|  |  |  |
| --- | --- | --- |
| Action | Commands (via CLI) | Description |
| Find IP and MAC address | ifconfig | Device network information |
| Scan network to identify connected devices | nmap -sT 192.168.99.0/24 | Show how we can scan a network – Can also point out that only devices on that network are being shown. |
| Ping the target device on the network | ping <ip.address> | We are sending ICMP packets to the target device – This will be visible on wireshark and highlight how we can identify ping traffic |

Session 1 – This will introduce you to some basic networking concepts and practices, such as different protocol uses find IP address and using nmap

Please note that throughout this worksheet anything in “<>” is to be treated as a placeholder and not typed, but replaced with the relevant value

Session2 – Here we will look at security issues caused by misconfigurations and how we can exploit and defend against these

When running any commands with “sudo” you may be prompted for a password – This will be the same password you logged in with. If you change your user password this changes the password for logging in and “sudo” usage

|  |  |  |
| --- | --- | --- |
| Action | Commands (via CLI) | Description |
| SSH onto target device | ssh kali@<ip.address> | Think about / consider what the password is likely to be… |
| Create file on desktop | mkdir /home/kali/Desktop/<dir\_name> | If the target device is being shown on a larger screen we should have visible proof of the action |
| The instructor will now terminate all SSH connections and change their default SSH port | - | This is to show case the importance and in some cases reliance on the use of default ports |
| SSH onto target device | ssh kali@<ip.address> | This should fail because SSH will assume that the default port is being used – You will need to find the new port |
| Scan target device | Nmap -sT ip.address   nmap -sT -p 0-1000 <ip.address>   nmap -sTV -p <target port> <ip.address> | Should now show that port 22, which was previously open, is ow missing / closed  You will need to re-run the scan and if the new port wasn’t shown then add in some port ranges to check for (provided). For example if the port picked was 999  This will then show the new SSH port (but likely won’t have it listed as SSH). To have the correct version info run with the -sTV and target port |
| SSH onto target device | ssh kali@<ip.address> -p <target port> | Login as before |
| The instructor will now change the kali user password | - | This will provide the setup for a bruteforce attack – While also highlighting issues around the use of default and easily guessed passwords |
| SSH onto target device | ssh kali@<ip.address> -p <target port> | Login as before – This should now fail due to the passwordchange |
| Bruteforce the login using fasttrack | hydra -l kali -P /usr/share/wordlists/fasttrack.txt <ip.address> ssh -s <target port> -V | This will run Hydra against the target device and will bruteforce the login – You should eventually get the correct username and password (this may take a few minutes) |
| SSH onto target device | ssh kali@<ip.address> -p <target port> | Login as before (using new creds) |

Session3 – Firewall rules using iptables. Here we build on the basics previously covered and look at some network configuration, changes and how these can be utilised to work around issues such as blocked IPs.

Please note that the Pi WiFi access point connection can be become unstable after manually changing an IP address. If this happens simply disconnect and reconnect (this is covered in the screenshots at the end of this document)

|  |  |  |
| --- | --- | --- |
| Action | Commands (via CLI) | Description |
| Block a random IP address | iptables -A INPUT -s <ip.address> -j DROP | This shows the use of iptables which can be used to drop or reject traffic to a device based on certain criteria (such as IP address or protocol) |
| Re-run the previous ping | ping <ip.address> | You should now notice that the blocked device can no longer ping you. This is due to the iptables rule |
| Remove current iptables rule and add in a new one | iptables -D INPUT 1 iptables -A INPUT -p ICMP -j DROP | We delete the previous rule and now we are blocking all ICMP traffic |
| Re-run the previous ping | ping <ip.address> | This time NO pi’s should be unable to ping you |
| Connect to HTTP on the target device | (via web browser) uri = <ip.address> | You should notice that the devices can all connect to the splash page – This highlights the different protocols used |
| Remove current iptables rule and add in a new one | iptables -D INPUT 1 iptables -A INPUT -p TCP -j DROP | Delete the previous rule and now block all TCP traffic – This will stop HTTP connection(s) |
| Connect to HTTP on the target device | (via web browser) uri = <ip.address> | This time the devices should be unable to access the splash page / it should hang on a refresh |
| Re-run the previous ping | ping <ip.address> | This time all pi’s should be able to ping the target device – Again we are highlighting the difference in protocols |
| Scan the target device | nmap -sT <ip.address> | Your scan results should come back with limited to no info – This is because we are blocking TCP |
| Scan the target device using UDP | nmap -sU <ip.address> | The results from this will be limited as well (and slow!)– But it will show UDP usage as well as highlight the different protocols |
| Block a specific IP address | iptables -D INPUT 1 iptables -A INPUT -s <ip.address> -j DROP | Showcase use of iptables / simulation of firewalls (sort of) – Also make sure we delete any pre-existing rules, if for example we are following on from the “Basics” |
| Ping + browser connect to target device | ping ip.address (via web browser) uri = <ip.address> | All devices except the blocked device should be able to access the target device with no issues |
| Change IP address | (via GUI) – Please see the end of the worksheet for screenshots | This will show how a device can change it’s IP address and use that to evade / workaround things like a block. |
| Ping + browser connect to target device | ping ip.address (via web browser) uri = <ip.address> | All devices should now be able to access the target device with no issues |
| Delete iptables rule and revert to DHCP for all PIs | iptables -D INPUT 1 (via GUI)  To revert to DHCP please see the screenshots at the end of this document. Where we currently set the Method to “Manual” change this back to “Automatic (DHCP)” | Point demonstrated we don’t need to risk the flake really |

|  |  |
| --- | --- |
| Description & Related Action | Screenshot |
| On your Kali in the top right hand corner of the screen you will see multiple symbols. To edit the connection settings we need to right click on the current connection.  In this example it is a Ethernet symbol – On your machine it will likely be a WiFi symbol.  From there select edit connections (highlighted in this screenshot) | A screenshot of a computer  Description automatically generated |
| This will open a new menu as shown here – We will want to edit the currently used connection. In this case that is “Wired 1” in your case it should be “**UWEcyber-RasPwnOS**”.  Click on the connection and then click the settings symbol (highlighted) | A screenshot of a computer  Description automatically generated |
| This will open up a new menu – This is where we will make changes to the current IPv4 settings (and therefore our address).  To make sure we are using the correct information we will open up a terminal and run the command:  “route -nee”  This shows our route information, including the gateway and netmask. We would then want to use that information to manually set our new address.  In this instance if we wanted to set our new address to “164.11.20.100” we would add the following:   |  |  |  | | --- | --- | --- | | Address | Netmask | Gateway | | 164.11.20.100 | 255.255.252.0 | 164.11.20.1 |   Based on the output of our “route -nee” command  We also need to set the Method to “Manual”  Finally once you have made the relevant changes click  “Checkmark with solid fillSave”  For the changes to take effect you will need to disconnect and re-connect. This can be done by simply left clicking on the same symbol you originally clicked on in step 1 above. Then click disconnect.  Repeat the process to connect by (left) clicking the symbol and clicking the connection name | A screenshot of a computer  Description automatically generated |
| To stop people connecting to your Pi via SSH you need to disable the service (which stops it starting up again) and then stop it. This is done using the two “sudo systemctl” commands shown here.  systemctl disable sshd systemctl stop sshd  Finally you will want to terminate all active SSH connections. This is done using the third command. Please note this should be copied and pasted as one command (as shown here).  lsof -w -t +d /dev/pts/ | sort | uniq | \  xargs ps -oppid= -p | \  xargs ps -ocomm=,pid=,user= -p | \  awk '($1 == "sshd") {print $2}' | \  xargs kill |  |