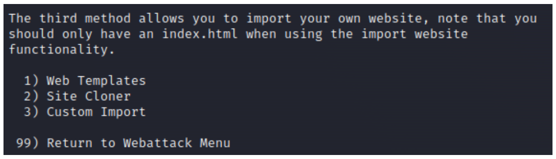
Deviations from design

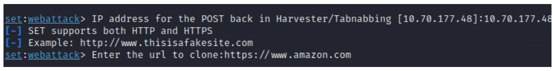
Switch from Apache to SEToolkit for Web Hosting:
In the original plan, we used Apache to locally host the fake website on the attacker's computer, following the tutorial for Part 1. While Apache worked on both Kali and Windows, we decided to switch to SEToolkit for a more automated approach. SEToolkit provided pre-cloned templates and automatically logged credentials from form inputs, making the attack more efficient. We used this tutorial to set up SEToolkit and followed this video to configure the cloning attack.

Decision to Use Gmail Login Instead of Amazon:
Initially, we planned to use Amazon's site for the credential logger. Although SEToolkit allows manual cloning of websites, we encountered issues with Amazon and Facebook login pages. The XML log file did not properly capture the username and password. This led us to use SEToolkit's pre-cloned Gmail template instead for demonstration purposes. Screenshots of our attempts with Amazon are attached for reference:



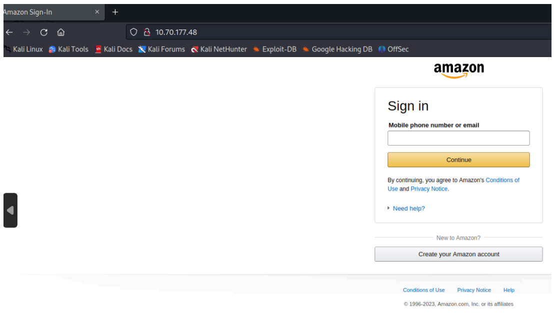
```
The third method allows you to import your own website, note that you
should only have an index.html when using the import website
functionality.

  1) Web Templates
  2) Site Cloner
  3) Custom Import

 99) Return to Webattack Menu
```

1) We chose 2 because Amazon was not in the list of pre-cloned templates.



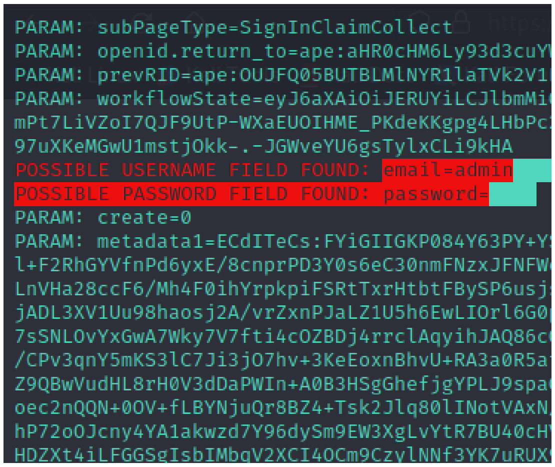
```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.70.177.48]:10.70.177.48
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:https://www.amazon.com
```

2) The pictured text is small. It is a screenshot of the SEToolkit asking for the url of the site.



3) After providing the URL, it successfully cloned the login page.

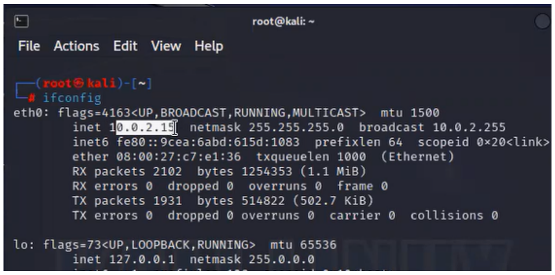
4) The credential logger only ended up working for the username because the site has a different page for the password, which was a good security decision but not ideal for the demonstration.



```
PARAM: subPageType=SignInClaimCollect
PARAM: openid.return_to=ape:aHR0cHM6Ly93d3cuY
PARAM: prevRID=ape:OUJFQ05BUTBLMlNYR1laTVk2V1
PARAM: workflowState=eyJ6aXAiOiJERUYiLCJlbmMi
mPt7LiVZoI7QJF9UtP-WXaEUOIHME_PKdeKKgpg4LHbPc
97uXKeMGwU1mstjOkk-.-JGWveYU6gsTylxCLi9kHA
POSSIBLE USERNAME FIELD FOUND: email=admin[redacted]
POSSIBLE PASSWORD FIELD FOUND: password=[redacted]
PARAM: create=0
PARAM: metadata1=ECdITeCs:FYiGIIGKP084Y63PY+Y
l+F2RhGYVfnPd6yxE/8cnprPD3Y0s6eC30nmFNzxJFNFW
LnVHa28ccF6/Mh4F0ihYrpkpiFSRtTxrHtbtFBySP6usj
jADL3XV1Uu98haosj2A/vrZxnPJaLZ1U5h6EwLIOrl6G0
7sSNLOvYxGwA7Wky7V7fti4cOZBDj4rrclAqyihJAQ86c
/CPv3qnY5mKS3lC7Ji3jO7hv+3KeEoxnBhvU+RA3a0R5a
Z9QBwVudHL8rH0V3dDaPWIn+A0B3HSgGhefjgYPLJ9spa
oec2nQQN+0OV+fLBYNjuQr8BZ4+Tsk2Jlq80lINotVAxN
hP72oOJcny4YA1akwzd7Y96dySm9EW3XgLvYtR7BU40cH
HDZXt4iLFGGSgIsbIMbqV2XCI4OCm9CzylNNf3YK7uRUX
```

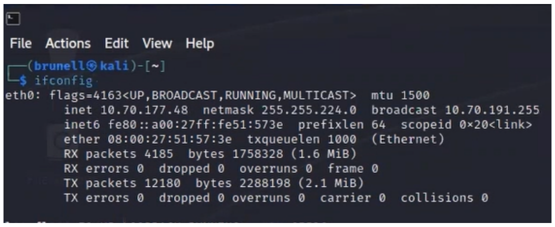
Challenges with the Virtual Machine:

Running Kali on a virtual machine posed several difficulties. Initially, we were confused by the IP mismatch between the Kali VM and the Windows 10 device, not realizing that it was normal. All our Kali machines had the same IP address, 10.0.2.15, which is Oracle's default for virtual machines. This also explained why our scans only detected three devices on the Wi-Fi network. After researching the issue, we found that resolving the IP problem expanded our network scan significantly. Once the IP resolved to 10.70.177.48, instead of the default, and the broadcast range was updated, we were able to detect many more devices with IPs ranging from 10.0.2.255 to 10.70.191.255, which included our own devices. We followed this tutorial to fix the IP issue. Screenshots for reference:



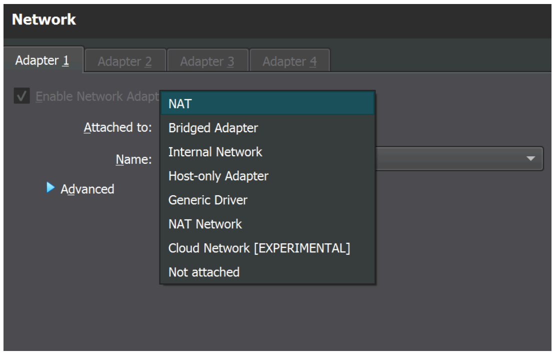
```
┌──(root㉿kali)-[~]
└─# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::9cea:6abd:615d:1083  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:c7:e1:36  txqueuelen 1000  (Ethernet)
        RX packets 2102  bytes 1254353 (1.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1931  bytes 514822 (502.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
```

The IP of Haley's Kali

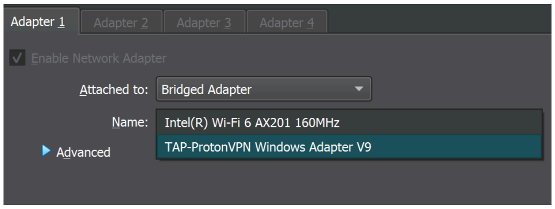


```
┌──(brunell㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.70.177.48  netmask 255.255.224.0  broadcast 10.70.191.255
        inet6 fe80::a00:27ff:fe51:573e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:51:57:3e  txqueuelen 1000  (Ethernet)
        RX packets 4185  bytes 1758328 (1.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12180  bytes 2288198 (2.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The IP of Chirag's Kali



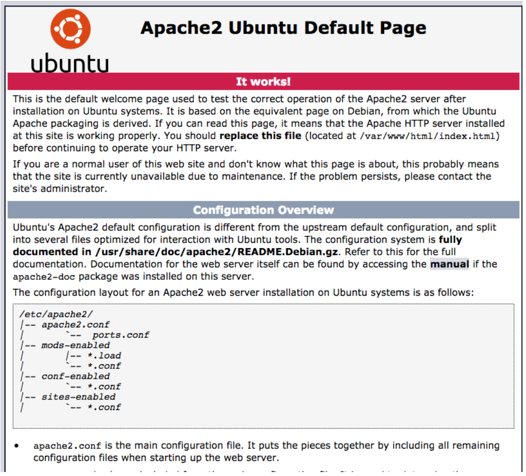
We simply switched the network setting in Oracle from NAT to Bridged Adapter, which allowed the virtual box to show its true IP, enabling the demo to run on the same device.



☑ Enable Network Adapter
Attached to: Bridged Adapter
Name: Intel(R) Wi-Fi 6 AX201 160MHz
TAP-ProtonVPN Windows Adapter V9

Selecting Bridged Adapter also lets you connect to a VPN adapter, which we would have used if the machine's IP matched the device's, which it did not.

Bypassing Apache Configuration:
Switching to SEToolkit eliminated the need to configure Apache and Ettercap for the attack, as described in the original tutorial. However, we did spend considerable time working with Apache and successfully hosted sites before moving on to SEToolkit. The remaining task with Apache would have been to capture POST requests and log form data, but SEToolkit handled this automatically.



Apache2 Ubuntu Default Page

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

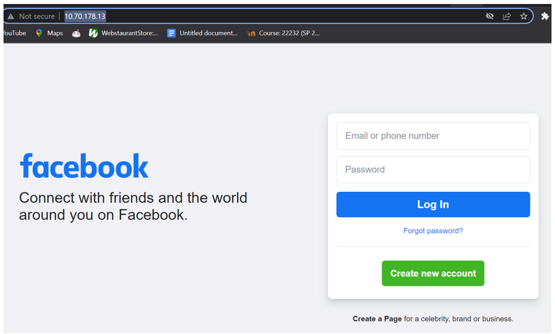
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.

The default apache page stored as index.html in Apache24 > htdocs



When we entered the device's IP in the browser, it displayed the index.html. To clone any site, like facebook.com, we could simply right-click, select "View Page Source," copy the HTML, and paste it into index.html.

Miscellaneous Adjustments:

Initially, I didn't have a virtual machine on my laptop and attempted to use the Kali Linux app from the Microsoft Store. Although I managed to install the non-graphical version of Ettercap, it was difficult to get it running. Eventually, I decided to set up a full virtual machine using Kali, following this tutorial. A project limitation we encountered was that, although ARP poisoning and DNS spoofing worked, the cloned website wasn't accessible on the target computer via the domain name; we had to manually enter the attacker's IP address for the demo.

Despite these deviations, the project's core purpose remained unchanged from Part 1: demonstrating a phishing attack that steals user credentials by masquerading as a legitimate site. As originally planned, we positioned ourselves between the victim and the router using Ettercap's ARP poisoning, launched a DNS spoofing attack to redirect traffic from the legitimate site to our cloned site, and captured the credentials entered by the user.

Controls against the attack

Several measures can help mitigate the type of phishing attack we executed:

1. **Double-Page Logins**:
   Websites like Amazon use double-page logins where the user first submits their email, then is directed to a separate page to enter their password. Since *setoolkit* can only clone and host one webpage at a time, this design effectively counters our attack by splitting the login process across multiple pages.
2. **Browser Security Alerts**:
   As shown in our video, Google Chrome includes a security feature that detects when an attack is successful. It warns the user that their data may have been compromised and advises them to change their password immediately. This can thwart the attack if the

user changes their credentials before we can exploit them further. Another giveaway is that after clicking "login," no action occurs, which should raise suspicion. Either way, using a secure and updated browser should considerably improve the user's security against a similar attack.

3. **URL Inspection**:
   A common indicator of phishing is the URL itself. If users inspect the URL and notice that the site is not secure (i.e., lacks HTTPS), they can recognize it as a fake. Social engineering attacks rely on the appearance of the site to deceive users, so educating people to always check the link for secure connections before entering personal information is an effective countermeasure.
4. **MAC Filtering**:
   Implementing MAC address filtering at the router level ensures only authorized devices can connect to the network. This control prevents attackers from gaining access to the network by limiting connectivity to pre-approved devices.
5. **Packet Filtering and Anti-Spoofing Policies**:
   Since the attack targets Layer 3 (network) and Layer 4 (data link), a packet filtering gateway with strong filtering rules and an anti-spoofing policy can block malicious traffic. Policies such as disallowing outbound packets with a source IP not associated with the ISP, or blocking inbound packets with spoofed IPs, can prevent attacks like ours.
6. **Source Routing Controls**:
   Disabling source routing in the network configuration ensures that attackers cannot manipulate the routing of packets, helping to prevent cache poisoning. Ensuring that the best route from source to destination is used further reduces vulnerabilities.
7. **ARP and Broadcast Monitoring**:
   Using ARP monitoring software helps detect man-in-the-middle attacks, like ARP poisoning, by flagging unusual traffic patterns. Similarly, broadcast monitoring and broadcast abuse controls can be implemented to identify and mitigate abnormal broadcast traffic.
8. **DNS Validation**:
   Ensuring that the local DNS server validates DNS responses protects against DNS spoofing attacks. This adds another layer of defense against domain redirection.
9. **Encrypted Form Data**:
   Encrypting form data before sending it via the POST method keeps the data secure, even if it is intercepted. Attackers would be unable to extract any meaningful information from the intercepted packets without the encryption keys.

Additional Citations

- *How to Use the Social Engineering Toolkit (SET) in Kali Linux for Phishing.* YouTube Video
- *Resolving IP Address Issues in VirtualBox for Kali Linux.* YouTube Video
- *Website Cloning in Kali Linux Using the SET Toolkit.* Medium Article
- *Installing Kali Linux in VirtualBox: A Step-by-Step Guide.* YouTube Video
- *Using Ettercap for Network Sniffing and ARP Spoofing in Kali Linux.* YouTube Video