

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.

Data Exfiltration	1
Introduction.....	1
Objectives	1
Data Exfiltration	3
Q1. Exfiltrating the data through HTTP comes under which technique?.....	3
Suspected DNS-Based Attack	4
Q2.What is the suspicious domain receiving the DNS traffic?	4
Q3. How many suspicious traffic/logs related to dns tunneling were observed?	4
Q3.Which local IP sent the maximum number of suspicious requests?	5
FTP Exfiltration	5
Q4. How many connections were observed from the guest account?	5
Q5.Apply the filter; what is the name of the customer-related file exfiltrated from the root account?	6
Q6. Which internal IP was found to be sending the largest payload to an external IP?	6
Q7. What is the flag hidden inside the ftp stream transferring the CSV file to the suspicious IP?	6
HTTP Exfiltration.....	7
Q8. Which internal compromised host was used to exfiltrate this sensitive data?	7
Q9. What's the flag hidden inside the exfiltrated data?	8
ICMP Exfiltration	8
Q10. What is the flag found in the exfiltrated data through ICMP?	8
Conclusion	9

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.

No.	Time	Source	Destination	Protocol	Length	Info
3	129.000000	192.168.1.104	8.8.8.8	DNS	81	Standard query 0x0000 A 95
5	240.000000	192.168.1.104	8.8.8.8	DNS	81	Standard query 0x0000 A 34
19	548.000000	192.168.1.104	8.8.8.8	DNS	81	Standard query 0x0000 A 70
15	849.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A gr
20	1149.000000	192.168.1.103	8.8.8.8	DNS	81	Standard query 0x0000 A m4
22	1269.000000	192.168.1.104	8.8.8.8	DNS	81	Standard query 0x0000 A 12
30	1748.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A 3v
39	2280.000000	192.168.1.102	8.8.8.8	DNS	81	Standard query 0x0000 A 0q
57	3360.000000	192.168.1.105	8.8.8.8	DNS	81	Standard query 0x0000 A t5
69	98488.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A uv
70	98548.000000	192.168.1.105	8.8.8.8	DNS	81	Standard query 0x0000 A ef
94	91980.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A qq
105	92640.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A tz
114	93188.000000	192.168.1.103	8.8.8.8	DNS	81	Standard query 0x0000 A 2j
143	181320.000000	192.168.1.101	8.8.8.8	DNS	81	Standard query 0x0000 A pc

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?

dns && dns.qry.name contains tunnelcorp.net

Displayed: 315 (15.8%)

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?

FTP contains "guest"						
No.	Time	Source	Destination	Protocol	Length	Info
2816	1987260.000000	192.168.1.106	185.203.119.12	FTP	89	Request: USER guest
2853	1987698.000000	192.168.1.105	185.203.119.12	FTP	127	Request: USER guest
2928	1988670.000000	192.168.1.101	185.203.119.12	FTP	89	Request: USER guest
3492	2419527.000000	192.168.1.103	185.203.119.12	FTP	93	Request: USER guest
3520	2419858.000000	192.168.1.103	185.203.119.12	FTP	89	Request: USER guest

Packets: 3719 · Displayed: 5

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

ftp contains "root"						
	Time	Source	Destination	Protocol	Length	Info
2079	1469319.000000	192.168.1.103	185.203.119.12	FTP	87	Request: USER root
2099	1469556.000000	192.168.1.101	185.203.119.12	FTP	91	Request: USER root
2893	1988217.000000	192.168.1.105	185.203.119.12	FTP	87	Request: USER root
2906	1988402.000000	192.168.1.104	185.203.119.12	FTP	88	Request: USER root
3473	2419272.000000	192.168.1.104	185.203.119.12	FTP	87	Request: USER root
3553	2420279.000000	192.168.1.104	185.203.119.12	FTP	64	Request: USER root

customer_data.xlsx

OR 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**.

2124	1469871.000000	192.168.1.101	185.203.119.12	FTP	93	Request: USER backup
2139	1470019.000000	192.168.1.106	185.203.119.12	FTP	93	Request: USER admin
2853	1987698.000000	192.168.1.105	185.203.119.12	FTP	127	Request: USER guest
2857	1987735.000000	192.168.1.106	185.203.119.12	FTP	93	Request: USER admin
2935	1988763.000000	192.168.1.103	185.203.119.12	FTP	93	Request: USER backup
2949	2419557.000000	192.168.1.102	185.203.119.12	FTP	63	Request: USER root

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

ftp contains "csv"

```
STOR internal_passwords.csv\r\nDATA: THM{ftp_exfil_hidden_flag}\r\nCurrent working directory: ]
```

HTTP Exfiltration

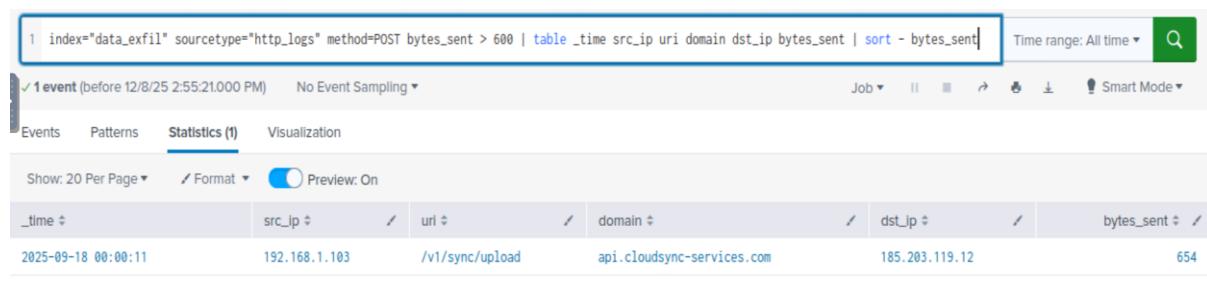
HTTP exfiltration occurs when attackers use HTTP(S) to move sensitive data outside a network, blending with normal web traffic. They may use POST requests, encoded GET queries, custom headers, chunked transfers, or cloud services to hide data.

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 **index="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.



1 index="data_exfil" sourcetype="http_logs" method=POST bytes_sent > 600 table _time src_ip uri domain dst_ip bytes_sent sort - bytes_sent						Time range: All time ▾	Search
✓ 1 event (before 12/8/25 2:55:21.000 PM) No Event Sampling ▾						Job ▾	Smart Mode ▾
Events Patterns Statistics (t) Visualization							
Show: 20 Per Page ▾	Format ▾	<input checked="" type="checkbox"/> Preview: On					
_time ▾	src_ip ▾	uri ▾	domain ▾	dst_ip ▾	bytes_sent ▾		
2025-09-18 00:00:11	192.168.1.103	/v1/sync/upload	api.cloudsync-services.com	185.203.119.12	654		

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.

The screenshot shows two windows. The top window is Wireshark with a table titled "icmp.type == 8 and frame.len > 100". It lists several ICMP Echo requests from 192.168.1.101 to 10.0.0.254. The bottom window is a terminal window titled "Wireshark · Data (data.data) · icmp_lab.pcap" showing the recovered flag and file hashes:

```
:  
THM{1icmp_3ch0_3xfiltr4t10n_succ3ss}  
---  
File Hashes:  
secret.txt: 9c8f3e2b7a1d4e5f6c7b8a9d0e1f2a3b  
backup.tar.
```

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.