Data Exfiltration

Introduction

Data exfiltration is the unauthorized transfer of sensitive data from a computer or other device. It's a primary objective for attackers who have breached a network. Our job is to detect and stop this before sensitive information walks out the door.

Objectives

- Understand the common methods used for data exfiltration.
- Learn how to detect exfiltration attempts using network traffic analysis.
- Identify signs of exfiltration on endpoint devices.
- Correlate logs in a SIEM to uncover hidden exfiltration channels.

By:Fábio Vieira

Provided: TryHackMe

Data Exfiltration

Data exfiltration is when sensitive data is taken from an organization without permission. This can be done by insiders or external attackers using malware or stolen credentials.

Attackers steal data for money, espionage, ransom, or to prepare future attacks.
The process usually involves finding sensitive files, preparing them, and sending them outside the network.

Detection focuses on unusual outbound traffic, suspicious processes, and abnormal activity in network, host, and cloud logs.

Q1. Exfiltrating the data through HTTP comes under which technique?

Network-Based

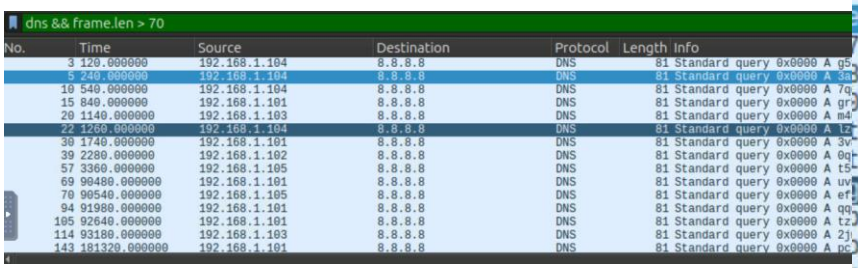
Network-based exfiltration is when attackers steal data by sending it out of the network over normal network traffic. They often use common protocols like HTTP, HTTPS, FTP, or DNS to blend in with legitimate activity.

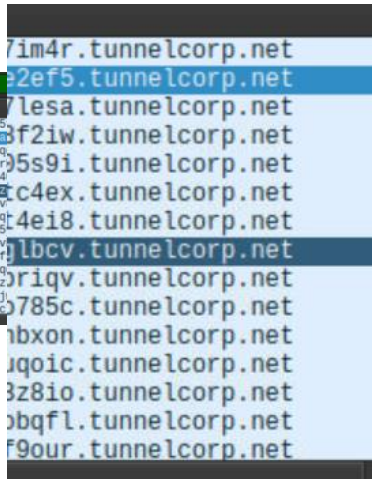
The goal is to hide malicious data transfers inside normal outbound connections, making detection harder. Signs of this technique include unusually large uploads, frequent outbound connections, or data being sent to unknown external servers.

Suspected DNS-Based Attack

There is a suspicion that a DNS-based attack may have occurred. The objective of this part is to analyze the situation using **Wireshark** and **Splunk** in order to identify malicious activity.

Starting with **Wireshark**, we can observe multiple types of packets. Some of them stand out due to their unusually large size. By applying the following filter "**dns && frame.len > 70"** we can see that these packets are all related to the domain tunnelcorp.net, which raises suspicion and suggests possible DNS tunneling activity.





This allows us to answer the second question:

Q2.What is the suspicious domain receiving the DNS traffic?

Tunnelcorp.net

Q3. How many suspicious traffic/logs related to dns tunneling were observed?

Q3.Which local IP sent the maximum number of suspicious requests?

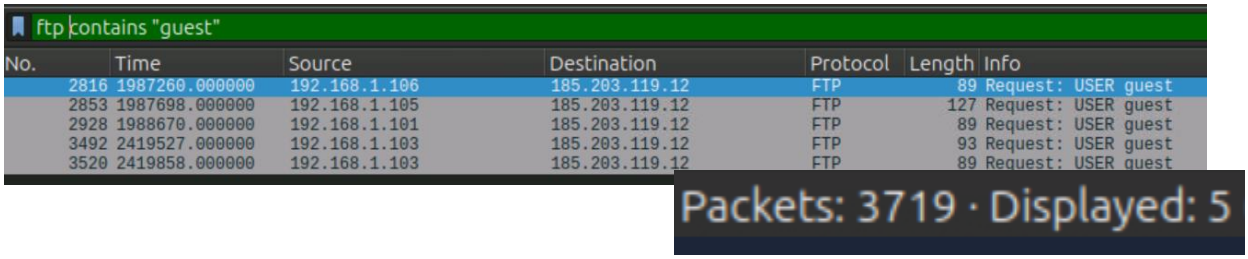
To answer this question, we switch to the SIEM **Splunk** and apply the filter **index="data_exfil" sourcetype="DNS_logs" | where len(query) > 30**. By analyzing the results, we can see in the src_ip field that the internal IP address responsible for sending the highest number of DNS requests is **192.168.1.103**.

Values	Count	%
192.168.1.103	72	22.857%
192.168.1.105	63	20%
192.168.1.104	62	19.682%

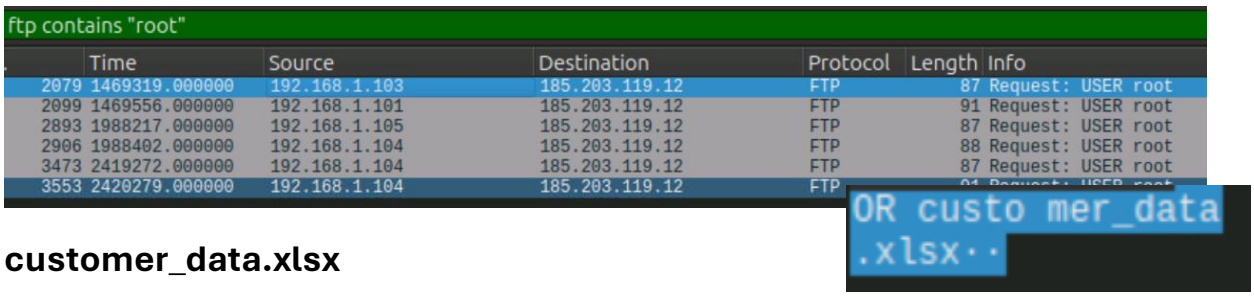
FTP Exfiltration

Attackers can use FTP to steal large amounts of data, often with stolen credentials. Signs of abuse include cleartext logins, file upload commands, and large data transfers to external servers. In this lab, the ftp-lab.pcap file is analyzed in Wireshark to identify FTP-based data exfiltration.

Q4. How many connections were observed from the guest account?



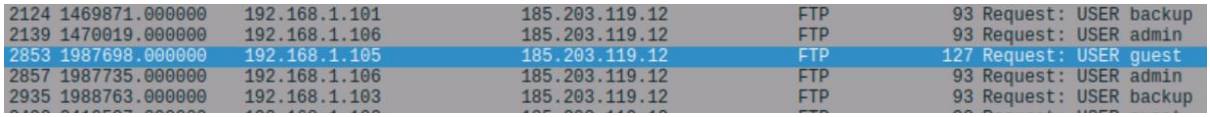
Q5.Apply the filter; what is the name of the customer-related file exfiltrated from the root account?



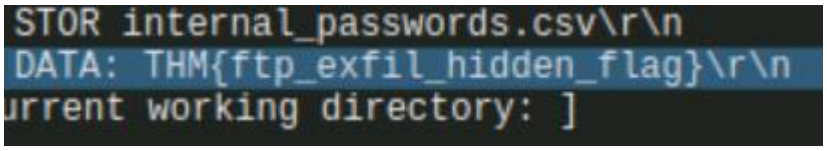
customer_data.xlsx

Q6. Which internal IP was found to be sending the largest payload to an external IP?

By applying the filter ftp && frame.len > 90, we can observe that the internal IP address exfiltrating the largest amount of data to an external destination is **192.168.1.105**.



Q7. What is the flag hidden inside the ftp stream transferring the CSV file to the suspicious IP?



HTTP Exfiltration

HTTP exfiltration occurs when attackers use HTTP(S) to move sensitive data outside a network, blending with normal web traffic. They may use POST requests, encoded GET queries, custom headers, chunked transfers, or cloud services to hide data.

Key indicators include unusually large POST requests, repeated small requests to the same host, transfers to uncommon or low-reputation domains, and multipart uploads.

We start by retrieving all HTTP logs with i**ndex="data_exfil" sourcetype="http_logs"** and then focus on POST requests using method=POST.

Next, we analyze the average, minimum, and maximum bytes sent per domain to identify unusual data transfers. Filtering out normal traffic, we isolate POST requests with large payloads (bytes_sent > 600) to highlight suspicious activity.

This analysis points to a single entry where a significant amount of data was uploaded to an external host.

Q8. Which internal compromised host was used to exfiltrate this sensitive data?

As seen below **192.168.1.103.**

Q9. What's the flag hidden inside the exfiltrated data?

By applying the filter **http.request.method == "POST" and frame.len > 750**, we can locate the suspicious POST request. By analyzing it using Follow → TCP Stream, we identify the exfiltrated content and recover the flag **THM{http_raw_3xf1ltr4t10n_succ3ss}**.

ICMP Exfiltration

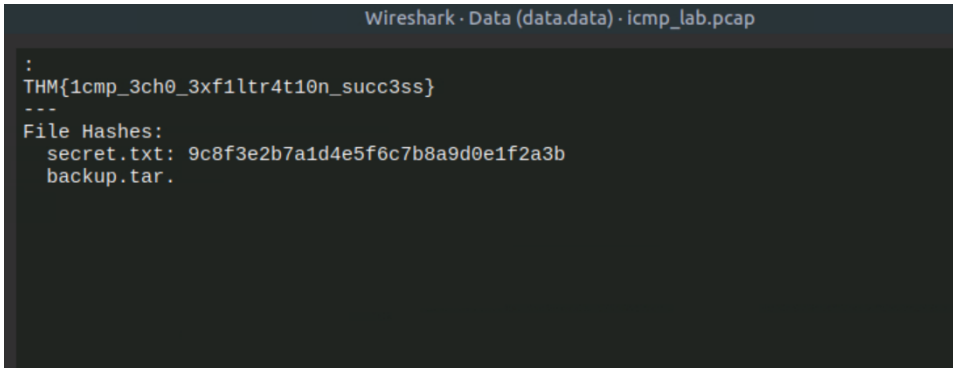
ICMP can be abused by attackers to exfiltrate data because it is often allowed through firewalls and poorly inspected. In this technique, data is encoded inside ICMP payloads and sent to an attacker-controlled host.

Common indicators include persistent ICMP traffic to unusual external hosts, large or frequent ICMP packets, high-entropy encoded payloads, and ICMP communication that does not match normal diagnostic behavior.

Q10. What is the flag found in the exfiltrated data through ICMP?

Start by filtering all ICMP traffic with icmp to identify unusual activity.
Next, isolate Echo Requests using icmp.type == 8.
Finally, focus on unusually large packets with icmp.type == 8 and frame.len > 100, as normal pings are around 74 bytes.

Any ICMP packet larger than usual may indicate suspicious exfiltration and should be investigated.



Conclusion

This lab covered multiple data exfiltration techniques and how to detect them using **Wireshark** and **Splunk**.

We analyzed **DNS-based exfiltration**, identifying suspicious domains (tunnelcorp.net) and internal hosts (192.168.1.103) sending unusual requests.

In **FTP exfiltration**, we observed large file transfers, identified customer-related files (customer_data.xlsx), and traced the internal host sending the largest payload (192.168.1.105).

For **HTTP exfiltration**, we filtered POST requests with large payloads, identified the compromised host (192.168.1.103), and recovered the exfiltrated flag **THM{http_raw_3xf1ltr4t10n_succ3ss}**.

Finally, **ICMP exfiltration** was examined by isolating large Echo Requests (icmp.type == 8 and frame.len > 100) to detect suspicious payloads.

Overall, detection relied on monitoring unusual traffic patterns, correlating logs with packet captures, and focusing on anomalies in network, host, and cloud activity. This lab highlights the importance of combined **packet analysis** and **SIEM log correlation** to uncover and investigate data exfiltration attempts effectively.