What is Kali?

It's a fact that these last years the Backtrack distribution has been the most used by security professionals and enthusiasts. Its path started right in 2006 and for seven years it was improved while gaining its place in the security community. Therefore, nowadays it's hard to find someone interested in computer security that has not listened about Backtrack.

In March 2013 the Offensive Security people went one step forward and published the definitive Backtrack evolution. His name: Kali.

Coming from a team called Offensive Security, even if they deny it, what an appropriate name is Kali! The Hindu goddess of time, change and destruction or perhaps because the Philippine martial art… Pretty offensive, isn't it? Leaving aside its name, we can assure that Kali is a powerful tool that any security professional can use for free.

The Good and The Bad

Trying to list the possible drawbacks of Kali is a hard task, so we will start enumerating several of its advantages.

Talking about Kali advantages is talking about the changes between Backtrack and Kali. We will suppose that the reader already knows what Backtrack is and which their capabilities are.

Briefly, for those who don't know anything about Backtrack, Backtrack is a Linux distribution, based on Ubuntu, with plenty of security tools cleverly classified and ready to use.

Then, why Backtrack had to evolve into Kali? These are some of the changes in Kali and, therefore, some of its advantages.

Kali is based on Debian

This implies many advantages. The first of all is that the repositories are synchronized with the Debian repositories so you can easily obtain security patches and repository updates. Maintaining your pentesting system updated is a key feature.

Another advantage is that every tool in Kali is compliant with the Debian packaging policy. This may seem trivial but eventually will assure more robustness and clarity to the overall system structure and tools, also giving you an easy way to obtain the tools source codes to review or modify them.

Architecture compatibility

A key feature in Kali is its improved ARM compatibility. Since Kali appeared, many impressive builds have been created. What do you think about building Kali on a Raspberry Pi or on a Samsung Galaxy Note? Pretty amazing, don't you think?

Advanced wireless support

One of the focuses of Kali developers has been to support a broad number of wireless devices being them internal hardware or USB dongles. This effort goes in conjunction with the implementation of a patched custom kernel including all the new patches focused in the injection of data through network interfaces. A main requirement when a security professional has to perform a Wireless Assessment.

Fully customization
Kali is very flexible when it comes to visual interface or system customization. As for visual interface, now the user has the capability to choose several desktops such as Gnome, KDE or XFCE, among others. Regarding system customization, now you are able to easily create ISO images fully customized thanks to the Debian live-build scripts.

Business aware
All the Kali customizations and the Debian stuff mentioned earlier give the capability to companies to deploy Kali in multiple systems and to perform network Kali installs from local or remote repositories.

Easy upgrades among future versions
This is a key feature for every system administrator who has to maintain Kali systems or, actually, for anyone using Kali. With Backtrack, for any new version of Backtrack one had to completely reinstall the system. Now, with Kali, thanks to the move to Debian kali offers an easy way to upgrade your system when new versions come out.

Great documentation
Another important thing to remark is that with Kali you have a lot of online documentation in order to guide you in all your tasks.

As you can see, Kali is not only a new version of Backtrack but a full new infrastructure. And with this effort, a lot of new and powerful features have come.

Regarding the disadvantages, it's embarrassing to say that the writer has not found any relevant drawback of using Kali with security assessment purposes. As to the system architecture, the migration to Debian has brought a lot of powerful features. This combined with all the provided tools and the purpose of Kali developers to maintain them and provide the last updates as soon as possible makes Kali the best choice for anyone searching for a security distribution.

Included Tools

Kali puts more than 200 toolas at your disposal. If these tools were not well organized and classified, the usability of that prenetrating testing framework wouldn't be quite good. But as with Backtrack, all the tools are consistently classified by its category. We will know explain what each category is and what the most representative tools are.

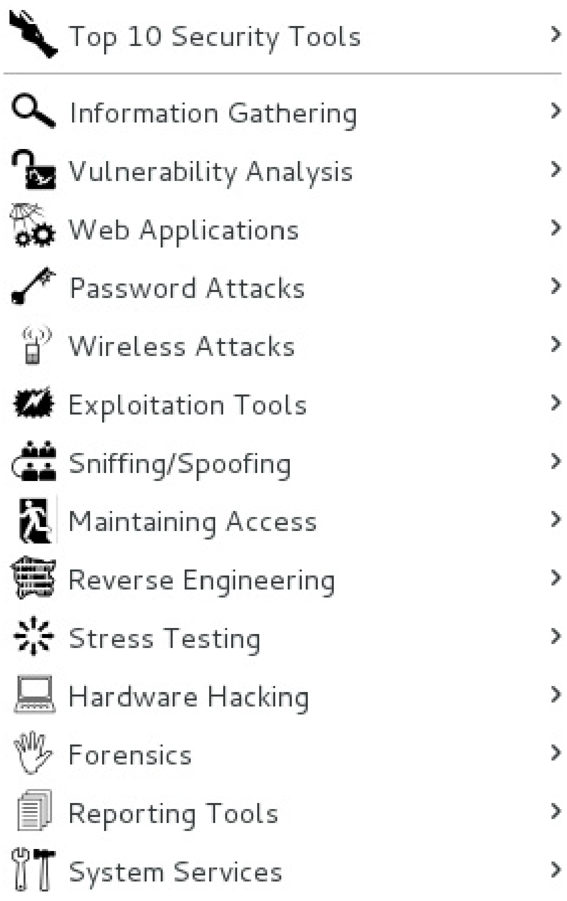


Figure 1. *Classification of tools*

The first category is *Information Gathering.* This category groups those tools focused in obtaining information about the target. Inside this category there is a huge amount of tools, each one divided by the kind of recognition that they do. For example, there is *OS Fingerprinting*, *Network Scanners*, *SSL Analysis*, *VoIP Analysis* and many more.

From all these tools, we can highlight the old known tool *Nmap* that is a really powerful network scanner. With Nmap, besides of being able to know which ports are open, filtered or closed, you are able to identify which services are behind them and also perform operating system recognition. Furthermore, with the new versions you have the possibility to program scripts that will be added through its Nmap Scripting Engine (NSE) functionality. As of today, in the official Nmap site you can find more than 400 scripts that give Nmap even more power than ever.

Another tool worth mentioning is `theHarvester`. This tool uses many search engines such as *google, google-profiles, bing, linkedin or shodan* to find information about, for example, a company. You can find email addresses, host names and much more information pertaining to that company.

The next category is *Vulnerability Analysis.* This one is focused in discovering vulnerabilities, so here you have tools such as vulnerability scanners or fuzzers. In this category you can find *sqlmap.* This is a great tool that really can help you finding and exploit SQL injection vulnerabilities. With this tool, you specify the web application and the parameters you want to check and then it sends a huge battery of tests. This tool is amazing and eases the repetitive task of testing all the payloads for a great number of database engines. Another important tool is *OpenVAS.* OpenVAS is a complete framework focused in the discovery of vulnerabilities. It was born as a fork of Nessus when this became non-free source.

In the *Web Applications* category you can find tools that identify web applications and their vulnerabilities. To this end you have at your disposal tools such as *Burp Suite*. One of the main and basic features of Burp is its capacity to intercept all the requests sent to web applications so you can modify and resend them. But Burp is not only an intercepting tool, it is one of the best tools to perform web application analyses being them automatic or manual. For example, with Burp you will be able to load many payloads from a file and modify the parameters sent to the web application with that payload. This can allow you to perform brute force attacks against authentication forms, load customized payload to find SQL injection or cross-site scripting attack vectors. Its UI is pretty intuitive so any user will become familiar with all the features. You also have tools such as *XSSer* that in a similar way to Sqlmap launches a bunch of payloads to find cross-site scripting vulnerabilities.

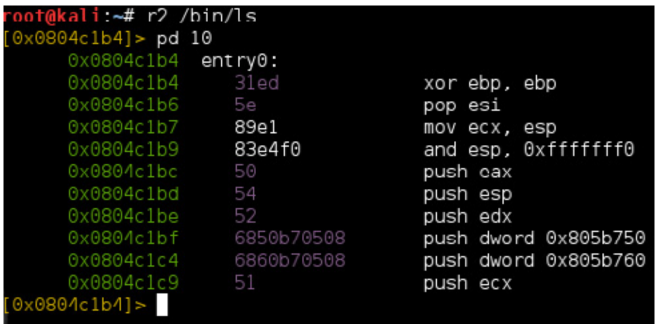
Then you have the *Password Attacks* category. This category is quite self-explanatory. You can find tools that crack passwords offline or launch attacks to online services. Remarkable tools are *John the Ripper, oclhashcat-plus, medusa and THC-Hydra*. The first one is an old but well maintained password cracking tool. One of the main features of the second tool is that you can use the power of GPUs to perform attacks on passwords and, finally, with medusa and THC-Hydra you will be able to launch brute force attacks against online services. THC-Hydra made a clear statement of intents (*http://www.thc.org/thc-hydra/network_password_cracker_comparison.html*) comparing its features against other tools such as medusa. And in that comparison, THC-Hydra is clearly the winner.

The next category is *Wireless Attacks.* In this section you can find tools to analyze and attack wireless protocols such as IEEE 802.11, RFID/NFC or Bluetooth. The quintessential tool to perform analyses of the IEEE 802.11 (WiFi) protocol is `aircrack-ng`. This tool is a complete framework that allows you to perform many attacks against the different authentication and authorization mechanisms of WiFi networks.

In the *Exploitation Tools* category you can find different tools that are designed to attack different kinds of systems or attack systems in different ways. One of the best tools that we have nowadays in order to exploit the vulnerabilities present in a system is *metasploit*. Metasploit is a complete framework that has a huge number of exploits ready to be launched against the objective. It is, more or less, a click and shoot tool that gives you everything built so you don't have to worry about the technicalities of the vulnerability being exploited. You also have another interesting tool, *SET*. The Social Engineering Tool is another framework that will help you to take control of systems but using social engineering to achieve your goal. For example, with this tool you will be able to easily build phishing web sites so you can deceive users and make them install malicious software such as PDF files with malware in them that will be also provided by SET.

The *Sniffing/Spoofing* category is used to store those tools used to intercept network, web or VoIP traffic. One of the best sniffers out there is *Wireshark*. With wireshark you will be able to intercept network traffic and the same tool will, where possible, identify the protocol used and highlight the important data. You will also be able to apply advanced filters to the data being intercepted once it is intercepted or while it's being intercepted. Another interesting tool is *dsniff*. This tool is a complete framework divided in many applications that will let you intercept and identify interesting data such as passwords and e-mails or sniff encrypted SSL data by exploiting weak configurations.

The following category is *Maintaining Access.* This category unifies all those tools that will help you to maintain access to the target and get the critical information stored in it. For example you have many operating system and web backdoors as well as different tools to encapsulate the outgoing traffic in protocols that are not normally filtered. For example, you have another old known tool called *netcat* (ncat). Netcat is a really flexible tool that allows you to perform client-server communication. Netcat is a basic tool that depending on your imagination can become a tool you will use every day for the many different things such as rapidly retrieving HTTP banners, transferring files from one machine to another and many more things.



```
root@kali:~# r2 /bin/ls
[0x0804c1b4]> pd 10
        0x0804c1b4   entry0:
        0x0804c1b4     31ed                xor ebp, ebp
        0x0804c1b6     5e                  pop esi
        0x0804c1b7     89e1                mov ecx, esp
        0x0804c1b9     83e4f0              and esp, 0xfffffff0
        0x0804c1bc     50                  push eax
        0x0804c1bd     54                  push esp
        0x0804c1be     52                  push edx
        0x0804c1bf     6850b70508          push dword 0x805b750
        0x0804c1c4     6860b70508          push dword 0x805b760
        0x0804c1c9     51                  push ecx
[0x0804c1b4]>
```

Figure 2. *Netcat tool*

The *Reverse Engineering* section unifies all those tools with which you can debug or disassemble binaries. In order to debug binaries you have *ollydbg* or *edb-debugger*. The first tool is a quite powerful debugger but it has to be executed through wine, given that it's only available for Microsoft Windows systems. For this reason you have edb-debugger that despite being quite new is still useful. Then you have a complete framework for reverse engineering, *radare2* (r2). This is the Swiss army knife of every reverse engineer that works with a Unix system as a workstation. Radare2 is not an easy tool to use. It has a hard learning curve, but once you get it, it becomes a really powerful tool. The framework radare2 is formed by many small tools such as r2, rabin, rasm or rax. Each one allows you to perform many different things. For example, with radare2 you will be able to inspect shellcodes, reverse engineer binaries from different platforms such as pe, elf, match0 and dex or java classes, analyze disk images to perform forensic analyses, find gadgets to build your ROP (Return Oriented Programming) payload, debug binaries, find differences between binaries (bindiffing) or patch binaries. All this can be extended with its capability to process plugins that you can program in Python, Go, Perl, Javascript, etc.

In the *Stress Testing* section you can find different tools to check the capacity of your network, web application, WLAN or VoIP service to handle huge amounts of traffic. For example, with these tools you will be able to simulate denial of service attacks.

With the tools found in the *Hardware Hacking* category you will be able to program sketches for Arduino devices and you will also find different tools to develop for Android – with the Android SDK – and analyze Android applications with tools such as *APKTool* and *dex2jar*.

In kali, the *Forensic* category is amazing. There you have plenty of tools focused in several forensic fields. For example, you can find tools to carry out network forensics, PDF forensics, RAM forensics and much more. One tool that is hitting hard these days is *volatility*. This tool is used to analyze data stored in RAM. You can give volatility an image of the RAM in a given point and volatility will extract for you a lot of interesting information. For example, you can extract all the running processes in the moment, opened sockets and network connections, LM/NTLM hashes and LSA secrets and a lot of other information. If you want to start playing with it, the volatility people provides (*http://code.google.com/p/volatility/wiki/PublicMemoryImages*) you many prepared RAM images with interesting data you can extract.

Finally, you have the last two categories, *Reporting Tools* and *System Services*. In the reporting tools section, as its name suggests, you can find tools to help you when reporting all the vulnerabilities you have found. For example, you have *recordmydesktop*, that it's simply a tool to create videos from your activities in your computer. Another important tool is *truecrypt*. Even it's not directly related with the documentation task, as a security professional you always have to be really careful with where you store the results of your work. Truecrypt gives you the possibility to securely store your pentesting results and save them encrypted so nobody but you can read them.

In the system services category you have different services you can launch to web you with your tasks. For example, you can launch an Apache HTTP server or a MySQL server.

As you have seen, Kali has everything a pentester might need. And thanks to its flexibility, if the tool of your need is not in your arsenal, you still can easily install it in your distribution.

Web application pentesting process

A penetration test it's usually divided in two kinds of tasks. Those tasks that are automatic and those that are manual. The manual tasks are the ones that add value to your reports as a pentester, and are guided by your experience and intuition. These manual tasks are the ones that will make of your report something amazing and beautiful or the conversely, something that seems factory created. If you don't go hard with these manual tasks you will end up with a report that anyone could produce given that all your findings will be discovered by automatic tools and, therefore, any one clicking a button will be able to get the same results.

Another important issue for a pentester is the time factor. When you are hired to perform a penetration test, your time will be limited and, normally, the final outcome of the analysis will be directly related with the time available to carry the analysis out. Thus, it is of great importance to be organized and have some kind of methodology. In this chapter we will explain many steps for carrying out a penetration test for a web application but we will suppose that you will work alone. If you were part of a team, the methodology explained here would not be enough. We will not take into account the way you would be communicating with the team in order to share your results so they could use them in their tasks.

In this chapter we won't cover in depth what kind of vulnerabilities you should search for. Instead we will explain a methodical way that will allow you to easily find them once you know what to look for.

If you are carrying out a complete penetration test against, for example, the network of a company, we'll suggest you to follow the OSSTMM (*http://www.isecom.org/research/osstmm.html*) (Open Source Security Testing Methodology Manual) methodology. This is an open source methodology so you don't have to pay in order to get and follow it. This methodology, broadly speaking, is a guide about when and what elements should be tested during a penetration test.

Talking about web application security, if you want something more specific that also allows you to know what kind of vulnerabilities you have to search and what these vulnerabilities are, we highly recommend you reading the "OWASP Testing Guide" (*https://www.owasp.org/index.php/OWASP_Testing_Guide_v3_Table_of_Contents*).

From its website, OWASP (Open Web Application Security Project) is "an open community dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted". In order to accomplish their objectives, OWASP have developed many different methodologies to many different problems such as pentesting web applications, reviewing source code, etc.

The OWASP Testing Guide is a great resource to know what kind of things you should look for when conducting a web application penetration test. They explain all kind of vulnerabilities you may find, what they are and how you can find them. Of course, the explanations are not in depth, but once read, you will have a good standpoint to go on and search more information about the vulnerabilities and how to exploit them.

Ok, then. Now that the concepts that we will explain, are clear, let's explain them. As we said, the manual tasks in a penetration test are the key to obtain a good outcome of your work, but given that we have a limited time, we will need to somewhat relay in some automatic tools that will work in the background while we manually check our target.

So the first step is to prepare those tools that will perform automatic scans. Here you have many possibilities. Some of them will be in the Kali distribution and some will not. If you are a security professional, meaning that you have some monetary gain from your work, we would advise you to buy one or more than one automatic scanners. Here we list some tools that can have the work done for a reasonable price. It's worth noting that the tools listed below are more focused in system vulnerabilities, but they can also detect web vulnerabilities.

- Nessus Vulnerability Scanner
- QualysGuard

Nessus is a tool that you will have to install in your local machine. On the contrary, QualysGuard is a service in the cloud. In our experience, QualysGuard makes a pretty good job and, comparing it with Nessus, QualysGuard probably gives less false positives. Both tools can produce reports as XML files so you can easily integrate their results to your reports.

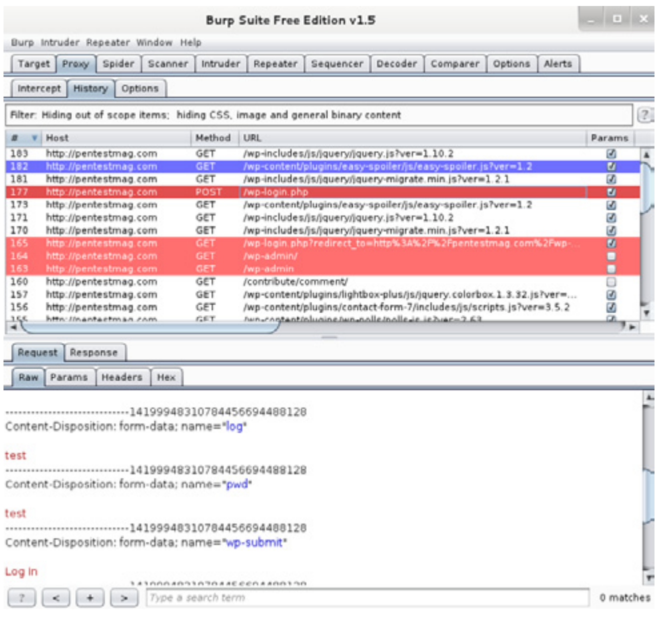
What we strongly advise is that you install more than one automatic scanner so you can compare their results, and what is of critical importance… ¡You must always check the vulnerabilities reported by them!

If you don't have the resources or don't really want to buy any tool, you can always use w3af or the new tool Arachni Web Scanner from your Kali distribution to perform web application scans. W3af is a great tool and it's already a pretty mature project.

Now that you have launched your automatic web scanners – hopefully from a different machine so we have plenty of memory/CPU resources-, we can start the manual part.

The following steps are completely related to a web application penetration test. Nowadays, web applications are a critical component in the business model of enterprises. The fact that a lot of web applications are specifically developed from the scratch to fulfill the company needs and that web vulnerabilities are easier to exploit than system vulnerabilities makes that attackers focus their efforts in exploiting them. This means that as pentesters we should also focus our efforts in securing them.

The first step we should take would be to launch Burp Suite. With Burp Suite you will configure a localhost proxy in your browser so all the requests go through Burp. Then you configure the scope in Burp settings so you only log those requests in which the destination address is your target. This way you will prevent a lot of trash requests from showing up.


Burp Suite Free Edition v1.5
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Options Alerts
Intercept History Options
Request Response
Raw Params Headers Hex
Type a search term
0 matches

```
Filter: Hiding out of scope items;  hiding CSS, image and general binary content

#    Host                     Method  URL                                                              Params
183  http://pentestmag.com    GET     /wp-includes/js/jquery/jquery.js?ver=1.10.2                        ✓
182  http://pentestmag.com    GET     /wp-content/plugins/easy-spoiler/js/easy-spoiler.js?ver=1.2         ✓
181  http://pentestmag.com    GET     /wp-includes/js/jquery/jquery-migrate.min.js?ver=1.2.1              ✓
177  http://pentestmag.com    POST    /wp-login.php                                                      ✓
173  http://pentestmag.com    GET     /wp-content/plugins/easy-spoiler/js/easy-spoiler.js?ver=1.2         ✓
171  http://pentestmag.com    GET     /wp-includes/js/jquery/jquery.js?ver=1.10.2                        ✓
170  http://pentestmag.com    GET     /wp-includes/js/jquery/jquery-migrate.min.js?ver=1.2.1              ✓
165  http://pentestmag.com    GET     /wp-login.php?redirect_to=http%3A%2F%2Fpentestmag.com%2Fwp-...      ✓
164  http://pentestmag.com    GET     /wp-admin/
163  http://pentestmag.com    GET     /wp-admin
160  http://pentestmag.com    GET     /contribute/comment/
157  http://pentestmag.com    GET     /wp-content/plugins/lightbox-plus/js/jquery.colorbox.1.3.32.js?ver=...  ✓
156  http://pentestmag.com    GET     /wp-content/plugins/contact-form-7/includes/js/scripts.js?ver=3.5.2     ✓
```

```
-----------------------------14199948310784456694488128
Content-Disposition: form-data; name="log"

test
-----------------------------14199948310784456694488128
Content-Disposition: form-data; name="pwd"

test
-----------------------------14199948310784456694488128
Content-Disposition: form-data; name="wp-submit"

Log In
```

Figure 3. *Requests*

The next step would be to navigate through the entire target website while watching the request history page in Burp. This way once you find an interesting web page in your target, you will know which request and response has been sent and received and you will be able to highlight or even comment it in Burp. Let's define what may be interesting.

- Requests with GET parameters
- Requests with POST parameters
- Requests setting cookies (Set-Cookie header)
- And everything your intuition tells you that might be important

You will end up with something similar to the next image: Figure 3.

At this point you will have visited the entire web site and will have an idea of where the critical sections are.

The third step would be to analyze all (or the most important) requests that might be important. Now is when you experience comes into play. You will have to apply all your knowledge to identify all kind of vulnerabilities. For example, if you are in front a request that sends POST parameters, that parameters will probably be used in a database query, so you will have to try different things to check if a SQL injection vulnerability exists.

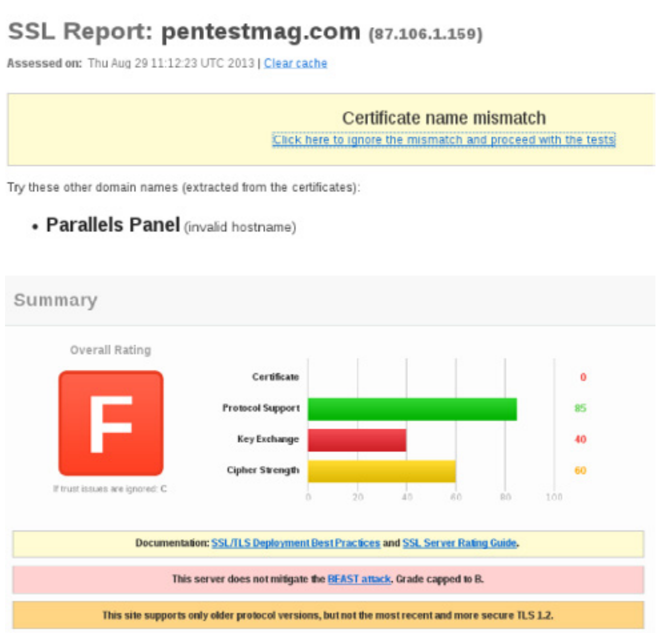
If you are certain that those parameters hit a database or might be vulnerable to a cross-site scripting vulnerability, you might want to combine your personal skills with the aid that tools such as Sqlmap or XSSer can provide. This way you will have all the tedious work done by automatic tools and you will be able to carry out the genius work.

You should proceed this way until you have checked all the important requests.

The next step would be to identify software weaknesses. Nowadays, the use of CMS is something really very widespread, so tools that help you identify if a CMS is being used would help. You can use, for example, WPScan in case you think that the CMS used is WordPress. This tool will help you identify the version of WordPress used and if it has any plugins installed so you can check if the WordPress or the plugin versions in use are outdated and, consequently, have vulnerabilities published.

Finally, you will try to find configuration weaknesses. Here you should look for things such as outdated server software in use, bad SSL configurations, etc. With Qualys SSLLabs you will be able to obtain a good colored graphic showing you the strengths and weaknesses of the target website SSL certificates as shown in the next image: Figure 4.

As you can see in the first image, the target website has an SSL certificate that does not have the correct domain associated with it. And in the second image you can see and overall score generated by SSLLabs that shows that the ciphers supported by the certificate in use are vulnerable to known waeknesses.



SSL Report: pentestmag.com (87.106.1.159)

Assessed on: Thu Aug 29 11:12:23 UTC 2013 | Clear cache

Certificate name mismatch
Click here to ignore the mismatch and proceed with the tests

Try these other domain names (extracted from the certificates):

- Parallels Panel (invalid hostname)

Summary

Overall Rating

F

If trust issues are ignored: C

Certificate	0
Protocol Support	85
Key Exchange	40
Cipher Strength	60

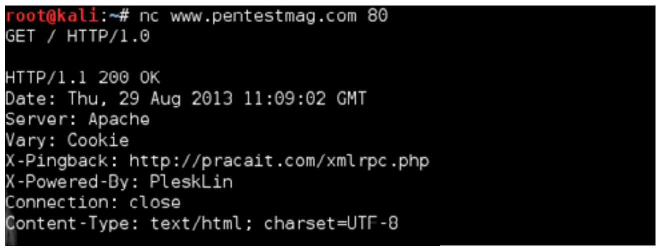
Documentation: SSL/TLS Deployment Best Practices and SSL Server Rating Guide.

This server does not mitigate the BEAST attack. Grade capped to B.

This site supports only older protocol versions, but not the most recent and more secure TLS 1.2.

Figure 4. *The strengths and weaknesses of the target website SSL certificates*

Needless to say that the same tasks that SSLLabs carry out, can be programmed by you with your own scripts so you don't have to depend on external services. On the other hand, one good approach to check server technologies is to check the HTTP headers. This can be done in different ways, for example, you could use Burp, but given that it is something really easy to check is as simple as sending a GET request with netcat as shown in the following image:



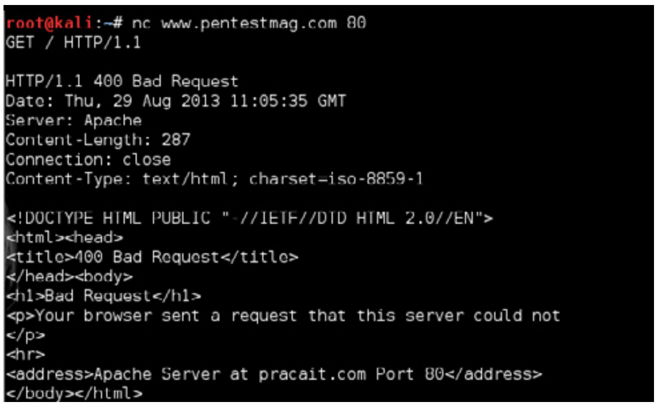
```
root@kali:~# nc www.pentestmag.com 80
GET / HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 29 Aug 2013 11:09:02 GMT
Server: Apache
Vary: Cookie
X-Pingback: http://pracait.com/xmlrpc.php
X-Powered-By: PleskLin
Connection: close
Content-Type: text/html; charset=UTF-8
```

Figure 5. *GET request*

As you can see, we can obtain some valuable information such that they are using Apache web server – although the Apache version is hidden – and thanks to the X-Powered-By header – that should have been removed – we can infer that they are using Parallels Plesk Planel. Then we can try to find published exploits for that software.

If it was not clear enough, sending a (malformed – without the host directive -) request using HTTP1.1 protocol will generate an error that will give you information about the hosting service in use. Look at the following image:



```
root@kali:~# nc www.pentestmag.com 80
GET / HTTP/1.1

HTTP/1.1 400 Bad Request
Date: Thu, 29 Aug 2013 11:05:35 GMT
Server: Apache
Content-Length: 287
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not
</p>
<hr>
<address>Apache Server at pracait.com Port 80</address>
</body></html>
```

Figure 6. *An error generated*

From the response of the server, you can infer that the website is hosted (or has something to do with) by the pracait.com service.

Once you have finished all the manual review of the web application, it will be the time to merge the results you have found with the results that the automatic scanners have found.

Explain how a good looking report should be done would be a subject to cover in another full article, so for this time, we will leave this out of the scope.

Following the all these steps you will be able to carry out a good penetration test. Of course, the results obtained will depend on your experience and the time you have to perform the task.

Here we only have outlined some of the steps and methodical techniques that will help you to optimize your time and efforts.

Learning from hacking

In the previous chapter we have mentioned many times that the resulting outcome of your work will depend, in part, on the experience and intuition you have.

It's hard to define where the line between experience and intuition is given that, in most cases, your supposed intuition really arises from past experiences. Therefore, the key to be a good pentester is his experience.

It is a usual practice that when companies are hiring security professionals, they always require them to have some prior proven experience, but, of course, if you have not worked before in the security field you will not have that experience. It is an infinite loop. What is the solution? Wargames!

Wargames are a kind of challenges made for you to learn and get fun at the same time. Of course, you only will get fun if you are geek enough! There are several kinds of wargames. You can divide them by their architecture but also by their subject.

Regarding its architecture, you have wargames that are thought to be downloaded to your local machines so you can overcome them offline. Other wargames are thought to be played online, therefore there are communities that provide online machines in which the challenge will be stored so you can access them. You can even download some wargames that are provided to you as virtual machines so you have a complete laboratory with all the tools you will need.

There are so many wargames that they might cover all the security field knowledge. You can find wargames to improve your system administration skills, about software exploitation, cryptography, protocol assessment and, of course, web application security. Given that this article has treated many web security concepts, we will talk about a wargame focused in web security.

Again, the amazing people from OWASP, provides us with a wonderful project. His name: OWASP Broken Web Applications Project.

With this wargame we will have to download a virtual machine that contains several vulnerable applications. These vulnerable applications are quite different between them.

On one hand, we have the training applications that are designed in a way in which any user can learn what web vulnerabilities are in a friendly way. In this category you have applications such as *Damn Vulnerable Web Application and OWASP WebGoat*. On the other hand, we have applications that have been left vulnerable on purpose, but they are more realistic than the ones presented in the previous paragraph. From this category we highlight a Google project called *Gruyere*. Gruyere addresses a lot of concepts related with web security such as the different topologies of XSS, CRSF, XSSI, Ajax vulnerabilities and much other stuff.

Finally, with the virtual machine comes a very interesting kind of vulnerable applications. They give us many real applications such as Wordpress or Joomla that are outdated, and given that its software is not updated they contain real vulnerabilities that have been found in the wild some time ago.

So with this wargame you can go from 0 to an acceptable level thanks to the all-in-one wargame and its incremental difficulty approach.

If you want to get a grasp of all the wargames out there, you can find a really good resource in this site: *http://www.amanhardikar.com/mindmaps/Practice.html*.

Farewell

I'm glad you are still here! As you can deduce, we only have scratched the surface of the covered topics.

No one will be outraged if you affirm that Kali distribution is the best security focused distro out there.

Regarding the pentesting point, I hope you could grasp the idea behind it. It's not an easy topic and it largely depends on your abilities, but being methodical is a big step to be successful.

For this reason, we encourage you to train yourself with the mentioned wargames, because unlike other careers, in the information security field you have plenty of opportunities to learn relevant subject by yourself.

Keep Hacking!

Bibliography

Chapter 1

- *http://en.wikipedia.org/wiki/BackTrack*
- *http://en.wikipedia.org/wiki/Kali_Linux*
- *http://en.wikipedia.org/wiki/Kali*
- *http://www.kali.org/about-us/*

Chapter 2

- *http://www.kali.org/news/kali-linux-whats-new/*
- *http://docs.kali.org/category/armel-armhf*
- *http://docs.kali.org/category/live-build*
- *http://docs.kali.org/network-install/kali-linux-network-pxe-install*
- *http://docs.kali.org/*

Chapter 3

- *http://nmap.org/nsedoc/index.html*
- *https://www.volatilesystems.com/default/volatility*

Chapter 4

- *https://www.owasp.org/index.php/About_OWASP*

Chapter 6

- *https://www.owasp.org/index.php/OWASP_Broken_Web_Applications_Project*

ALBERT LÓPEZ (NEWLOG)

Is a computer engineer that although finishing his degree the last year, has worked during two years developing system security software and making source code reviews in a company from Barcelona called Víntegris. Nowadays is gladly working for Internet Security Auditors as a security analyst where he has to perform security analyses to all kind of targets. However, his passion is exploiting software and everything related with low-level software development. For this reason, in 2009 he founded the Overflowed Minds community with the idea of spreading information of such an amazing subject.