Wireless Survey/Phishing

Wireless Survey

What did I do?

I conducted a network survey in my area using Wireshark, focusing on open or unsecured wireless networks. I captured wireless traffic to analyze the protocols and data transmitted over the open networks. I also checked the range of my own network to see how far its signal extended beyond my home and tested its security.

Tools Used

Wireshark: A powerful network protocol analyzer that captures and displays network traffic. It is particularly useful for diagnosing network issues and monitoring for suspicious activity. Wireshark allows for the analysis of a wide range of protocols, making it ideal for identifying security flaws like unencrypted data transmissions.

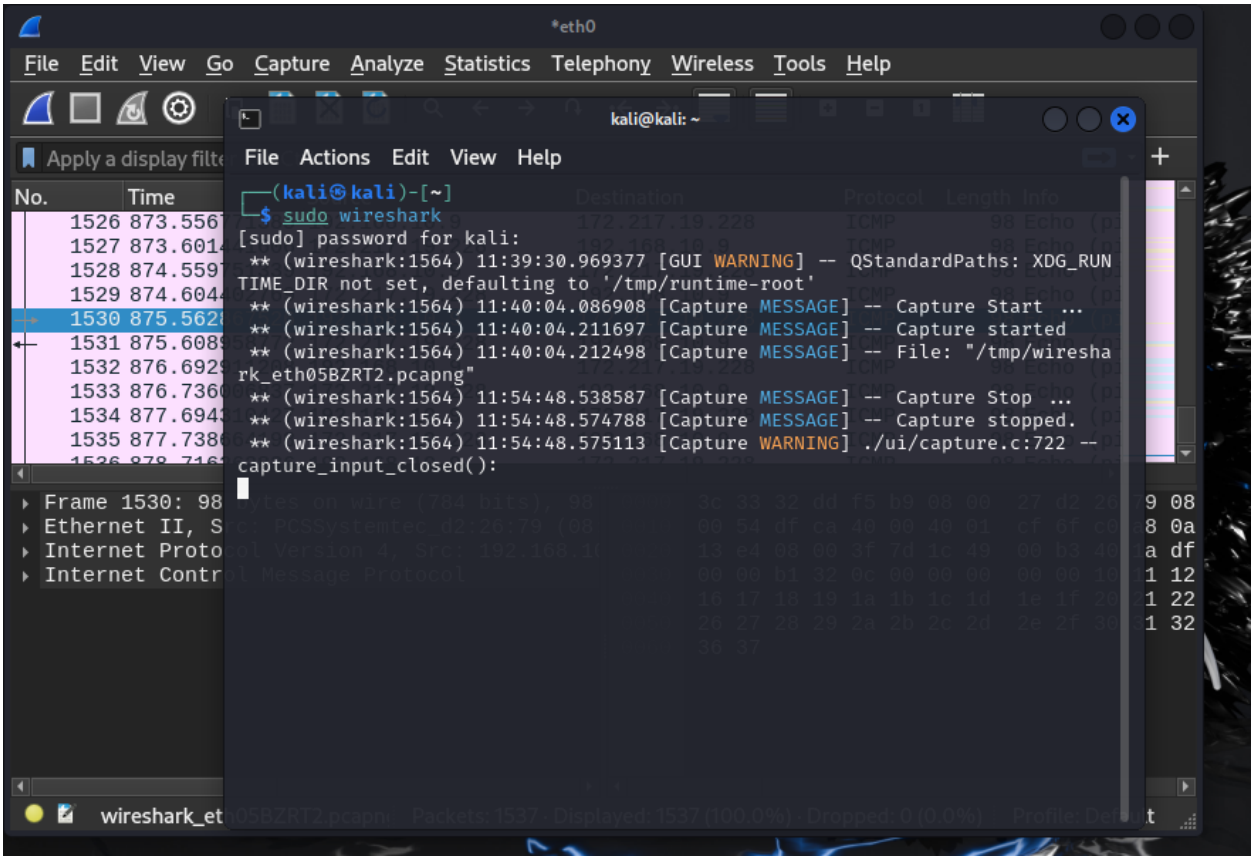
Kali Linux: An operating system specialized for penetration testing and network security analysis. Wireshark runs seamlessly on Kali Linux, providing a robust environment for monitoring wireless traffic and testing network defenses.

Installing Wireshark on Kali Linux

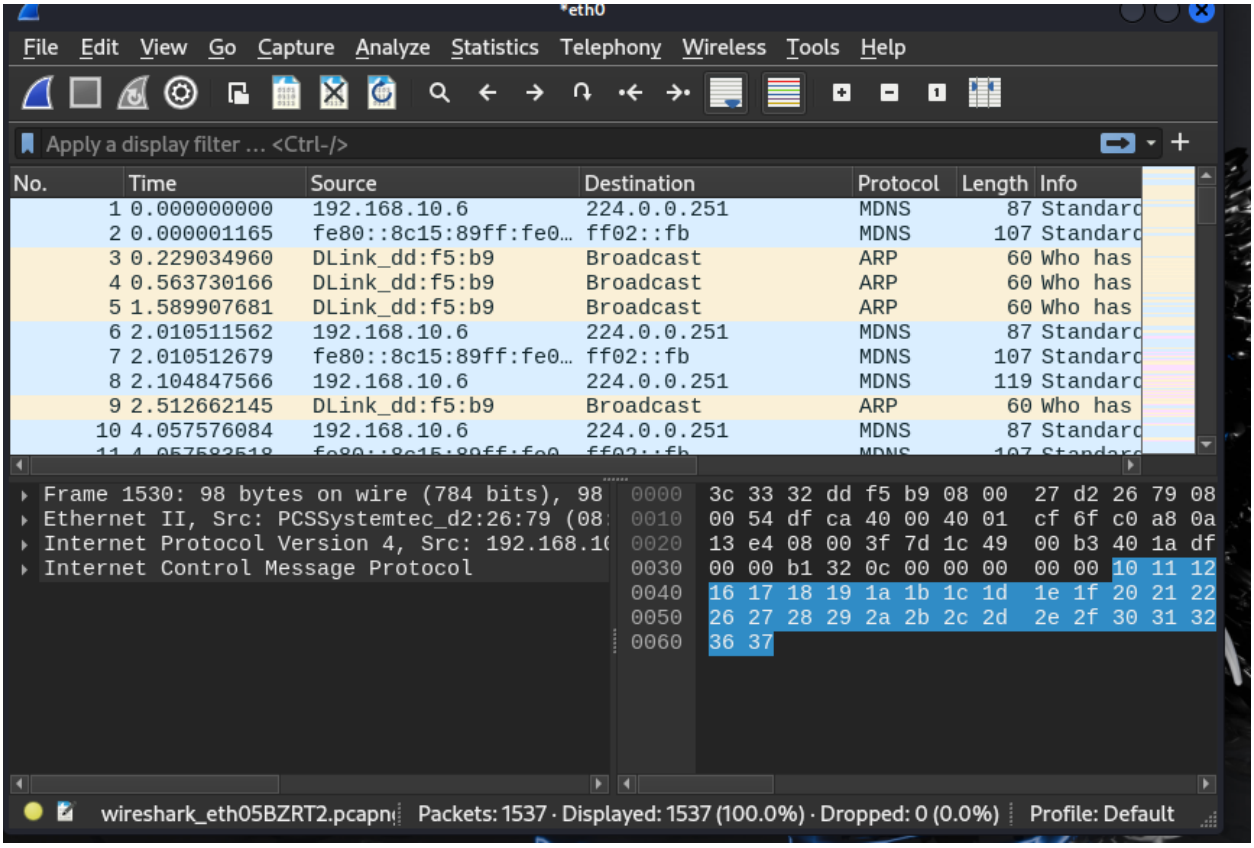
I began by installing Wireshark on my Kali Linux virtual machine. Kali Linux is a popular operating system for penetration testing and network analysis, and Wireshark is one of the most powerful tools available for network traffic analysis. To install Wireshark, I used the following commands:

During the installation, I was prompted to decide whether to allow non-root users to capture packets. I selected **Yes** to enable packet capturing without needing elevated privileges.

After installation, I launched Wireshark through the terminal with the command:

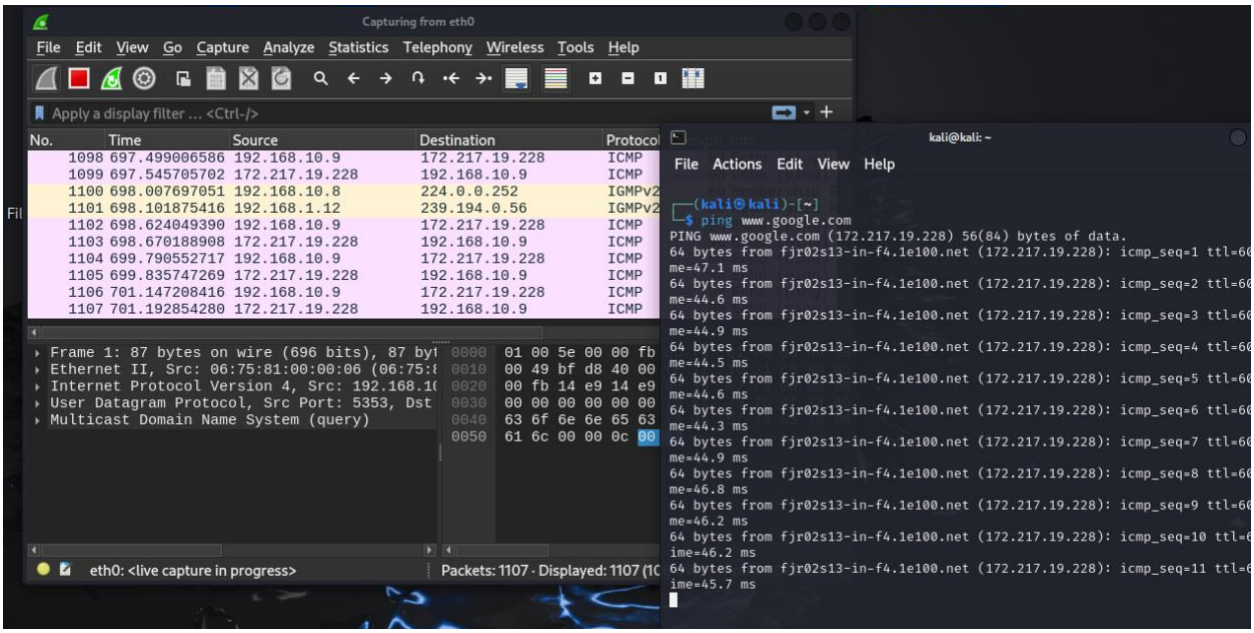


This opened the Wireshark graphical interface, which I used to interact with and analyze network traffic in real-time.



What were the results?

- From the network capture, I detected several wireless networks, some of which were unsecured or had weak encryption protocols (e.g., WPA, WPA2).
- Once Wireshark was running, I selected my **wireless network interface card** (WLAN) to capture wireless network traffic. Since I was working in a virtualized environment, I used a USB wireless adapter for this task. I made sure that **monitor mode** was enabled on my wireless interface card so that I could capture all the traffic in the air, not just traffic directed to my machine.
- To begin capturing, I clicked on the **Start Capture** button and Wireshark immediately began logging packets of network data flowing through the wireless network.
- After starting the capture, I limited the traffic to **Wi-Fi data only** by setting appropriate filters.
- Once the capture was in progress, I performed a series of actions on my network to generate different types of network traffic. The first thing I did was open a terminal and use the following command to ping Google:



This generated **ICMP (Internet Control Message Protocol)** traffic, as ping works by sending ICMP echo request and echo reply to messages. As expected, I started seeing ICMP packets in my capture.

In addition to ICMP traffic, Wireshark also captured other types of network traffic that I did not directly initiate but were part of my virtual machine's network environment. These included various protocols such as:

- **UDP (User Datagram Protocol)**
- **SSDP (Simple Service Discovery Protocol)**
- **MDNS (Multicast DNS)**
- **LLMNR (Link-Local Multicast Name Resolution)**
- **IGMPv2 (Internet Group Management Protocol version 2)**
- **DHCP (Dynamic Host Configuration Protocol)**
- **ARP (Address Resolution Protocol)**
- **DHA (Dynamic Host Allocation)**
- **HTTP (Hypertext Transfer Protocol)**

These protocols were generated by background processes or by devices in my network that communicated using multicast or broadcast traffic.

Once I had captured a significant amount of data and analyzed different network protocols, I saved the captured packets for future reference. Wireshark allows you to save captures in various formats, including the default. pcap format. I saved my capture by selecting **File** > **Save As** and saved it with a descriptive filename.

What did you learn?

Understanding Network Protocols

One of the most valuable things I learned from this exercise was how to analyze various network protocols in real-time. Wireshark's detailed breakdown of each packet allowed me to examine how different layers of the network stack work together to transmit data. Specifically, I learned:

- How **ICMP** works for network diagnostics.
- How **ARP** functions for address resolution.
- The difference between unicast, multicast, and broadcast traffic (as seen with **MDNS** and **LLMNR**).
- The role of **DHCP** in dynamic IP address assignment.

1. **Packet Filtering and Analysis**

Wireshark's filtering capabilities are incredibly powerful. By applying filters, I could focus on specific types of traffic, such as HTTP or ICMP, which made the analysis more efficient and manageable. I learned how to create custom filters and use logical operators to combine multiple filters, such as:

2. **Traffic from Virtualized Environments**

Running Wireshark on a virtual desktop introduced me to some unique network traffic, particularly **UDP protocols** associated with virtualization and local services like **SSDP**. This gave me a better understanding of how virtual machines interact with their host systems and the network.

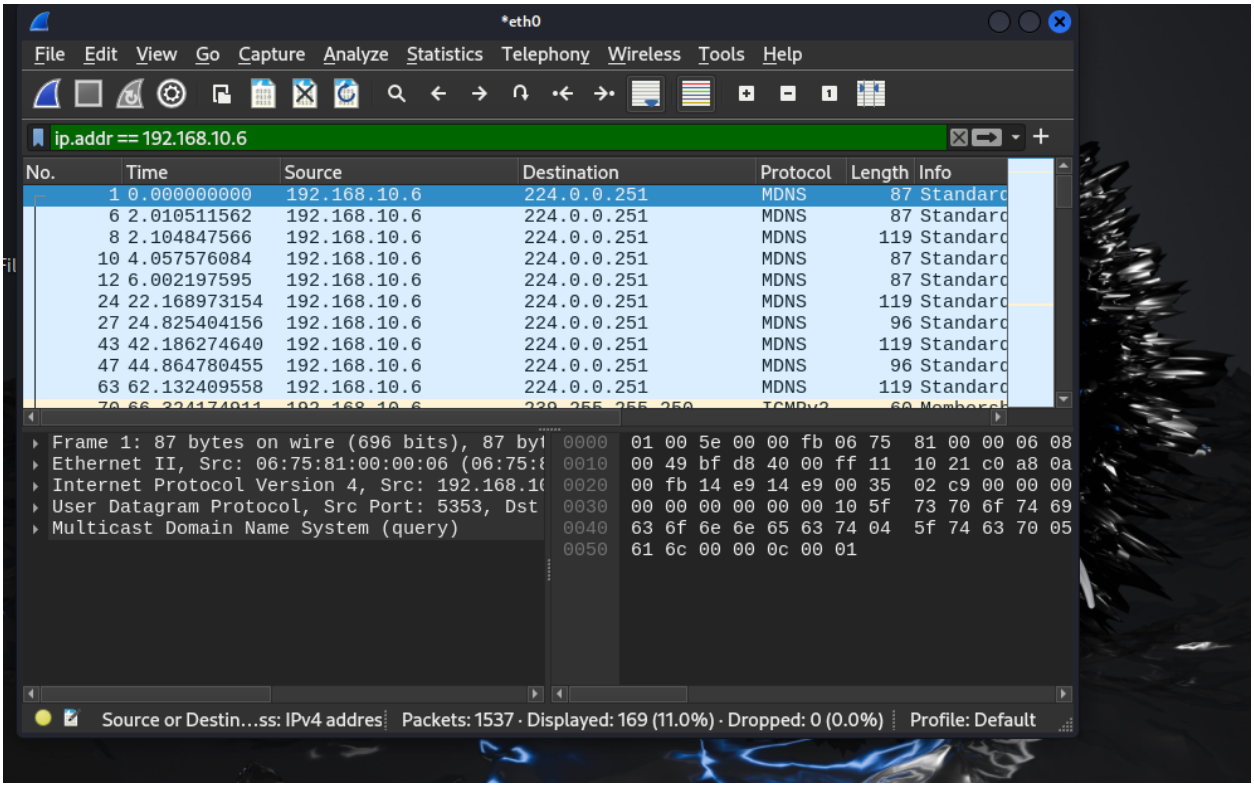
3. **Security Implications**

This exercise also reinforced how easy it can be to capture unencrypted data on a network. Without any special privileges, I was able to capture HTTP traffic, including details of the requests and responses. This highlights the importance of encryption (e.g., using HTTPS instead of HTTP) for securing network communications. In real-world scenarios, unsecured traffic could easily be intercepted by malicious actors using tools like Wireshark.

4. **Real-World Use Cases**

Wireshark is not only useful for academic purposes but also has practical applications in the real world, including:

- **Troubleshooting network issues:** Identifying bottlenecks, high latency, or packet loss.
- **Security audits:** Monitoring for suspicious activity such as unusual DNS queries or unauthorized HTTP requests.
- **Learning networking concepts:** Seeing real traffic flow helps reinforce theoretical concepts like the OSI model, TCP/IP stack, and various network protocols.



Phishing Test

What did I Did?

I conducted the phishing tests for two different websites to get an idea about how I can differentiate between the phishing and actual messages sent by the website. The above tests give various emails, websites and login page, out of which some were real and some of them were of the phishing groups but designed very brilliantly.

http://www.sonicwall.com/phishing/phishing-quiz-question.aspx
https://www.opendns.com/phishing-quiz/

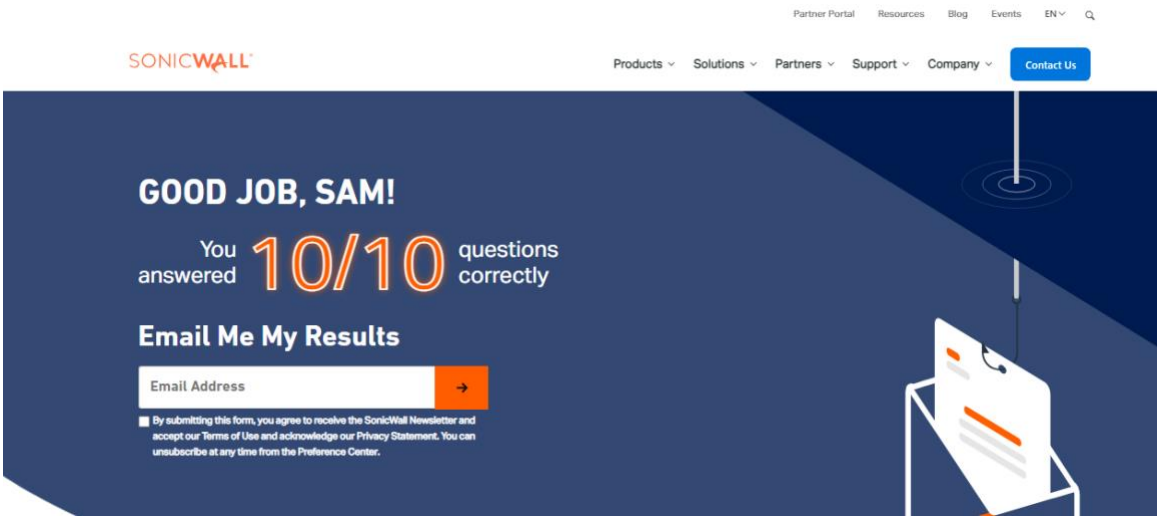
These platforms provided realistic simulations of phishing emails and websites. Through these tests, I learned to identify common phishing tactics such as the use of forged URLs, scare tactics, and poor grammar.

In the first one, the goal was on recognizing phishing emails. The test offered several cases, for which I would have to decide on the sender's e-mail, the domain, grammar, and the lettering style. I had to rely on my knowledge of how phishing messages look like, some of the features being fake links, wrong grammar, and threats.

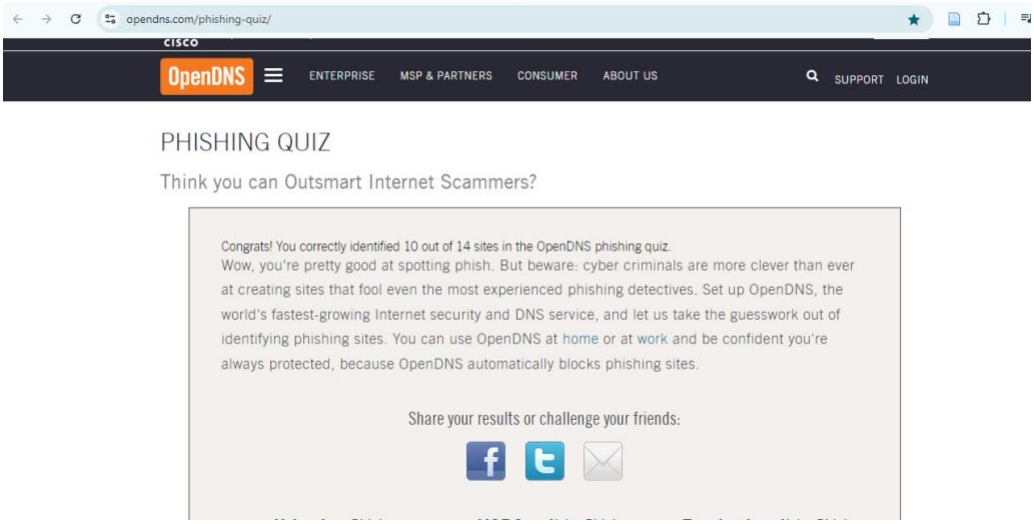
The second test was more general in nature and encompassed the ability to recognize phishing attempts on Web sites. This test presented several real and fake login pages, fake promotion offers, and other sites which might be used as a link for a phishing attack. I examined each page based on the key points that would tell the version of the site that was safe to use as well as the URLs used as well as the authenticity of the page.

What Were the Results?

During the first trial of the phishing emails, I performed flawlessly: I got all ten correct, meaning I was able to distinguish between phishing scams and legitimate emails. This proved that I possess a clear understanding of phishing indicators in matters relating to emails.



I got 11/14 in the second test which touched the area of website phishing. I successfully classified the majority of the sites as phishing and only failed to detect three, meaning that, even the most vigilant users can be sometimes fooled by a well-designed phishing site.



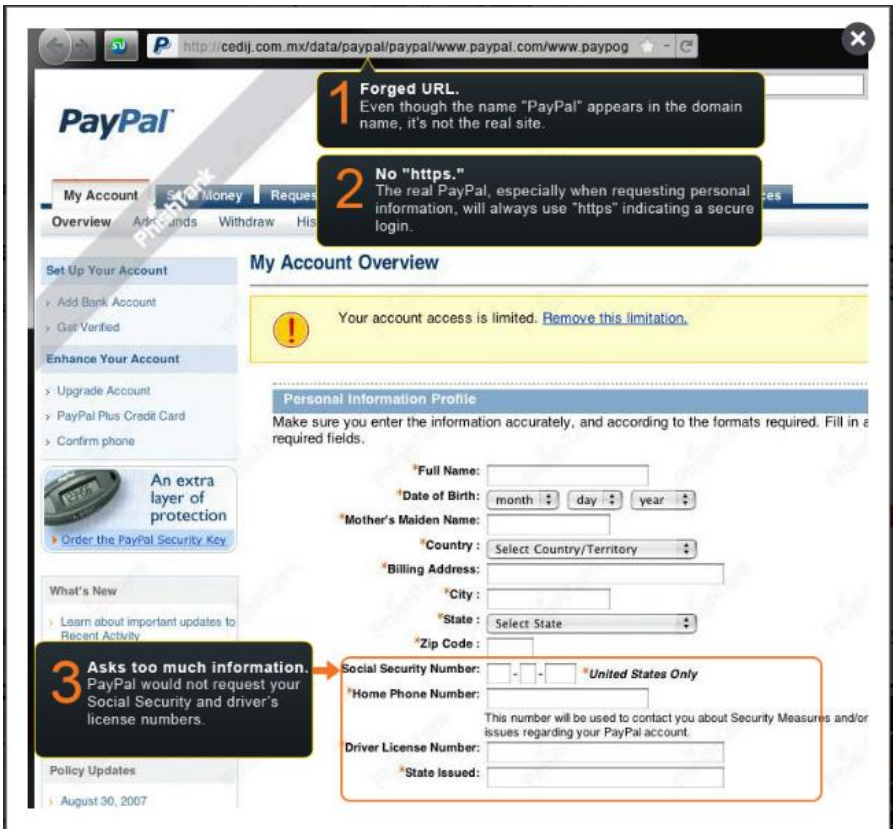
The screenshots I took show examples of the different elements I reviewed, such as:

- **Forged URLs**: In many phishing cases, the URL appears legitimate at first glance but contains subtle errors like extra characters or slight misspellings of trusted domains.

- **No HTTPS**: Some phishing sites did not use secure HTTPS protocols, a clear indicator that the site was not authentic.

- **Poor Grammar**: Many phishing attempts used broken English or confusing phrases, which is often a giveaway of a scam.

- **Scare Tactics**: Fake emails often used threats like "Your account will be deleted" or "Verify now to avoid closure" to push users into making hasty decisions.



What I Learned?

These phishing tests reinforced several key lessons about how to spot phishing attempts:

1. **Always Inspect URLs Closely:** The common tricks employed by the attackers include strategic use of URLs, which resemble the genuine ones but have slight variations. This means that being keen in identifying these slight differences can go a long way in helping one avoid the scams.
2. **Check for HTTPS Security:** A genuine website will have an encrypted protocol (https) at least on the login page. If a site requires user to input some details and this is not provided, then such a site may be a phishing site.
3. **Watch for Poor Grammar:** Another characteristic feature, which is typical for the phishing messages, and links are rough uses of language and grammar. The problem is that any professional company always carefully selected with their words, and obvious typos should alert the user.
4. **Avoid Acting on Scare Tactics:** Phishing emails also tend to use emotions as one of their tactics by applying pressure by threatening that the account will be closed,

or by stating that there were some activities going on that are suspicious. That is why it is wise not to act impulsively when receiving messages like that and check whether they are genuine or not by contacting the company.

5. **Phishing Techniques Are Getting More Sophisticated**: While I managed to get good results, an example of this is that I only failed the website test thrice in the phishing attempts, which illustrate that the common scams are changing. Phishing pages range from simple to very advanced and very subtle, so some of them are almost impossible to tell are fake even for a savvy person.
6. **Continuous Learning is Crucial:** Today phishers are also very active and that is why one should never stop sharpening his/her ability to recognize such threats. Since phishing attacks to depend on the ability of the targets to fall into the tricks, it is wise to from time to time engage into practice exercises of the same to reduce the impact of future attacks.

Insights and Potential Future Applications

This experience demonstrated how accessible and powerful network analysis tools like Wireshark are for both defenders and attackers. The growing complexity of phishing attacks also underscores the need for continuous education and awareness among users.

Looking ahead, the skills developed in this exercise could be applied to:

1. **Cybersecurity Audits**: Regular monitoring of network traffic to detect unusual or unauthorized activities is a key preventive measure against cyber threats.
2. **Forensic Analysis**: In the event of a data breach or cyber-attack, Wireshark can help reconstruct the event by analyzing captured traffic, offering insights into how the attack unfolded.
3. **Threat Detection**: Using real-time packet analysis tools in an organizational setting can help detect phishing attempts or suspicious network behavior before damage occurs.

By developing these skills, I am now better equipped to understand and mitigate the risks associated with both unsecured networks and social engineering attacks.

References

CaptureSetup/WLAN. (2022, 08 11). Wireshark. https://wiki.wireshark.org/CaptureSetup/WLAN

Găină, M. A., Popescu, M., Dobre, R. A., & Vizireanu, D. N. (2023, March). Traffic sniffing in wireless communication, using Kali Linux and Wireshark. In *Advanced Topics in Optoelectronics, Microelectronics, and Nanotechnologies XI* (Vol. 12493, pp. 678-683). SPIE.

Ndatinya, V., Xiao, Z., Manepalli, V. R., Meng, K., & Xiao, Y. (2015). Network forensics analysis using Wireshark. *International Journal of Security and Networks*, *10*(2), 91-106.

Sanders, C. (2017). *Practical packet analysis: Using Wireshark to solve real-world network problems*. No Starch Press.

Beale, J., Orebaugh, A., & Ramirez, G. (2006). *Wireshark & Ethereal network protocol analyzer toolkit*. Elsevier.

Free Cybersecurity Tools: Types, Examples, And Importance. https://impanix.com/security/free-cybersecurity-tools/