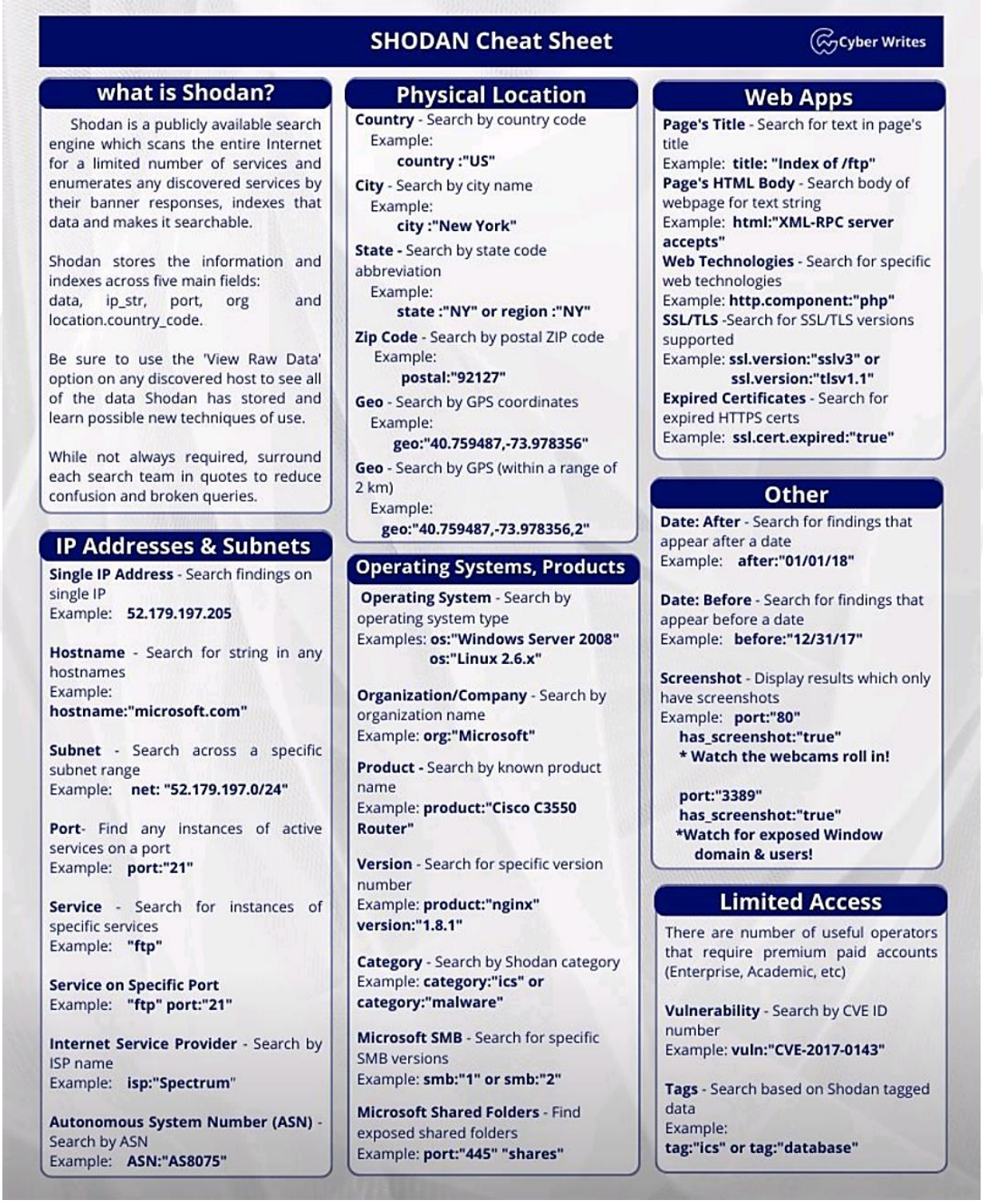
Expected results

This activity will be completed when you can perform a query in *Shodan* over a single URL or IP, analyzing the output you obtain and why you think this information may be a danger for the target.

🔍 ***You have more freedom if you create a free account, but it is not necessary to do this exercise if you limit yourself to single IPs.***

Additional information required to do it

Shodan (http://www.shodanhq.com/) is a search engine able to find devices instead of web pages: routers, servers, CCTV recording cameras, traffic lights, ... More details about its functionality and usage can be found **in the first seminar** (also including URLs with usage examples). You can also use this cheatsheet to implement this exercise:



SHODAN Cheat Sheet — Cyber Writes

what is Shodan?

Shodan is a publicly available search engine which scans the entire Internet for a limited number of services and enumerates any discovered services by their banner responses, indexes that data and makes it searchable.

Shodan stores the information and indexes across five main fields: data, ip_str, port, org and location.country_code.

Be sure to use the 'View Raw Data' option on any discovered host to see all of the data Shodan has stored and learn possible new techniques of use.

While not always required, surround each search team in quotes to reduce confusion and broken queries.

IP Addresses & Subnets

Single IP Address - Search findings on single IP
Example: **52.179.197.205**

Hostname - Search for string in any hostnames
Example:
hostname:"microsoft.com"

Subnet - Search across a specific subnet range
Example: **net: "52.179.197.0/24"**

Port- Find any instances of active services on a port
Example: **port:"21"**

Service - Search for instances of specific services
Example: **"ftp"**

Service on Specific Port
Example: **"ftp" port:"21"**

Internet Service Provider - Search by ISP name
Example: **isp:"Spectrum"**

Autonomous System Number (ASN) - Search by ASN
Example: **ASN:"AS8075"**

Physical Location

Country - Search by country code
Example:
country :"US"

City - Search by city name
Example:
city :"New York"

State - Search by state code abbreviation
Example:
state :"NY" or region :"NY"

Zip Code - Search by postal ZIP code
Example:
postal:"92127"

Geo - Search by GPS coordinates
Example:
geo:"40.759487,-73.978356"

Geo - Search by GPS (within a range of 2 km)
Example:
geo:"40.759487,-73.978356,2"

Operating Systems, Products

Operating System - Search by operating system type
Examples: **os:"Windows Server 2008"**
os:"Linux 2.6.x"

Organization/Company - Search by organization name
Example: **org:"Microsoft"**

Product - Search by known product name
Example: **product:"Cisco C3550 Router"**

Version - Search for specific version number
Example: **product:"nginx" version:"1.8.1"**

Category - Search by Shodan category
Example: **category:"ics" or category:"malware"**

Microsoft SMB - Search for specific SMB versions
Example: **smb:"1" or smb:"2"**

Microsoft Shared Folders - Find exposed shared folders
Example: **port:"445" "shares"**

Web Apps

Page's Title - Search for text in page's title
Example: **title: "Index of /ftp"**
Page's HTML Body - Search body of webpage for text string
Example: **html:"XML-RPC server accepts"**
Web Technologies - Search for specific web technologies
Example: **http.component:"php"**
SSL/TLS -Search for SSL/TLS versions supported
Example: **ssl.version:"sslv3" or ssl.version:"tlsv1.1"**
Expired Certificates - Search for expired HTTPS certs
Example: **ssl.cert.expired:"true"**

Other

Date: After - Search for findings that appear after a date
Example: **after:"01/01/18"**

Date: Before - Search for findings that appear before a date
Example: **before:"12/31/17"**

Screenshot - Display results which only have screenshots
Example: **port:"80"**
has_screenshot:"true"
*** Watch the webcams roll in!**

port:"3389"
has_screenshot:"true"
***Watch for exposed Window domain & users!**

Limited Access

There are number of useful operators that require premium paid accounts (Enterprise, Academic, etc)

Vulnerability - Search by CVE ID number
Example: **vuln:"CVE-2017-0143"**

Tags - Search based on Shodan tagged data
Example:
tag:"ics" or tag:"database"

Exercise L2B2_CENSYS: Censys

Activity description / Practical application

You can use the *Censys* machine search engine and other auxiliary tools to know information about any public endpoint on Internet

Expected results

This activity will be completed when you can use Censys to scan an IP or network range (for example one range that belongs to Uniovi). After doing that, you must implement the following operations:

- Analyze the output it provides and the information that is displayed about each host.
- Analyze this information to determine why it could be dangerous from an attack point of view.
- Since Censys lacks the geolocation map that Shodan provides, the same functionality must be achieved by combining the https://www.iplocation.net/ web and *Google Maps* (which supports latitude and longitude coordinates)

Additional information required to do it

Censys is like *Shodan*, but it presents information in a more structured way. It allows you to do search operations based on the data it obtains from the servers it analyzes all over the Internet. **The first seminar** offers more details about it. *Censys* usage examples can be found here:

- https://search.censys.io/search/examples?resource=hosts
- https://support.censys.io/hc/en-us/articles/360059720271-Search-2-0-Example-Host-Queries

You can also use this cheatsheet to help you implementing this exercise:

Cheatography

Censys Search Cheat Sheet

by himajedi via cheatography.com/196419/cs/41272/

IP Addresses and Subnets

Single IP (*supports IPv4 and IPv6*)	8.8.8.8 or ip:8.8.8.8
Subnet by CIDR	ip: "23.0.0.0/8"
Subnet by IP Range	ip: [1.12.0.0 to 1.15.255.255]
Hostname	dns.names:"*.zip"
Autonomous System # (ASN)	autonomous_system.asn:16509
Autonomous System Name	autonomous_system.name:"AMAZON-02"
IPv6 hosts	ip: "2001::/3" or labels:ipv6

Ports, Protocols, and Software

Port	services.port:22 services.port:[20 to 25] services.port:{20,21,22}
Number of open ports on host	service_count: [1 to 20]
Service / Protocol	services.service_name:SSH
Transport protocol	services.transport_protocol:TCP
Software by product and/or vendor	services: (software.vendor:"Apache" AND software.product:"HTTPD")
Software by URI / CPE	services.software.cpe =`cpe:2.3:o:mikrotik:routeros:*:*:*:*:*:*:*:*`
Banner grab	services.banner:"HTTP/"

Geography

Country	location.country:"United States"
City	location.city:"Ann Arbor"
State	location.province:"Michigan"
GPS Coordinates	(location.coordinates.latitude=41.85003 AND location.coordinates.longitude=-87.65005)

Pro tip: Use Map To Censys to draw a box over the geographic area of interest and click "Open in Search" to see hosts in the area

Labels

search by label	labels:`<la bel -na me>`

Some useful host labels: scada, c2, login-page, open-dir, network.device, cryptocurrency, managed-file-transfer, ipv6. See a breakdown of common labels here

Handy Censys Search CLI JQ filters

List of IP addresses	'.[].ip'
Banners	'.[] | .ip as $ip | .services[] | [$ip, .transport_protocol, .port, .service_name, .banner]'

Usage: `censys search <qu ery> | jq <fi lte r>`

Web Entities (HTTP/S)

HTML Title	services.http.response.html_title:"dashboard"
Response Body	services.http.response.body:"login"
HTTP Status code	services.http.response.status_code=200
Server header	services.http.response.headers.Server:"nginx"
Certificate Issuer	services.tls.certificates.leaf_data.issuer.organization:"Let's Encrypt"
Certificate Subject Common Name	services.tls.certificates.leaf_data.subject.common_name:`*.hero kua pp.com`
TLS version (*Highest negotiated version*)	services.tls.version_selected:"TLSv1_1"
Favicon MD5 Hash	services.http.response.favicons.md5_hash:*

Use Case Examples

Hacked web servers	services: (service_name:"HTTP" and http.response.html_title:"hacked by")
U.S. Government or Military hosts	dns.names: *.gov or dns.names:* .mil or name: *.gov or name:* .mil
Hosts serving login pages with port 22 open	services.port:22 and labels:`login-page`
Servers in Russia running remote access protocols	location.country:"Russia" and labels:`remote -access`
Filter out hosts with 100+ ports open	services.truncated: false
Compromised MikroTik routers	services.service_name: MIKROTIK_BW and "HACKED"

Pro tip: Get more results by including virtual hosts -- click the gear icon and toggle **Virtual Hosts: INCLUDE**

By **himajedi**
cheatography.com/himajedi/

Published 14th November, 2023.
Last updated 2nd December, 2023.
Page 1 of 2.

Exercise L2B2_WAYBACK: The Wayback Machine

Activity description / Practical application

You can use the *Internet Archive* (aka "*Wayback Machine*") search engine to know information about past published websites on many Internet domains

Expected results

This activity will be completed when you are able to use the *Wayback Machine* to

- **Locate the historic URLs list of any website you like**, any interesting file you choose, and the different historic versions of any document or webpage that was in the site. Once you can do that, you also must think about what this could mean from the security point of view (the questions at the end of the lab may help you to analyze it).
- **Locate old posts from a public account of the social network of your** choice from a public figure. You choose the date. *Do you think the Wayback Machine can locate posts that have been deleted long after they were published?*

Additional information required to do it

*(**NOTE**: The Wayback Machine is massive, and therefore is sometimes quite slow. It is not your connection, or your ISP, it is just like that, and there is nothing you can do about it 😊)*

🔍 ***Most people (including a sizable amount of web administrators!) are used to search for compromising content in web pages to remove it, but they do not consider what happens with the content that WAS there, but no longer do, or just delete the content without considering that the Wayback Machine could have archived it. This content may prove especially useful to attackers, and this lab section will teach you how to deal with it.***

There is a reference web page to know about the "history" of any website on Internet and its name is "***The Wayback Machine***" (aka *The Internet Archive*): https://archive.org/web/. This website is a date-sorted massive archive of previous contents displayed by any public website that was active on Internet in the past, but it also has "hacking" usages.

For example, you can inspect **ALL the URLs** that have been extracted from a particular domain using a query like this: http://web.archive.org/web/*/<URL>/* (e. g. http://web.archive.org/web/*/http://www.uniovi.es/*). Please note that the URL load is asynchronous, so it may take a while. Do not worry if you initially find an empty table (for Uniovi, it usually takes 30s to load).

Once we have the historic URL list of a site, we can use this information to obtain a series of remarkably interesting information about it. You can find more details here: https://www.elladodelmal.com/2013/04/hacking-con-archivecom-wayback-machine.html

- The URL table may be processed with code to extract all the URLs, or the **URLs of a certain type** that may be interesting for a certain purpose. For example, you can extract images or documents so you can inspect its contents (or metadata! 😉 😉). This can also be done in the *Wayback Machine* web page, as it has a filter textbox.

100,000 URLs have been captured for this domain.

Filter results (i.e. '.txt'): pdf

URL	MIME TYPE	FROM	TO	CAPTURES	DUPLICATES	UNIQUES
http://www.uniovi.es/-/defecto.pdf	text/html	Jun 16, 2013	Jul 15, 2013	2	0	2
http://www.uniovi.es/-/file.pdf	text/html	Jun 16, 2013	Jul 15, 2013	2	0	2
http://www.uniovi.es/-/preistabelle.pdf	text/html	Jun 16, 2013	Jul 15, 2013	2	0	2
http://www.uniovi.es/-/presitabelle.pdf	text/html	Jun 16, 2013	Jul 15, 2013	2	0	2
http://www.uniovi.es/aal/archivos_pdf/neutro_materia.pdf	text/html	Jun 6, 2011	Dec 15, 2018	7	6	1
http://www.uniovi.es/accesoyayudas/estudios/373/Oficio+de+Adjunto+Plantilla+Solicitud.pdf/5412a9a3-9730-40a4-8	text/plain	Sep 4, 2014	Sep 4, 2014	1	0	1
http://www.uniovi.es/accesoyayudas/tramites/tramite/-/asset_publisher/v9UAvM9qJpFc/content/wwww.uniovi.es/documents/31582/243104/INSTANCIA+VICERRECTOR+ESTUDIANTES_131028.pdf/d7cdc975-d56c-404f-b5a0-ef50be637791	text/html	Sep 8, 2014	Sep 8, 2014	1	0	1
http://www.uniovi.es/Alfonso_Garcia_Leal/1993478.pdf	text/html	Jan 19, 2012	Apr 12, 2012	2	1	1
http://www.uniovi.es/Areas/Mecanica.Fluidos/becasempleo/investigacionaerodinamica_paniaguaSMALL.pdf	unk	Nov 11, 2014	Nov 11, 2014	1	0	1
http://www.uniovi.es/Areas/Mecanica.Fluidos/docencia/_asignaturas/maquinas_de_fluidos/Lecc6_r1.pdf	unk	Apr 12, 2017	Apr 12, 2017	1	0	1
http://www.uniovi.es/Areas/Mecanica.Fluidos/docencia/_asignaturas/maquinas_de_fluidos/Presenta_Leccion3.pdf	unk	Apr 12, 2017	Apr 12, 2017	1	0	1
http://www.uniovi.es/Areas/Mecanica.Fluidos/docencia/_asignaturas/mecanica_de_fluidos/05_06/8.%20FLUJO_CONDUCTOS.pdf	unk	Jul 29, 2015	Jul 29, 2015	1	0	1

- You can also obtain the contents of **interesting files** of a web. E. g.: the `robots.txt` file
- **History of any file on the website**: The URL table column shows the number of different copies that any file has had over time. This means that we can locate any version of any file that has been indexed. You can iterate through them in the calendar view once you click in the file

Saved **4 times** between June 28, 2012 and July 11, 2013.

1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 **2013** 2014 2015 2016 2017 2018

JAN

	1	2	3	4	5	
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

FEB

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28		

MAR

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

APR

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

MAY

		1	2	3	4	
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

JUN

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

JUL

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

AUG

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Block 3: Inspecting IPs and domains

Exercise L2B3_DOMINIOS: Sub-domains discovery

Activity description / Practical application

You need to know if a particular domain name also has registered subdomains

Expected results

This activity will be completed when you are able to enumerate all the subdomains of `uniovi.es` with any of the web-based provided tools. Command-line tools must be used against the domain included in the infrastructure of this lab, `lab02.ssi.es`.

We recommend using at least one of the command-line tools and one of the websites provided to gain better understanding of how to perform this activity. Evaluating all the mentioned tools is NOT REQUIRED.

Additional information required to do it

Domain names are very curious because normally most people know the main one but, most of the times, there are **secondary subdomains** that are not so popular, only known by some people, or just forgotten. In any case, they suppose more vectors of enumeration, and hence the potential of discovering more things about a target. There are lots of ways of discovering this information automatically. These are some of them:

- `knockpy` (https://github.com/guelfoweb/knock). *Python* tool designed to enumerate subdomains on a target domain through a wordlist. On *Ubuntu*, you can install it with `sudo apt install knockpy` and run a basic scan with `./knockpy <target>`. You can find full usage instructions it their repository README.
- `dnsenum` (https://github.com/fwaeytens/dnsenum): E. g. `dnsenum uniovi.es`. If no `-f` tag is supplied, it will default to `/usr/share/dnsenum/dns.txt` or a `dns.txt` file in the same directory as `dnsenum.pl`

NOTE: A few students have reported that `dnsenum` may freeze their labs when is launched. If it is your case, quickly cancel the operation (CRTL+C) and use another tool. This is due to a VirtualBox installation problem, but we could not trace its exact origin, as it does not happen to every student.

dnsenum 1.2.6 Cheatsheet (ingenieriainformatica.uniovi.es)

multithreaded perl script to enumerate DNS information of a domain

https://github.com/fwaeytens/dnsenum

GENERAL USAGE

dnsenum [Options] <domain>

NOTES

If no -f tag supplied will default to /usr/share/dnsenum/dns.txt or the dns.txt file in the same directory as dnsenum.pl

OPTIONS

GENERAL OPTIONS

- **--dnsserver <server>**: Use this DNS server for A, NS and MX queries.
- **--enum**: Shortcut option equivalent to --threads 5 -s 15 -w.
- **-h, --help**: Print this help message.
- **--nocolor**: Disable ANSIColor output.
- **--noreverse**: Skip the reverse lookup operations.
- **--private**: Show and save private ips at the end of the file domain_ips.txt.
- **--subfile <file>**: Write all valid subdomains to this file.
- **-t, --timeout <value>**: The tcp and udp timeout values in seconds (default: 10s).
- **--threads <value>**: The number of threads that will perform different queries.
- **-v, --verbose**: Be verbose: show all the progress and all the error messages.

GOOGLE SCRAPING OPTIONS

- **-p, --pages <value>**: The number of google search pages to process when scraping names, the default is 5 pages, the -s switch must be specified.
- **-s, --scrap <value>**: The maximum number of subdomains that will be scraped from Google (default 15).

BRUTE FORCE OPTIONS

- **-f, --file <file>**: Read subdomains from this file to perform brute force. (Takes priority over default dns.txt)
- **-r, --recursion**: Recursion on subdomains, brute force all discovered subdomains that have an NS record.
- **-u, --update <a|g|r|z>**: Update the file specified with the -f switch with valid subdomains.

a (all)	Update using all results.
g	Update using only google scraping results.
r	Update using only reverse lookup results.
z	Update using only zonetransfer results.

```
operario@kali:~$ dnsenum uniovi.es
dnsenum VERSION:1.2.6

-----   uniovi.es   -----

Host's addresses:
__________________

uniovi.es.                               1216     IN    A        156.35.233.105

Name Servers:
_____________

enol.si.uniovi.es.                       172800   IN    A        156.35.14.2
chico.rediris.es.                        6186     IN    A        162.219.54.2
zeus.etsimo.uniovi.es.                   1800     IN    A        156.35.23.24
sun.rediris.es.                          3781     IN    A        199.184.182.1
solid.net.uniovi.es.                     1800     IN    A        156.35.11.170

Mail (MX) Servers:
__________________

mx02.puc.rediris.es.                     30       IN    A        130.206.19.162
mx02.puc.rediris.es.                     30       IN    A        130.206.19.130
mx01.puc.rediris.es.                     30       IN    A        130.206.19.162
mx01.puc.rediris.es.                     30       IN    A        130.206.19.130

Trying Zone Transfers and getting Bind Versions:
________________________________________________


Trying Zone Transfer for uniovi.es on enol.si.uniovi.es ...
AXFR record query failed: REFUSED

Trying Zone Transfer for uniovi.es on chico.rediris.es ...
AXFR record query failed: REFUSED
```

WHOIS NETRANGE OPTIONS:

- **-d, --delay <value>**: The maximum value of seconds to wait between whois queries, the value is defined randomly, default: 3s.
- **-w, --whois**: Perform the whois queries on c class network ranges. **Warning**: this can generate very large netranges and it will take lot of time to perform reverse lookups.

REVERSE LOOKUP OPTIONS:

- **-e, --exclude <regexp>**: Exclude PTR records that match the regexp expression from reverse lookup results, useful on invalid hostnames.

OUTPUT OPTIONS:

- **-o --output <file>**: Output in XML format. Can be imported in MagicTree (www.gremwell.com)

- https://dnsdumpster.com/: Website that allow us to locate subdomains of a given domain.
- https://searchdns.netcraft.com/: Another website that allow us to locate subdomains of a given domain.
- https://www.virustotal.com : If we go to “Search”, provide a DNS name (e. g. uniovi.es), and perform a URL detection scan like the one we previously saw, the “Details” tab also gives its associated subdomains.
- https://crt.sh/: Providing a domain name (e. g. uniovi.es) also gives us associated subdomains.

Finally, it is possible that some of these domains may be, as we said, **abandoned**. In this case we have two options:

- A **truly abandoned** domain (no content)
- An **unmaintained domain** (potentially vulnerable contents, old versions of services, unpatched, unfixed…).

These are not the only tools that can do this work, but they cover most use cases. `Knockpy` and `dnsenum` require installation to use them, but they are available in the standard *Kali Linux* packages. You can find these installed in the *Kali* container of the **Lab 2 infrastructure**.

Exercise L2B3_RANGO_IP: Target an IP range

Activity description / Practical application

You need to know the assigned IP ranges of any internet domain

Expected results

This activity will be completed when you can check the IP range assigned to `uniovi.es` or any `.com` organization you want.

Additional information required to do it

Once we know the DNS domains and subdomains of any organization, we can search its assigned public IP ranges to know which IPs could be used by this organization machines. This enables us to inspect for machines or services exposed belonging to this organization. For .com sites, we can check https://whois.arin.net. To do this, we can input the organization name (e. g. "Microsoft", "Yahoo") in the *SEARCH WHOIS-RWS* textbox and scroll down to see the IP ranges this organization have assigned.

For `.es` domains, we can use www.nic.es for the same purpose. Clicking in the details of a domain ("*View data*") shows us the DNS servers that serve the different networks assigned to the organization, from which we can easily extract IPs and IP ranges.

Exercise L2B3_NMAPSCAN: LAN scans with nmap

Activity description / Practical application

You need to inspect a Local Area Network for "alive" machines

Expected results

This activity will be completed when you are able to locate active hosts in the mentioned LAN and its DNS names. You can answer this question: *Do you understand the relationship between the three activities in this block (Domain->Subdomains->IP Ranges of each subdomain->nmap) and what it would allow you to do?*

Additional information required to do it

The purpose of this activity is to locate all active machines in a LAN network (not from Internet!). To do that, we provide it the LAN network of the **Lab 2 infrastructure**. This extends the **Lab 1** activity, that only determines if a server is "alive". Use a *list scan* from this cheatsheet to see if the DNS names of the alive hosts are also provided.

Nmap 7.80 Cheatsheet series (ingenieriainformatica.uniovi.es)

Part 1: Reconnaissance (Basic)

https://nmap.org/

GENERAL USAGE

nmap [Scan Type(s)] [Options] {target specification}

NOTES

Target specifications can be host names, IP addresses, ranges, networks, etc. (scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254)

Use these options to locate "alive machines" (sometimes only returns that, sometimes they also return some port / service information)

TARGET SPECIFICATION

-iL <inputfilename>: Input from from file a list of hosts/networks

-iR <num hosts>: Choose random targets

--exclude <host1[,host2][,host3],...>: Exclude hosts/networks

--excludefile <exclude_file>: Exclude list from file

RECONOISSANCE EXAMPLES

nmap -sn scanme.nmap.org

sudo nmap -PS22,80 scanme.nmap.org

sudo nmap --traceroute scanme.nmap.org

```
operario@kali:~$ nmap -sn 192.168.20.10
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-21 18:38 CEST
Nmap scan report for 192.168.20.10
Host is up (0.00085s latency).
Nmap done: 1 IP address (1 host up) scanned in 13.01 seconds
```

```
operario@kali:~$ sudo nmap -PS22,80 192.168.20.10
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-21 18:42 CEST
Nmap scan report for 192.168.20.10
Host is up (0.00012s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 08:00:27:67:7A:EF (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.30 seconds
```

HOST DISCOVERY OPTIONS (WAYS TO CHECK "ALIVE" MACHINES)

--dns-servers <serv1[,serv2],...>: Specify custom DNS servers

-n/-R: Never do DNS resolution/Always resolve [default: sometimes]

-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes

-Pn: Treat all provided hosts as online -- skip host discovery

-PO[protocol list]: IP Protocol Ping

-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports

-sL: List Scan - simply list targets to scan

-sn: Ping Scan - disable port scan

--system-dns: Use OS's DNS resolver

--traceroute: Trace hop path to each host

Block 4. Privacy and browsing security

Exercise L2B4_PRIVACIDAD: Blacklight: Website privacy inspector

Activity description / Practical application

You need to know how any internet website uses **data about how you browse** for its own benefit

Expected results

This activity will be completed when you are able to use *Blacklight* to inspect how any site you choose is collecting your data and analyze the results. You can answer these questions:

- *Have you detected trackers on the websites of public organizations (governments, universities, etc.)?*
- *How do typical internet stores behave?*
- *What about newspapers?*

Additional information required to do it

Blacklight (https://themarkup.org/blacklight) scans and reveals the specific user-tracking technologies of any site. This means that it reveals who is getting your data when browsing in a site and, therefore, who is going to use them to serve ads to you that are based (supposedly) in your preferences.

Exercise L2B4_NAVEGAR: Create a secure and truly private browsing environment on your VM

Activity description / Practical application

You can create a safer internet browsing environment for your daily life

Expected results

This activity will be completed when your secure browser is operational and able to browse to webpages with all the mentioned plugins active. You can also answer these questions:

- *Do you think it's appropriate to mount all these security measures on a virtual machine that you use only for browsing?*
- *Do you think it's a good idea to combine all of this (virtual machine included) with the NextDNS from the previous lab for security?*

Additional information required to do it

*(**NOTE**: Secure does not mean private. Your connection attempts will be still logged by the destination servers, and your IP will be still revealed to them. To achieve a certain degree of browsing privacy you need other tools (like ToR), but this will be reviewed in a future session 😊)*

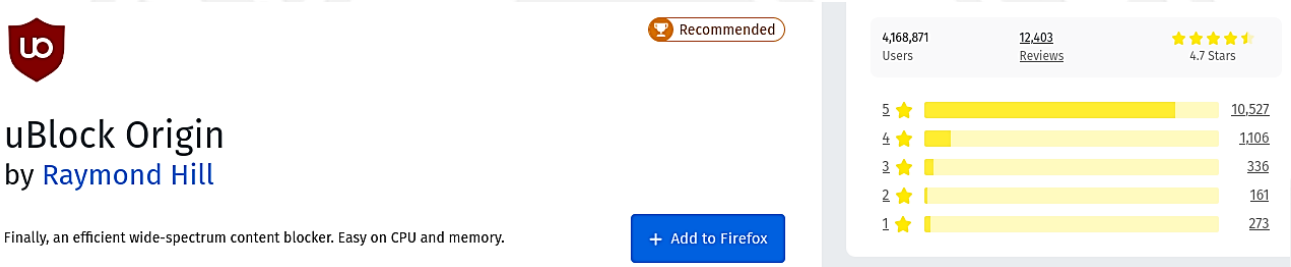
The purpose of this section is to fight against the techniques you have discovered with *Blacklight*, as the amount of data collected by a website may be troublesome in certain usage scenarios. Apart from that, it is very handy to have a more secure browsing environment just in case you accidentally browse a malicious or trojanized web page, pages with malicious ads, and other web threats. This way, you can browse on a safer way **if you use the following browser plugins**.

🔍 ***Combining it with a "VM to browse through Internet", you obtain a more secure browsing environment, especially if it is a Linux system.***

We will use *Firefox* as a browser example, but the same plugins are available in other browsers too (also on mobile versions). This also connects with the introduction topic of the theory.

To deal with browser extensions, you need to consider the following things:

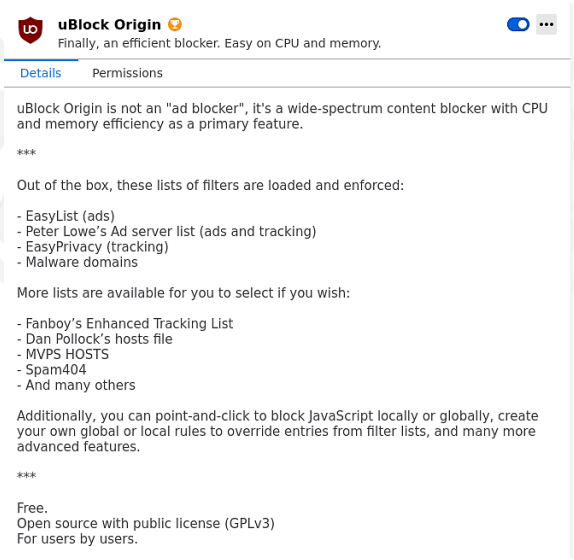
- **There are malicious extensions**: to prevent problems, install only the *Recommended* ones (inspected by the browser vendor), with high score, and lots of user recommendations.



- Be wary of extensions asking for too many or unreasonable permissions.
- Extensions are normally installed using a specific browser option (*Add-ons*) or a market (*Google Chrome Store*) not from an external web.

Once this is said, it is recommended to install extensions that deal with:

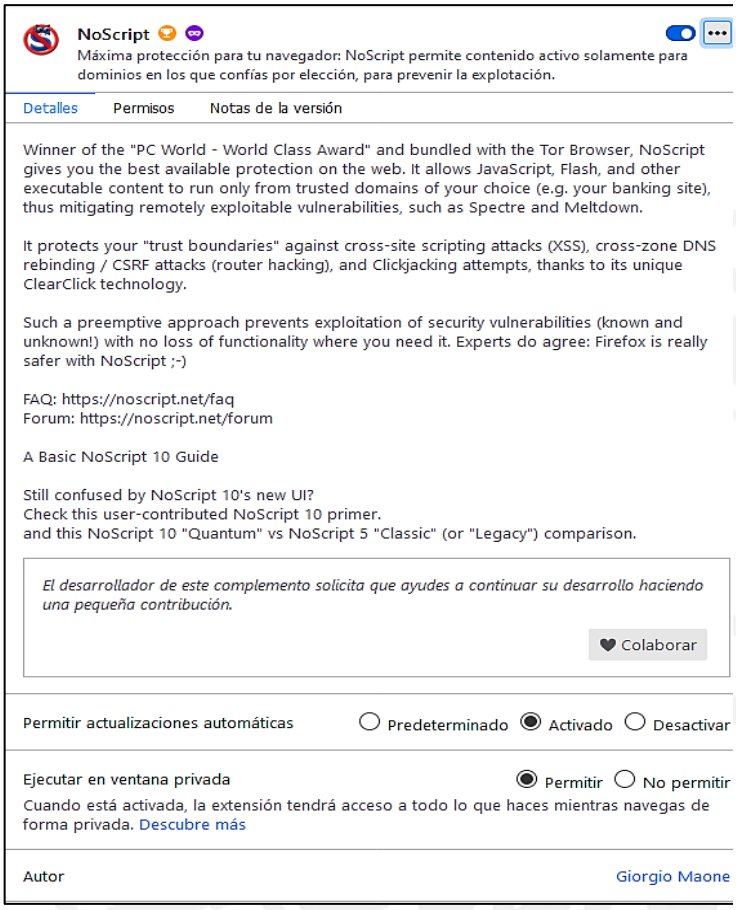
- **Ads**: as they can be obtrusive and/or malicious, even being a legal income source for page owners. ***uBlock Origin*** is the recommended add-on.

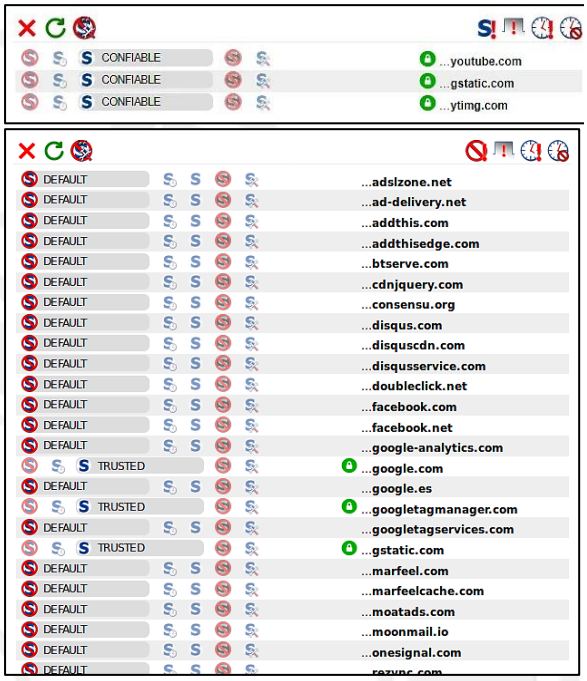


- **Script blockers**: Using *JavaScript* in webpages is quite common and legitimate, but sometimes scripting is also used for nefarious purposes. To prevent this, the ***NoScript*** extension is recommended. It not only blocks scripts, but also identifies malicious scripting usages in web pages.

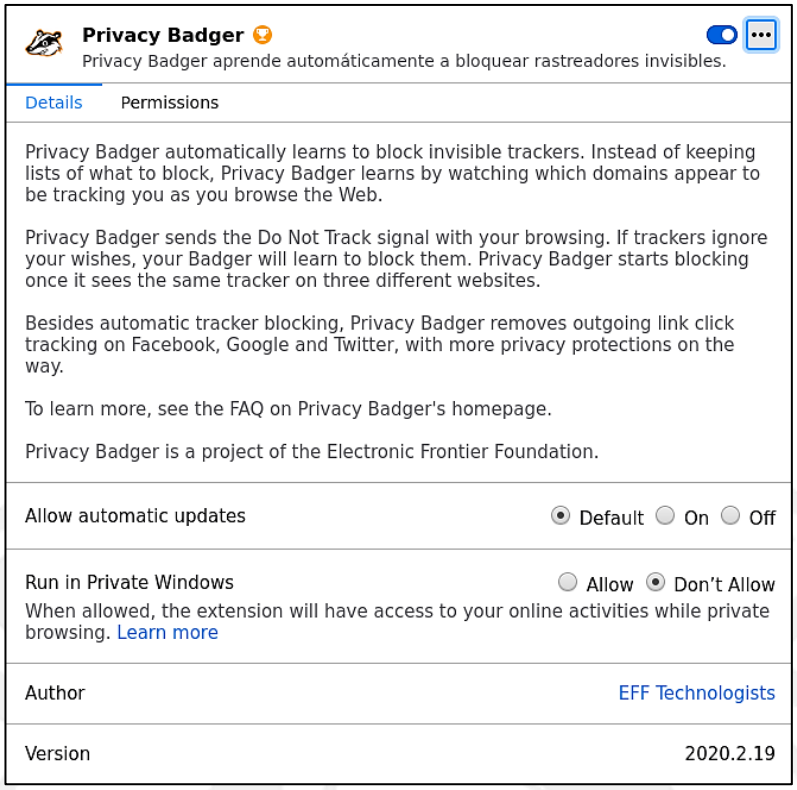
However, using this requires "training", as it initially blocks most scripts on a web page rendering it unusable (or making their contents invisible) most of the time.

🔍 ***You can unlock script sources until the page is usable enough for your purposes. Do not unlock obvious ads or malicious domains! Normally you should start with script sources that are on the same DNS name than the page you are viewing. This kind of training is stored between sessions, so you only need to do this once per webpage.***





- **Privacy**: ***Privacy Badger*** is the recommended extension that prevents webpages to track your browsing history and create a profile of your browsing behaviors. It also blocks popular social networks, website tracking attempts (*Facebook*, *Google*, *Twitter*…), and implement other measures protecting user privacy.



Exercise L2B4_PWDFILTRADAS: Have I been pwnd?

Activity description / Practical application

You need to know if the password of any of your accounts in an internet service **has been potentially leaked** through one of the many data leaks that are produced nowadays

Expected results

This activity will be completed when you inspect any email account you want to know if it is part of a known data leak. You can also think about possible uses of this tool that violate privacy. For example, *what do you think it could happen if you put in the email of someone you know, and it turns out that they've leaked on sites like Tinder?*

Additional information required to do it

Have I been pwnd? (https://haveibeenpwned.com/) is a website that **stores and allows to query user data leaks**. It accepts email account names, and it tells if this account name has been leaked in any of the known user databases that have been stolen. Therefore, if the requested email account is inside one of the multiple stolen user databases it manages, it tells you the site whose users have been stolen, when, and other details about the data leak.

🔍 ***Due to the probability of using the same password and email in multiple sites, it is especially important to verify this to prevent that a leak in one site facilitate intrusions in other sites.***

✍️ Badges and Self-Assessment

NOTE: You have a version of this badge table in an editable format available in the *Virtual Campus*. You can use this file to easily create a document in the format you want and take extended notes of your activities to create a log of what you did. Remember that this material can be brought to lab exams.

Badge Level	Unlocked when	Unlocked?
	You can find and analyze any `robots.txt` file	
	Answer this question: *What do you think it will be the security problems that a badly written* `robots.txt` *could provoke?*	
	You can analyze the metadata of any individual file of an appropriate type you find. You can answer this question: *Why metadata can be a security threat?*	
	You can analyze a single URL or IP using *Shodan* and interpret the results. You can also answer this question: *what part of the results links with something that we saw in the previous lab, and may indicate a severe problem with the webpage?*	
	You can search all the historic URLs of a domain.	
	You can answer this question: *what security problems may happen if we have an historic archive of the different contents of a* `robots.txt` *file?*	
	You can answer this question: *why subdomains are interesting from a security point of view? What could happen if a subdomain (and its contents) has been forgotten for a long time?*	
	You can analyze how a website collect user data and deals with privacy using *Blacklight*.	
	You can check if a password has been potentially compromised in a data leak and answer this question: *what should you do if you find your email in one of these documented intrusions?*	
	You can use *Censys* to analyze the hosts of a network and interpret the results, both from the network and from any individual hosts. You can also answer this question: *are you able to find services running in the hosts? How can you find if some are vulnerable?*	
	You know how to operate the *Google Hacking Database* against a concrete target.	
	Answer this question: *What kind of security problem do you think finding past content could suppose to a website?*	
	You can locate subdomains of any domain using one of the provided tools and analyze the domain information they provide.	
	You can locate the IP networks of any *.com* or *.es* domain.	

	You can geolocate any IP and answer the following question: *Do you think this geolocation is accurate?*	
	You can locate active hosts in LAN networks.	
	You can setup a private and secure browsing environment	
	You can answer this question: *What happens if instead of an email of yours, you decide to investigate emails from other people? What happens if the email is reported as compromised?*	
	You can answer this question: *If I am able to find any image or document that is (or was) in any website, what can we do with these files that may reveal me more information? What happens if I can find the history of versions of any of these files? What kind of information it may provide?*	
	You can answer this question: *If I am able to find any version of any web page of that once was part of a website, what can we do with these files that may reveal me more information? Do you think this could allow locating potential vulnerabilities?*	
	You can answer this question: Imagine that I leave by mistake a compromising comment in a webpage (indicating a product name and/or version, part of the server code in a comment…). *What should we do to really eliminate the security problem this could suppose?*	
	You can answer this question: It is a fact that UDP scans are slow, especially in WANs / Internet. It is also known that `nmap` usually scans by default the topmost used ports worldwide (a finite list of known ports). Knowing this, if you want to create an UDP "hidden" service, which is difficult to locate, *what would you do?*	
	**I have seen things you people would not believe**: Use multiple enumeration and OSINT techniques to extract lots of information about machines, companies, and persons that you can use to analyze them.	