GOBIERNO DE ESPAÑA
VICEPRESIDENCIA TERCERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Webinar: Secure development

Practical and research exercises

017 TU AYUDA EN CIBERSEGURIDAD
incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

INDEX

1. EXERCISE 1 3
2. EXERCISE 2 4

FIGURE INDEX

Figure 1 Wappalyzer 4
Figure 2 BuiltWith Technology Profiler 4
Figure 3 Retire.js 5
Figure 4 CMSmap 5
Figure 5 WPScan 6
Figure 6 Linkclump 6
Figure 7 Dirbuster 7
Figure 8 Burp Suite 7

1. EXERCISE 1

Sanitization of input variables is critical as we saw in the webinar. In this case we present the "sanitize" library that will allow us to do this work for HTML and directories in GO.

https://github.com/kennygrant/sanitize

We will be able to do the sanitization of the following functions:

Accents: replaces a set of accented characters with their ASCII equivalent.

```
sanitize.Accents(s string) string
```

Filenames: normalizes the path and name of files securely for use.

```
sanitize.BaseName(s string) string
```

HTML entity replacement: escapes "<" and ">" characters to avoid being interpreted as HTML code by browsers.

```
sanitize.HTML(s string) string
```

Replacement of HTML entities with filter: performs the same process as the previous one, but leaving as code the tags indicated in the function parameters.

```
sanitize.HTMLAllowing(s string, args...[]string) (string, error)
```

Path: processes the URL to make it secure.

```
sanitize.Path(s string) string
```

As we have seen before, thanks to this library and the previous examples, we can secure the inputs.

2. EXERCISE 2

In this research exercise we will see the most used tools to identify the directories of an application, how it is built and, finally, how a validation of entries would be done in order to obtain errors in the web application. This kind of techniques are also used for the identification of vulnerabilities and also in a pentesting.

Obtaining information is fundamental when looking for errors that are not shallow. Some tools that will help us quickly and as an extension of Chrome are:

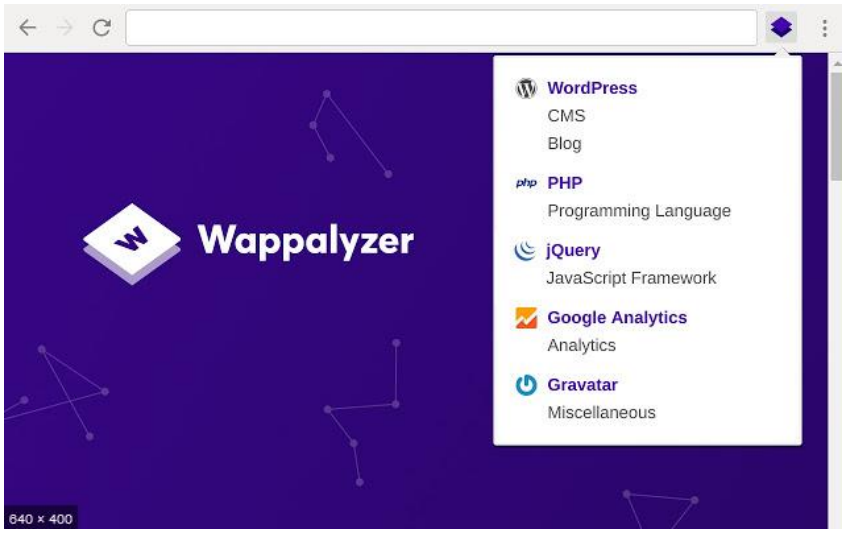

Figure 1 Wappalyzer

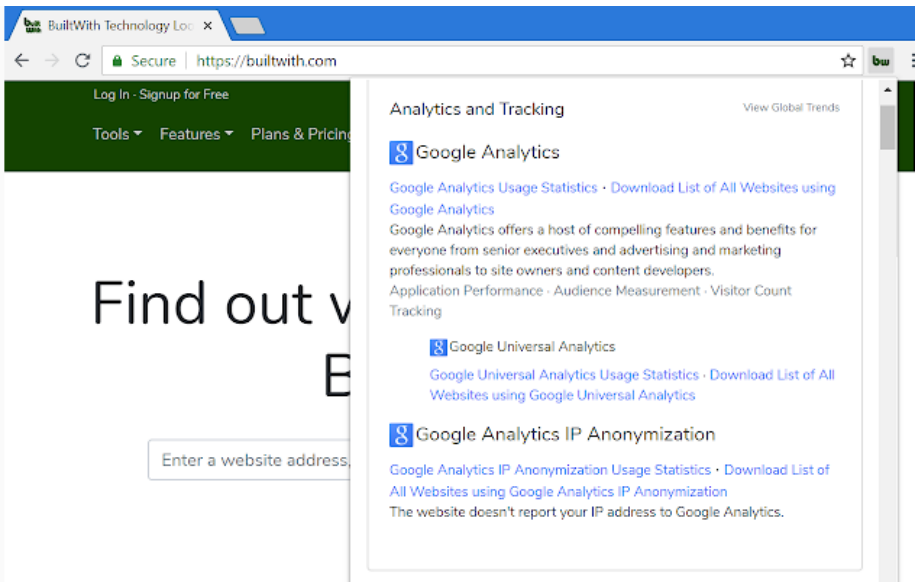

Figure 2 BuiltWith Technology Profiler

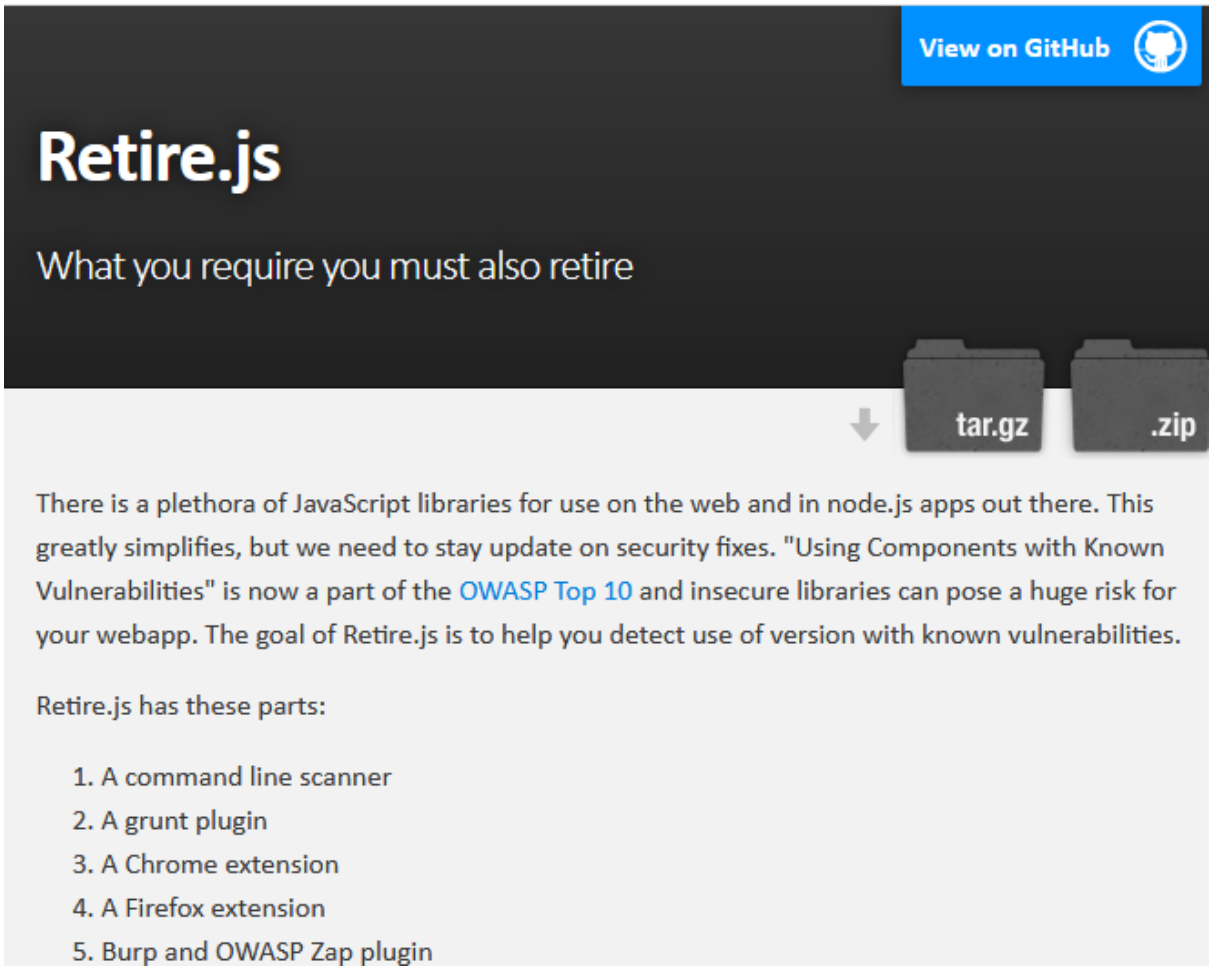

Figure 3 Retire.js

The above extensions are just some of the methods you can use to obtain information and perform a correct profiling of the web application.

If a CMS (Content Management System) is obtained when identifying the web application, specific tools can be used:

Figure 4 CMSmap

Figure 5 WPScan

Many times it is an art to identify the paths where the applications are running and especially their directories. To do this, you can use the Linkclump tool to facilitate this task.

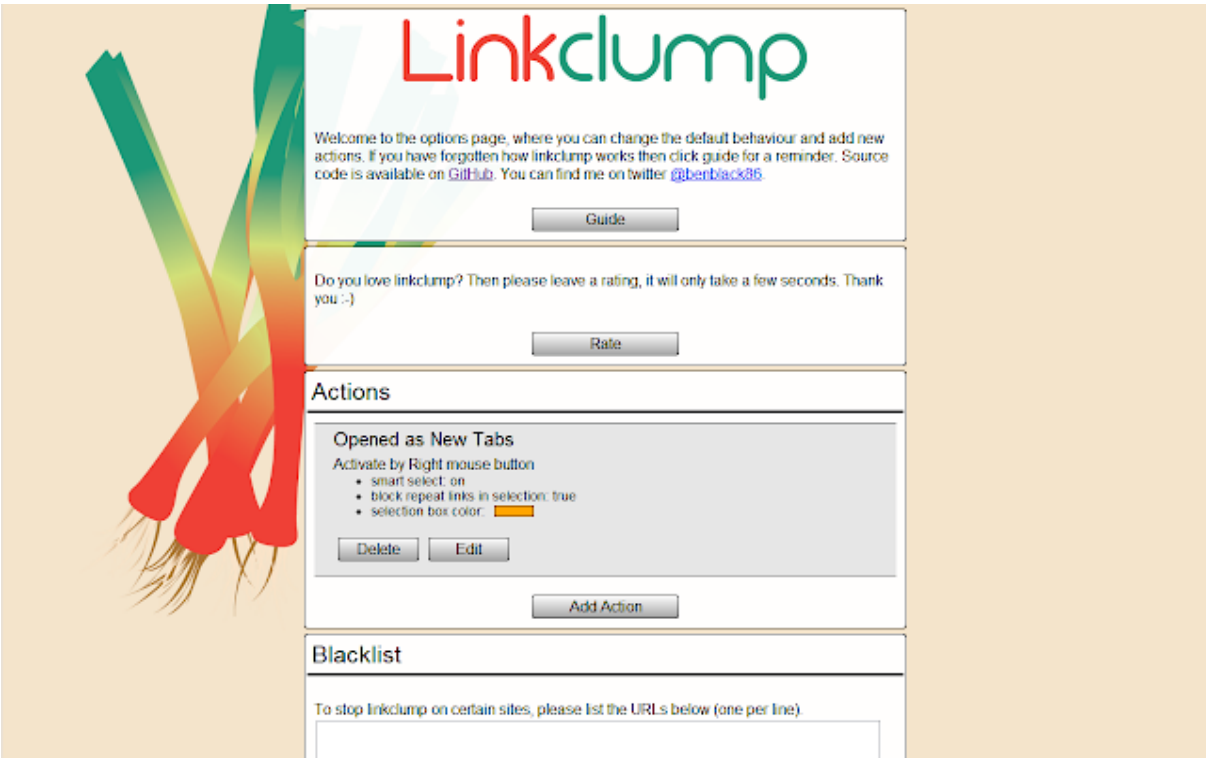

Figure 6 Linkclump

Another type of tool that the community uses to discover directories would be Dirbuster.

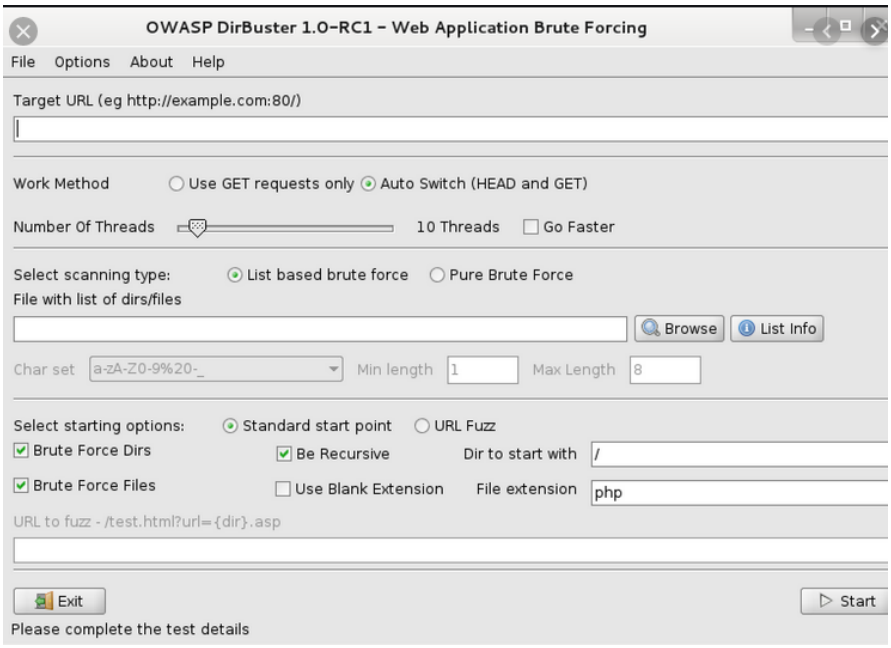

Figure 7 Dirbuster

Finally, when we have correctly identified the application and the directory of the application, we could analyze the input variables to it by POST or GET.

Thanks to BurpSuite and its Intruder module we could perform this kind of actions:

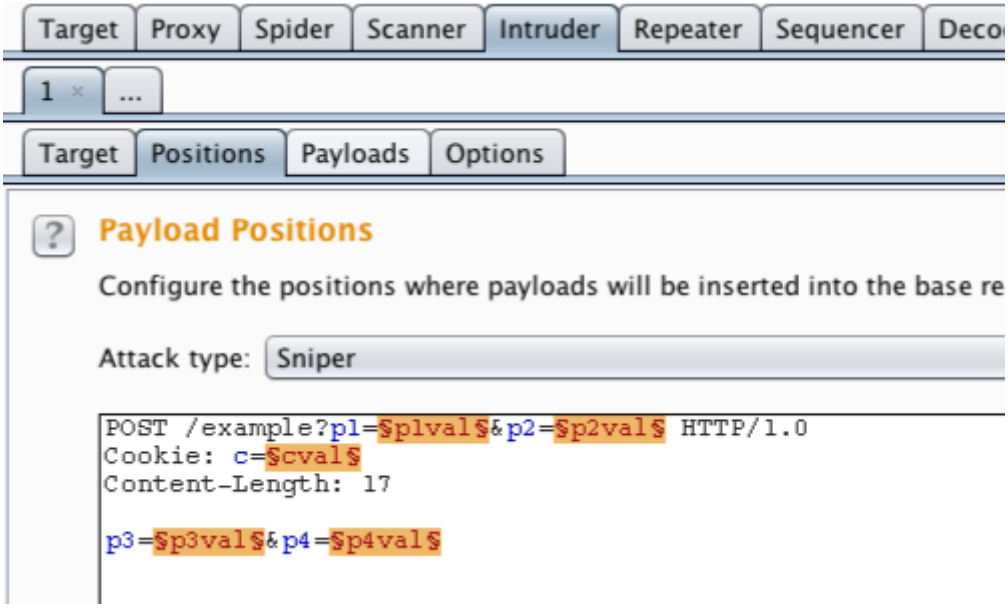

Figure 8 Burp Suite

Within the following link you will find a small manual on how to proceed to use the Intruder module, as well as a website for testing. http://testphp.vulnweb.com/login.php