Nulltech
Purple team bootcamp

Lateral Movement DNS(x-filtrate)

Prepared By:
Kazim Ali Obad

Supervisor:
Anmar Mohammed
Mohammed baqer

Contents

Where We Are in the Attack Chain

when an attacker targets an organization, the process goes through stages. We first stage Reconnaissance , in Recon, the attacker mostly uses HTTP. After Recon, what happens? The attacker gets into the system. After he gets Initial Access, the next step is Lateral Movement. Lateral Movement means after he gets that first access, he starts moving. For example, if the web server is the first machine he hits, he then moves to the other servers inside the organization. And we said this mostly happens through SMB. Why SMB? Because SMB is for file sharing, and through file sharing he can move around.

Why does the attacker use DNS? After the attacker does the compromise, gets Initial Access, and moves laterally, he starts exfiltrating the organization's data sending it to his external server. We call that server the C2 Command and Control. And today we're going to talk about how we extract the objects and Indicators of Compromise related to DNS, specifically around C2 communication.

DNS: The Basics

What is DNS?

DNS stands for Domain Name System. DNS is basically like your phone's contact book. Think about it in your phone you save someone's name instead of memorizing their number. Computers always talk in IP addresses. So DNS is what maps the names to those numbers. You type a name like facebook.com, DNS figures out the IP behind it and takes you there.

So when you type something like canva.com into your browser, there is an IP address behind it. The A Record is what gives you that IP let's say it comes back as 109.10.1. something. That's the A Record.

DNS Record Types

1. **A Record**

This maps the domain name to an IPv4 address. You type the name, it gives you the IP.

2. **AAAA Record**

Same idea but for IPv6. If you want to know the IPv6 address of a domain, this is the record that gives it to you.

3. **CNAME Record — the Alias**

CNAME is short for Canonical Name. We call it an alias a nickname. Sometimes you have two domains and you want both of them to point to the same place. For example, if I have cyber.com as my main domain, and I want www.cyber.com to also go to the same page I make a CNAME. So www.cyber.com is the CNAME (alias) pointing to cyber.com, the real domain.

if you type fb.com it opens Facebook. If you type facebook.com it also opens Facebook. facebook.com is the real domain, fb.com is the CNAME pointing to it.

4. **NS Record**

This tells you which name server is responsible for the domain. Who is managing this domain's records? That's what the NS record answers.

5. **MX Record — Mail Exchange**

This tells email where to go. If you send an email to support@tesla.com, the MX record decides which mail server should receive it.

6. **PTR Record — Pointer**

This is the reverse of the A Record. With an A Record: you give a domain name, you get back an IP. With a PTR Record: you give an IP, you get back the hostname. It's a pointer going the other direction.

7. **SRV Record — Service Locator**

SRV tells you where a specific service lives. For example, VoIP , Voice over IP. VoIP is how phones work inside companies, or apps like WhatsApp any voice call over the internet is VoIP. The SRV record says: here is this service, here is its type.

8. **TXT Record**

TXT stores text information. It's commonly used to hold SPF and DKIM data Those are stored in TXT records. This record is very important from a security perspective, and it's also one that attackers abuse heavily for DNS tunneling more on that in a moment.

How DNS Works

When you type a domain into your browser, the browser asks its local DNS server: "Who is the IP for this domain?" The DNS server responds with the IP address. The browser then uses that IP to connect to the server and load the page. Simple as that.

DNS Protocol — UDP vs. TCP

DNS uses UDP on Port 53. Now here is an important point that people often miss DNS usually works over UDP, but there are two situations where it switches to TCP:

- When the response size exceeds 512 bytes.
- During a Zone Transfer.

> *Note : DNS runs on UDP, Port 53 — but it switches to TCP when the response is bigger than 512 bytes, or when a Zone Transfer is happening.*

Zone Transfer

Zone Transfer is the process of copying DNS records , the A records, AAAA records, all of them , from one DNS server to another. DNS Server it has all the records, and we move them over to DNS Server 2. This process is called a Zone Transfer this happens over TCP.

Organizations have multiple DNS zones is to make management easier. You break the DNS namespace into zones and manage each one from one place in a hierarchical tree structure, with a root domain at the top.

How Attackers Use DNS for C2 Communication Why DNS?

The attacker after the compromise after he gets Initial Access and does Lateral Movement starts exfiltrating the organization's data to his external server. Now the question is: why DNS specifically? Two reasons:

First: most networks do not block outbound DNS traffic at the firewall level. Almost every organization allows DNS out, because without it nothing works.

Second: attackers can embed data inside TXT records or NULL records instead of their normal purpose. Instead of storing SPF, they load those records with stolen data.

And even if you do deny outbound DNS at the firewall, the attacker can use an internal DNS server as a relay ,The point is: DNS tunneling is common because most networks don't block DNS, and the records themselves can carry data.

What the Attack Looks Like

The attacker's malware makes DNS queries. But instead of querying for a real website, the subdomain part of the query is actually stolen data encoded as a hash. Something like: a1b2c3d4e5f6.attacker-c2.com. That hash in front is the data being smuggled out to the C2 server. It looks like a DNS lookup. It goes through the network as a DNS lookup. But it is exfiltration.

Extracting DNS Indicators of Compromise (IOC)

The Scenario

> **Scenario:** *Someone from the SOC or Network Security team tells us: one of our machines made more than 10,000 DNS queries in a single 24-hour period. And the domains being queried are ones we've never used before. That's suspicious. How do we investigate?*

Step 1 — Get the File

First thing: I ask the team what time window this happened. They tell me. I say: give me the full 24-hour file. I'm going to use PassiveDNS.

I run PassiveDNS against the capture file and I tell it: write your output here, name it dns_out.log. PassiveDNS starts running and produces readable output.

```
Sudo passivedns -r tunneling.pcap -l dnsout.log
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo passivedns -r tunneling.pcap -l dnsout.log
[sudo] password for kali:

[*] PassiveDNS 1.2.1
[*] By Edward Bjarte Fjellskål <edward.fjellskaal@gmail.com>
[*] Using libpcap version 1.10.5 (with TPACKET_V3)
[*] Using ldns version 1.8.4
[*] Reading from file tunneling.pcap
```

```
-- Total DNS records allocated            :      103262
-- Total DNS assets allocated             :       55164
-- Total DNS packets over IPv4/TCP        :           0
-- Total DNS packets over IPv6/TCP        :           0
-- Total DNS packets over TCP decoded     :           0
-- Total DNS packets over TCP failed      :           0
-- Total DNS packets over IPv4/UDP        :      173693
-- Total DNS packets over IPv6/UDP        :           0
-- Total DNS packets over UDP decoded     :      173693
-- Total DNS packets over UDP failed      :           0
-- Total packets received from libpcap    :      347495
-- Total Ethernet packets received        :      347495
-- Total VLAN packets received            :           0

[*] passivedns ended.
```

Step 2 — Find the Top Domains

Say I see more than 10,362 results come out. What do I care about? I want to see which domains are being queried the most. The domain column after we read the log header using (cat command) and count the fields turns out to be column 9. So I run:

```
awk '{print $9}' dnsout.log | sort | uniq -c | sort -nr | head
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ awk -F'|' '{print $9}' dnsout.log | sort | uniq -c | sort -nr | head

     45 d2h67oheeuigaw.cloudfront.net.
     40 d1j5ogajzvcx9c.cloudfront.net.
     12 api.snapcraft.io.
      8 security.ubuntu.com.
      6 openshift.gnome.org.
      6 download.opensuse.org.
      6 connectivity-check.ubuntu.com.
      4 proxy-nue.opensuse.org.
      4 ppa.launchpad.net.
      4 p2.shared.global.fastly.net.
```

This gives me each unique domain with a count of how many times it was queried, sorted from most to least. What came out? The vast majority of communication was going to Cisco-related domains specifically subdomains of something like cisco-update.com.

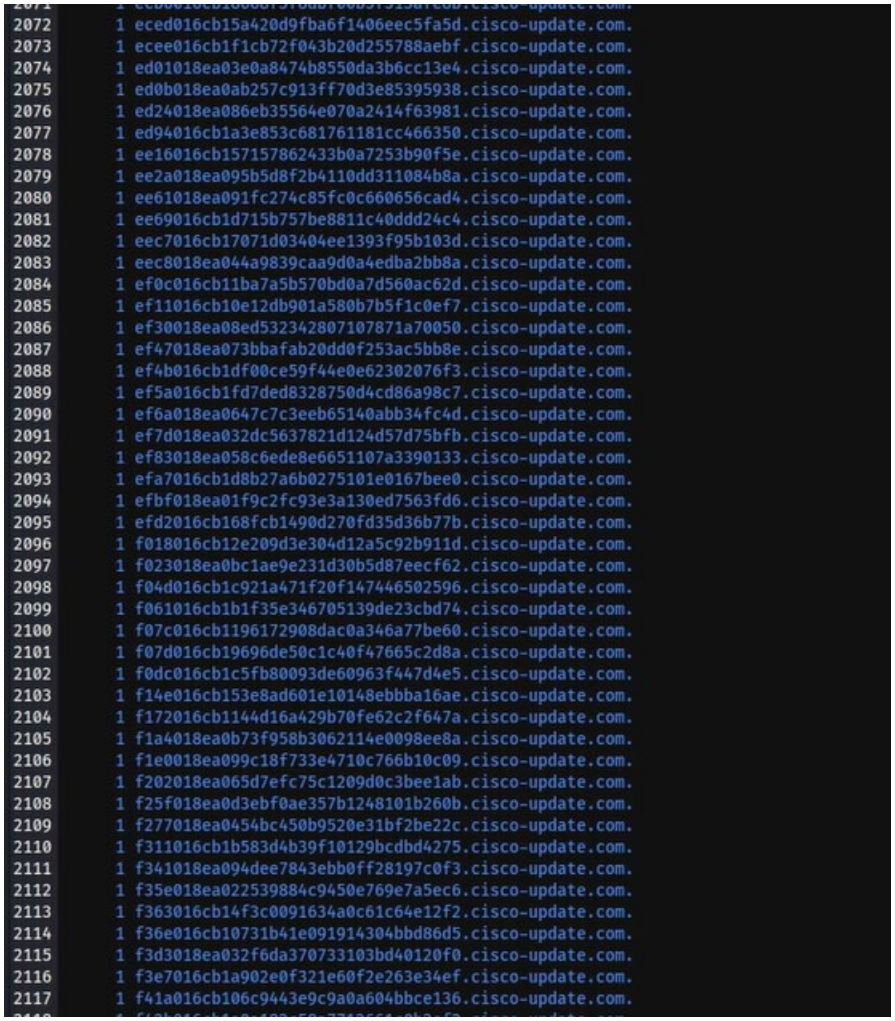
```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat dnsout.log | awk -F '|' '{print $9}' | sort | uniq -c | sort
```

cat dnsout.log | awk -F '|' '{print $9}' | sort |uniq -c | sort

```
  1 1714016cb1e75888493f99ee74b32d4ab6.cisco-update.com.
  1 1715018ea0fee97714772b2c6c501669d4.cisco-update.com.
  1 1717016cb135a2e6ff756e3c4e64bbaeca.cisco-update.com.
  1 171c016cb18ed913025d8c45e7401b8b93.cisco-update.com.
  1 171c016cb1c59018f128e4d70ae7386abc.cisco-update.com.
  1 171d016cb19ff34d9d3813461a5e379140.cisco-update.com.
  1 171d018ea084a2b568b8393e7e0301e146.cisco-update.com.
  1 171d018ea0895299d76ef2c75dbeb10e6d.cisco-update.com.
  1 171f018ea0956455e45d6c19eee4e89168.cisco-update.com.
  1 1720018ea0f7a2b33dd5b1198affa55629.cisco-update.com.
  1 1721016cb1a2ab9db71322768c996d2ab5.cisco-update.com.
  1 1722018ea0b12dd1cb0d458aa6db99e5f2.cisco-update.com.
  1 1723016cb193275420de6b4f898e292eac.cisco-update.com.
  1 1723018ea02983c57c1adb721ecf7a81e2.cisco-update.com.
  1 1724016cb1ad081c345f1429c8bfcfa049.cisco-update.com.
  1 1725018ea04995c33842b9ccf20059564e.cisco-update.com.
  1 1726018ea0d0f37a5f9b286ee2ee6b6d39.cisco-update.com.
  1 1727016cb1a9333c47c64538db72b8a691.cisco-update.com.
  1 1728016cb199d94046af6624994fdfecc6.cisco-update.com.
  1 1728016cb1c1e797a3dac21c3e731b6fb2.cisco-update.com.
  1 1728018ea0278cc5edbd71de4c21afe73b.cisco-update.com.
  1 172c016cb134d84628019a0512d96230ab.cisco-update.com.
  1 172c016cb179c638ceb44e2c9cda9feabc.cisco-update.com.
```

Now look at those subdomains. Their beginning the subdomain prefix looks like a hash. Not a readable word. That is suspicious. A normal domain subdomain looks like mail.company.com or support.company.com. Not a42f8b19c.company.com. When you see hash-looking prefixes, that's your first signal.

```
cat dnsout.log | awk -F  '|' '{print $9}' | sort |uniq -c | sort >> dns.txt
```



To same result as txt file for further use and investigate

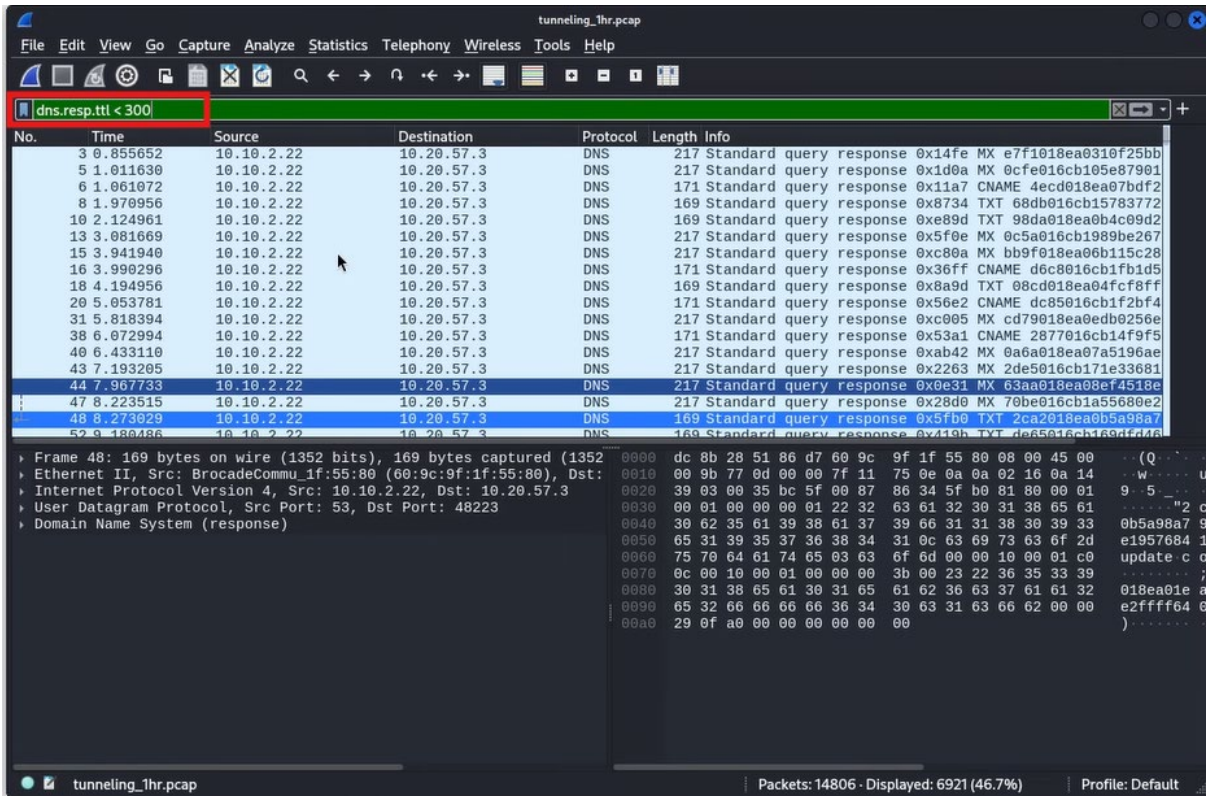
IoC 1 - TTL Less Than 300

Open the results in Wireshark. Apply this filter:

```
dns.resp.ttl < 300
```

What is TTL? TTL is Time To Live it tells any caching device how long it can keep the IP address for a domain before it needs to ask again. Normal TTL values are 300 and above that's 5 minutes or more, up to 24 hours.

So if I say TTL 300, I'm telling browsers and resolvers: keep this IP for 5 minutes, then ask again. Normal.



What Does Low TTL Mean?

If the TTL is below 300 say 60, or even lower what does that mean?

It means the IP is changing very frequently. The attacker wants the IP to change every minute. Why? Because if your security team blocks the IP, within 60 seconds it's already pointing to a different IP. This technique is called Fast Flux.

> *Fast Flux: the attacker rotates the IP address behind the same domain name very quickly. He runs the malware through many compromised hosts that act as proxies. So the domain stays the same, but the IP keeps rotating making it very hard to block.*

TTL 60 = IP changes every 60 seconds. TTL 120 = IP changes every 2 minutes. Normal is 300 and above 5 minutes or more. If you see below 300, that is not normal. That's our indicator.

We checked in Wireshark. The results came back: TTL values of 58 seconds on those Cisco-related queries. Confirmed Fast Flux is active. C2 communication is happening.

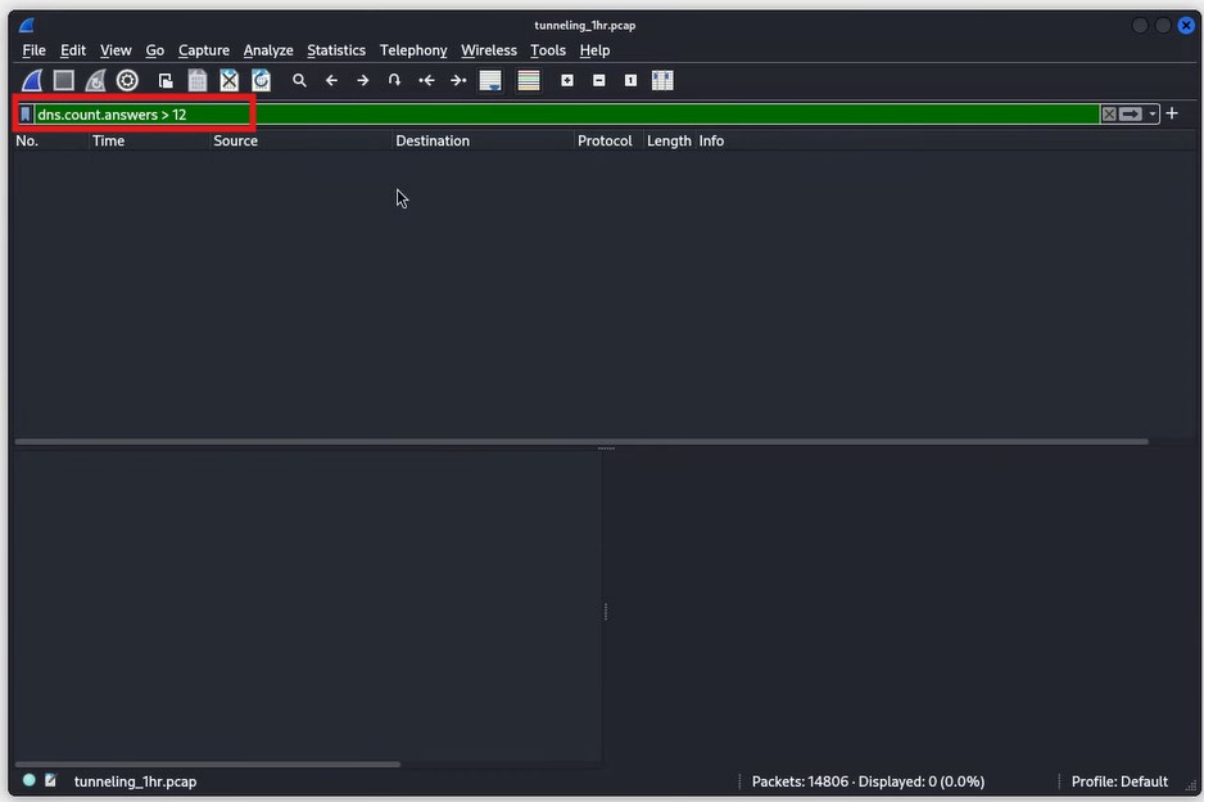
IoC 2 — DNS Answer Count Greater Than 12

A normal DNS response has at most 12 answer records in a single packet. If you see a DNS response with more than 12 answers, that's not normal traffic. That indicates a more complex C2 setup where the attacker is returning many proxy IPs at once.

Apply this Wireshark filter:

```
dns.count.answers > 12
```

this filter returned nothing. Zero packets matched. What does that tell us? It tells us the C2 communication exists — we already confirmed that with the TTL — but it's not a complex or sophisticated setup. It's a simpler attack, easier to detect.



So: Fast Flux is there, but it's not a heavy multi-node proxy network. The attacker is doing DNS exfiltration but in a straightforward way.

The Three Checklist

So how do we confirm we have C2 communication? We look for these three things:

1. High volume of DNS queries from one internal IP to domains we've never seen thousands of queries in a short window.
2. Subdomain prefixes that look like hashes not readable names.
3. TTL below 300 seconds Fast Flux is active.

SMB: Server Message Block

What is SMB?

SMB stands for Server Message Block. It's a protocol for sharing files across a network. SMB can share files, folders, and printers. And the attacker, after he gets Initial Access, uses SMB to move from that first machine to other servers inside the organization.

SMB Versions and Ports
SMB v1 — NetBIOS

The old version. It used something called NetBIOS over TCP ports 137, 138, and 139. There are a lot of vulnerabilities in NetBIOS and SMB v1, which is why most organizations have moved away from it. Some still run it, but most have switched.

SMB v2 — Port 445

SMB v2 goes directly over TCP on port 445 no NetBIOS needed. This is the version used in Windows 10 and Windows Server 2016 and later.

SMB v3 (v3.1.1) — Port 445

Same port 445. This is what modern Windows environments run

> *Why did we move away from NetBIOS and SMB v1? Because red teamers and researchers found many attacks possible through it. There are still some organizations that haven't migrated, but most have moved to v2/v3.*

Attacker Techniques on SMB

Pass the Hash

Windows stores passwords as hashes. The attacker doesn't need to know your actual password. He just needs your hash. With a hash, he can authenticate to another machine without ever cracking the password. Tools like Mimikatz and secretsdump.py are commonly used for this.

1. **PsExec**

PsExec is a legitimate Microsoft tool it comes from Microsoft's Sysinternals suite. Its purpose is to run commands on remote machines over SMB. The attacker deploys it onto the target device and uses it to install services and execute commands remotely. It communicates using SMB and DCERPC (remote procedure calls) to create and manage services on the remote host.

2. **Administrative Shares**

Windows has hidden administrative shares that always exist C$ and ADMIN$. These are for administrative access to the root of drives. The attacker uses these to browse the filesystem and drop or retrieve files. They're built-in to Windows, which makes them a reliable entry point.

3. **Data Exfiltration via SMB**

After lateral movement, the attacker can use SMB itself to transfer files from internal shared directories out to an external server. Same protocol the organization uses internally, now used to steal data.

Investigating an SMB Lateral Movement Incident

The Scenario

> **Scenario:** *An IDS alert comes in. The signature says: DCERPC Remote Service Control Manager Access. A user attempted privilege escalation they took permissions higher than they should have. Logged action: suspicious. We need to investigate.*

The correct process is to investigate:

1. Ask the team: what time window? When did the IDS fire?
2. Request the PCAP file for that period from the network team.
3. Distil the file into time segments before you do anything else.
4. Establish a baseline — top IPs, top protocols.
5. Then start extracting Indicators of Compromise.

Splitting the File

Usually we start with 4-hour segments. Why? Because 4 hours gives you a wide enough view to understand the impact. After you look at those 4 hours, you can then narrow down maybe 15 minutes right around the exact moment the attack happened.

Also: find out when exactly the Incident Response team stopped the attack. Say the attack started Monday at 4pm and IR stopped it Tuesday at 4pm that's 24 hours. Your PCAP window is 24 hours. But if IR stopped it in 1 hour, your window is 1 hour.

Zeek — SMB File Analysis

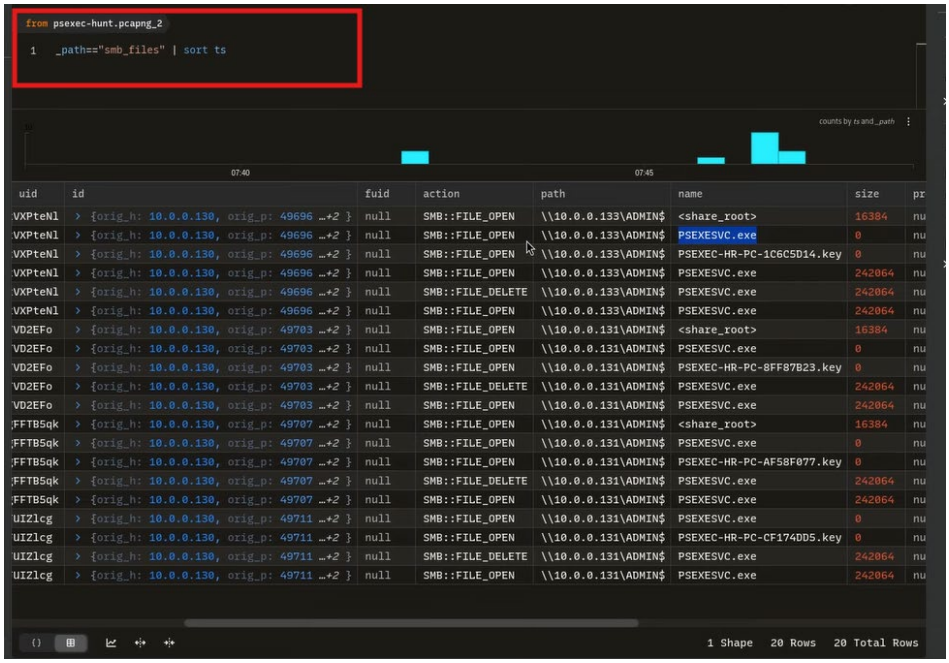
Open the file in Zeek. Import the PCAP. Zeek will start parsing everything.

Filter: SMB Files

In Zeek, filter on smb_files.path to list every file that was opened, created, read, or deleted over SMB during this window. What did we find in our exercise?

- One file was accessed through the ADMIN$ share the administrative share. Classic sign of PsExec.
- The file name was PsExec itself (psexesvc.exe). That confirms PsExec was deployed on the target.
- Another file was opened and then deleted shortly after. The attacker cleaned up.

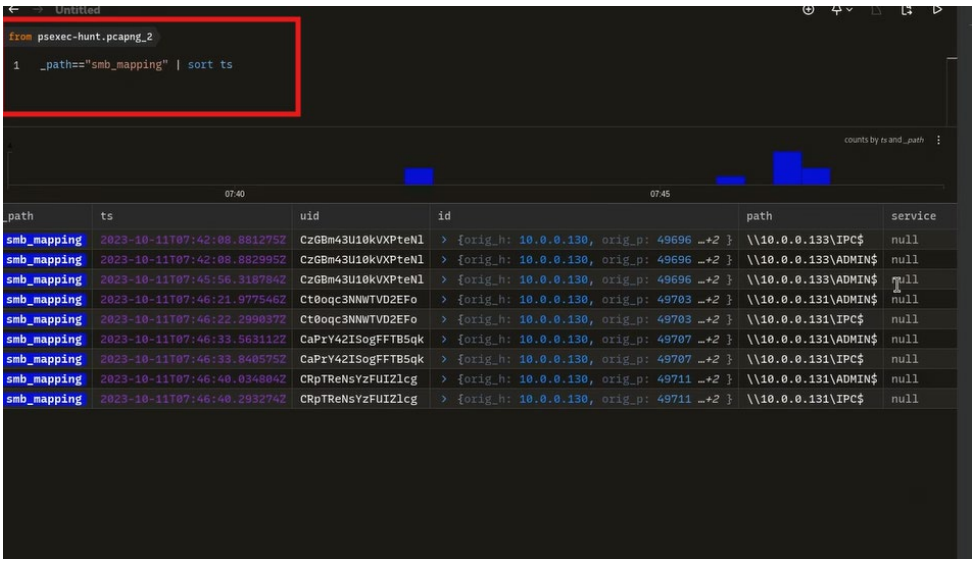
```
_path == "smb_files" | sort ts
```



Filter: Mapping the Access Paths

Remove the path restriction and run the smb_files filter broadly. Look at all directories the attacker touched. In our case: the attacker was repeatedly going into ADMIN$ and sub-paths inside it. And we saw the access came from two different internal IP addresses something like .31 and .133.

```
_path == "smb_maping" | sort ts
```



Filter: DCERPC — Remote Service Creation

This shows every service that was started, stopped, queried, or created remotely. For each entry you can see: what service ran, the source IP, the destination port,

and the action taken. This is where you confirm what PsExec actually did on the remote machine.

And this maps directly to a MITRE ATT&CK technique — T1021, Remote Services. If you go to the MITRE ATT&CK page for this technique, it shows you: how detection works, which groups use it, and what mitigations to apply. That's your reference.