REPORT
Enterprise IPsec VPN Troubleshooting Guide

v1.0.0

Author:
Eldon Gabriel

August 3, 2025

SECTION 0.0 PORTFOLIO OVERVIEW

0.1 Secure Enterprise VPN Deployment & Diagnostics

This report documents how I built a dynamic network for a medium-sized organization using Cisco Packet Tracer v8.2.2. I followed cybersecurity best practices and documented everything step-by-step.

I completed a network topology based on the **MSAF – System Administration Fundamentals** course exercise:

> ***Use Cisco Packet Tracer to create a dynamic network for a medium-size organization of 1,000 employees and employ cybersecurity best practices***.

I created this guide as an extra project portfolio activity so I could further my understanding. Plus, apply system administration and network security skills.

The goal was to design a network that uses secure routing, VLANs, and an IPSec tunnel between two networks. I also made sure internet access and traffic segmentation remained intact.

Specifications

The project included the following setup and configurations:

- Installed Cisco Packet Tracer v8.2.2
- Set up a firewall and a multilayer switch on separate subnets
- Created VLAN 1 with two servers and VLAN 2 with two workstations
- Deployed both VLANs behind the switch
- Built a DMZ with a simple web server
- Configured a working IPsec site-to-site VPN
- Set up a secure guest Wi-Fi using a separate switch and router
- Connected the internal networks to the internet using an ISP router

Validation Steps

To make sure everything worked:

- All device links showed green (active) status
- Ran traceroute to confirm the IPsec tunnel was working
- Verified VLAN-to-VLAN communication across the network using traceroute

Learning Outcome

This project helped me go from theory to applied skills. I resolved real-world simulation issues—such as configuration persistence, NAT behavior, and crypto map validation—and reinforced my ability to:

- Design secure network topologies
- Troubleshoot misconfigurations and broken tunnels
- Produce structured, technical documentation with command validation

This work lines up with the **Advanced Beginner** level of the ASD Cyber Skills Framework. It shows that I can:

- Write clear and structured technical documentation
- Spot the limits of the tools I'm working with
- Prove my configurations work through testing and command-line outputs

---

Disclaimer: This is an independent project using the skills and tools taught in the MSAF – System Administration Fundamentals course at Mosse Cyber Security Institute. While it builds on core topics from the curriculum, the full lab design, configurations, and troubleshooting documented here were created outside the scope of any official assignment.

In line with my academic pledge, I do not publish or share official course solutions. However, I may share this guide on my portfolio website, which includes certificates and project reports like this one for academic or professional review.

TABLE OF CONTENTS

SECTION 0.0 PORTFOLIO OVERVIEW....1
TABLE OF CONTENTS....3
REVISION HISTORY....5
SECTION 1.0 INTRODUCTION AND SETUP....6
1.1 Project Description....6
1.2 Network Topology Mapping....7
1.2.1 Network Segments....10
1.3 Comprehensive Configuration Steps....11
1.3.1 Initial Setup and Licensing Activation....11
1.3.1.1 Host Static IP Configuration....11
1.3.1.2 Interface IP Assignment and Device Role....12
1.3.1.3 Security License Activation (Optional)....16
1.3.2 Interface Configuration and Static Routing....17
1.3.2.1 TO-FW-EX - Toronto Firewall....18
1.3.2.2 NY-FW-EX - New York Firewall....19
1.3.2.3 ISP-Router - Internet Transit Device....20
1.3.3 NAT Configuration and VPN Traffic Exemption....21
1.3.3.1 TO-FW-EX - NAT and VPN Bypass Rules....22
1.3.3.2 NY-FW-EX - NAT and VPN Bypass Rules....23
1.3.4 ISAKMP (Phase 1) Policy Configuration....24
1.3.4.1 TO-FW-EX - ISAKMP Configuration....25
1.3.4.2 NY-FW-EX - ISAKMP Configuration....26
1.3.5 Crypto IPSec (Phase 2) Transform-Set Configuration....27
1.3.5.1 TO-FW-EX - IPSec Settings....28
1.3.5.2 NY-FW-EX - IPSec Settings....29
1.3.6 Define ACL for Interesting Traffic....30
1.3.7 Crypto Map Definition and Interface Binding....32
1.4 VPN Deployment Challenges and Troubleshooting Log....34
1.4.1 CLI Input Errors and Syntax Failures....37
1.4.2 Non-Persistence Configuration Across Reboots....37
1.4.2.1 Workaround: Full Block Paste via CLI....38
1.4.3 ISAKMP Phase 1 Negotiation Failures....39
1.4.4 IPsec Phase 2 Encapsulation Failures....39
1.4.5 Unrelated Local LAN Connectivity Issue During Testing....39
1.4.6 Key Lessons and Future Recommendations....40
1.5 Tunnel Traffic Initiation and Final Checks....41
1.5.1 Tunnel Verification Commands....42
SECTION 2.0 VPN TUNNEL SYMMETRY and DIAGNOSTICS....44
2.1 Critical Tunnel Matching Guidance....44
SECTION 3.0 CONCLUSION....47

3.1 Final VPN Status Verification and Wrap-Up.....47
3.1.1 Traceroute Verification..... 47
3.1.2 Validation via show crypto Commands.....48

REVISION HISTORY

Version	Date	Author	Description of Changes
v1.0.0	08/3/2025	Eldon G.	Initial draft.

SECTION 1.0 INTRODUCTION AND SETUP

1.1 Project Description

This report documents how I built a dynamic network using cybersecurity best practices. The network connects 1,000 users across two sites—Toronto and New York. I used **Cisco Packet Tracer v8.2.2** to simulate the full setup.

The main goal was to create a secure site-to-site IPSec VPN tunnel. While the configuration seemed simple at first, the project turned into a deeper learning experience. I ran into unexpected problems with protocol behavior, tool limitations, and human error. Each challenge helped me think more like a system administrator and network troubleshooter.

Packet Tracer was useful for learning, but it sometimes gave confusing results during advanced setups. I learned how to tell the difference between a real misconfiguration and a simulator issue. Many problems weren't my fault—they were caused by how Packet Tracer handles certain protocols.

This project helped me develop a key networking skill: working through problems step-by-step. I used CLI commands, traceroute, and controlled traffic tests to confirm when my setup was working, even when the GUI didn't reflect it. In the end, the process taught me how to troubleshoot and validate configurations with a critical mindset.

1.2 Network Topology Mapping

The implemented topology consists of three primary routing devices:

- **TO-FW-EX** (Toronto Firewall)
- **ISP-Router** (Internet Provider)
- **NY-FW-EX** (New York Firewall)

These devices are linked together to build a secure site-to-site IPSec VPN tunnel. The ISP-Router acts as a simulated public internet. The Toronto office has several internal networks that connect to the New York network.

Key Network Components & Interconnections

ISP Server (Internet Simulator)

- **Device Model:** Server-PT
- **Role:** Simulates external public Internet services
- **Interfaces:**

Interface	IP Address	Connection
FastEthernet0	8.8.8.1 /30	Connects to ISP-Router Gig2/0

ISP-Router (Internet Backbone Router)

- **Device Model:** PT-Empty
- **Role:** Simulates the public internet, routing traffic between Toronto and New York edge routers
- **Interfaces:**

Interface	IP Address	Connected Device
GigabitEthernet0/0	203.0.113.1/30	NY-FW-EX's Gig0/0/1
GigabitEthernet1/0	203.0.113.5/30	TO-FW-EX's Gig0/0/0

TO-FW-EX (Toronto Firewall/Edge Router)

- **Device Model:** ISR4331
- **Role:** Edge router for the Toronto site, VPN endpoint
- **Interfaces:**

Interface	IP Address	Connected Device
GigabitEthernet0/0	203.0.113.6/30 (External)	ISP-Router Gig1/0
GigabitEthernet0/1	10.0.0.2/30 (Internal)	MSW-TO-L3 Gig0/2

MSW-TO-L3 (Toronto Multi-Layer Switch)

Device Model: 3560 24PS

- **Role:** Routes internal Toronto VLANs
- **Note:** Guest VLAN is routed via TO-Guest-Rtr for security segmentation
- **VLAN Configuration:**

VLAN ID	NAME	Subnet	Ports
10	Workstation	192.168.10.0/24	Fa0/7, Fa0/8
20	Server Closet	192.168.20.0/24	Fa0/3, Fa/04
30	DMZ	192.168.30.0/24	Fa0/9
40	Guest Wi-Fi	192.168.41.0/24	Fa0/10
50	Reserved	192.168.50.0/24	*Future Use

> ***Note:*** *VLAN 50 is reserved for future segmentation, such as a secure wireless network for a project-specific team.*

NY-FW-EX (New York Firewall/Edge Router)

- **Device Model:** ISR4331
- **Role:** New York edge router and VPN endpoint
- **Interfaces:**

Interface	IP Address	Connection Device	Network
GigabitEthernet0/0/0	192.168.2.1/24	MSW-NY-L3 Fa0/1	NY Internal LAN
GigabitEthernet0/0/1	203.0.113.2/30	ISP-Router Gig0/0	NY WAN

1.2.1 Network Segments

Toronto Internal LANs (behind `MSW-TO-L3`)

VLAN ID	VLAN Name	Device Name	IP Address	Subnet	Default Gateway
10	Workstation	TO-PC-1	192.168.10.10 /24	192.168.10.0 /24	192.168.10.1
10	Workstation	TO-PC-2	192.168.10.11 /24	192.168.10.0 /24	192.168.10.1
20	Server Closet	TO-Server1	192.168.20.10 /24	192.168.20.0 /24	192.168.20.1
20	Server Closet	TO-Server2	192.168.20.11 /24	192.168.20.0 /24	192.168.20.1
30	DMZ	TO-Web-Server	192.168.30.10 /24	192.168.30.0 /24	192.168.30.1
40	Guest Wi-Fi	TO-Guest-Rtr	192.168.41.2 /24	192.168.41.0 /24	*N/A
40	Guest Wi-Fi	TO-Guest-SW	*N/A	*N/A	*N/A
40	Guest Wi-Fi	TO-Guest-AP	*N/A	*N/A	*N/A
40	Guest Wi-Fi	TO-Guest-PC	192.168.41.3 /24	192.168.41.0 /24	192.168.41.2
50	Reserved	*N/A	*N/A	*N/A	*N/A

Note: VLAN 50 is reserved for future segmentation and is currently unused. Devices marked *N/A do not have IP configurations, as they are Layer 2 infrastructure or unassigned.

New York Internal LAN

Network Segment	Subnet	Connected Devices
TO Internal Link	10.0.0.0/30	TO-FW-EX and MSW-TO-L3
NY Internal LAN	192.168.2.0/24	NY-FW-EX (Gig0/0/0), NY-PC-1
TO Public WAN Segment	203.0.113.4/30	TO-FW-EX and ISP-Router
NY Public WAN Segment	203.0.113.0/30	NY-FW-EX and ISP-Router

IPsec VPN Tunnel Parameters

VPN Endpoint (Local)	VPN Endpoint (Remote)	Networks Secured (Interesting Traffic)
`NY-FW-EX` (203.0.113.2)	`TO-FW-EX` (203.0.113.6)	**Local:** 192.168.2.0/24 **Remote:** 192.168.X.0/24 (All TO subnets)

1.3 Comprehensive Configuration Steps

This section outlines the core setup procedures completed across all routing devices. It covers physical topology validation, static IP assignments and interface configuration, NAT and VPN ACL design, and license activation.

1.3.1 Initial Setup and Licensing Activation

This step ensured all devices were connected according to the logical diagram. This prepared the routers for advanced configuration tasks such as NAT, VPN, and licensing (where applicable).

1.3.1.1 Host Static IP Configuration

Next, I configured the static IP addresses and default gateways for the host machines. This ensures they can communicate within their respective local networks and reach their gateway.

- **TO-PC:** `192.168.10.100/24` → Gateway of `192.168.10.1`
- **NY-PC:** `192.168.2.100/24` → Gateway of `192.168.2.1`

These settings enable local LAN connectivity. However, initial cross-site ping attempts failed due to improper NAT behavior. I later traced this to a hidden `no ip cef` directive, which I removed to restore correct NAT functionality.

1.3.1.2 Interface IP Assignment and Device Role

All three routers were configured with interface addresses, basic NAT setup, static routes, and ACLs to exempt VPN-bound traffic. Below is a summary of each device's role and commands used.

> ***Note:*** *For more reliable results in Cisco Packet Tracer, all configuration steps in Section 1.3 were ultimately consolidated and applied using full configuration block pastes. See Section 1.4.2.1 for details on this workaround.*

TO-FW-EX Configuration (Toronto Edge)

cli

```
configure terminal or conf t
hostname TO-FW-EX

interface GigabitEthernet0/0/0
ip address 203.0.113.6 255.255.255.252
ip nat outside
no shutdown
exit

interface GigabitEthernet0/0/1
ip address 10.0.0.2 255.255.255.252
ip nat inside
no shutdown
exit

! Routing
ip route 0.0.0.0 0.0.0.0 203.0.113.5
ip route 192.168.10.0 255.255.255.0 10.0.0.1
ip route 192.168.20.0 255.255.255.0 10.0.0.1
ip route 192.168.30.0 255.255.255.0 10.0.0.1
ip route 192.168.41.0 255.255.255.0 10.0.0.1
ip route 192.168.50.0 255.255.255.0 10.0.0.1

! ACL for VPN traffic exemption
ip access-list extended NO_NAT_VPN_TRAFFIC_TO
deny ip 192.168.10.0 0.0.0.255 192.168.2.0 0.0.0.255
deny ip 192.168.20.0 0.0.0.255 192.168.2.0 0.0.0.255
deny ip 192.168.30.0 0.0.0.255 192.168.2.0 0.0.0.255
deny ip 192.168.41.0 0.0.0.255 192.168.2.0 0.0.0.255
deny ip 192.168.50.0 0.0.0.255 192.168.2.0 0.0.0.255
permit ip any any

! NAT with VPN exemption
no ip nat inside source list 100 interface GigabitEthernet0/0/0 overload
ip nat inside source list 100 interface GigabitEthernet0/0/0 overload

end
write memory
```

ISP-Router Configuration (Transit Node)

```
cli

configure terminal or conf t
hostname ISP-Router

interface GigabitEthernet0/0
 ip address 203.0.113.1 255.255.255.252
 no shutdown
exit

interface GigabitEthernet0/1
 ip address 203.0.113.5 255.255.255.252
 no shut
exit
```

NY-FW-EX Configuration (New York Edge)

```
cli

configure terminal or conf t
hostname NY-FW-EX

! Interfaces
interface GigabitEthernet0/0/0
 ip address 192.168.2.1 255.255.255.0
 ip nat inside
 no shutdown
exit

interface GigabitEthernet0/0/1
 ip address 203.0.113.2 255.255.255.252
 ip nat outside
 no shutdown
exit

! Routing
ip route 0.0.0.0 0.0.0.0 203.0.113.1


! ACL for VPN exemption

! For VPN traffic not to be NAT'd:
ip access-list extended NO_NAT_VPN_TRAFFIC
 deny ip 192.168.2.0 0.0.0.255 192.168.10.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.20.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.30.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.41.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.50.0 0.0.0.255
 permit ip any any

! NAT with VPN exemption
no ip nat inside source list 100 interface GigabitEthernet0/0/1 overload
ip nat inside source list NO_NAT_VPN_TRAFFIC interface
GigabitEthernet0/0/1 overload

end
write memory
```

1.3.1.3 Security License Activation (Optional)

In production environments using Cisco ISR routers, VPN features require a security license. While Cisco Packet Tracer assumes pre-licensed routers, these are the real-world steps.

```
cli

! Verify current license
status show version | include license

! Activate the securityk9 technology package
license boot module ISR4300 technology-package securityk9

copy run start
reload

! Post-reboot verification
show version | include license
```

> *Note: Replace* `ISR4300` *with your router's actual model from* `show version` *if different.*

1.3.2 Interface Configuration and Static Routing

This section documents the essential interface and routing configurations applied to the core network devices: Toronto (`TO-FW-EX`), New York (`NY-FW-EX`), and the `ISP-Router`. The settings enable IP reachability, prepare the network for NAT, VPN operations, and establish static routes to ensure proper end-to-end packet delivery.

1.3.2.1 TO-FW-EX - Toronto Firewall

This manages LAN traffic routing, NAT, and VPN termination for the Toronto network.

```
cli

configure terminal or conf t

! --- Interface Configuration ---
interface GigabitEthernet0/0/0
 ip address 203.0.113.6 255.255.255.252
 ip nat outside
 duplex auto
 speed auto
 no shutdown
exit

interface GigabitEthernet0/0/1
 ip address 10.0.0.2 255.255.255.252  router
 ip nat inside
 duplex auto
 speed auto
 no shutdown
exit



! --- Static Routing ---
ip route 0.0.0.0 0.0.0.0 203.0.113.5
ip route 192.168.10.0 255.255.255.0 10.0.0.1
ip route 192.168.20.0 255.255.255.0 10.0.0.1
ip route 192.168.30.0 255.255.255.0 10.0.0.1
ip route 192.168.41.0 255.255.255.0 10.0.0.1
ip route 192.168.50.0 255.255.255.0 10.0.0.1

end
write memory
```

1.3.2.2 NY-FW-EX - New York Firewall

This router serves as the VPN peer in New York and handles traffic from its internal LAN.

```
cli


configure terminal or conf t

! --- Interface Configuration ---
interface GigabitEthernet0/0/0
 ip address 192.168.2.1 255.255.255.0
 ip nat inside
 duplex auto
 speed auto
 no shutdown
exit

interface GigabitEthernet0/0/1
ip address 203.0.113.2 255.255.255.252
 ip nat outside
 duplex auto
 speed auto
 no shutdown
exit

! --- Static Routing ---
ip route 0.0.0.0 0.0.0.0 203.0.113.1

end
write memory
```

1.3.2.3 ISP-Router - Internet Transit Device

The ISP router acts as the bridge between the New York and Toronto networks, forwarding public traffic.

```
cli


configure terminal or conf t

! --- Interface Configuration ---
interface GigabitEthernet0/0/0
 ip address 203.0.113.1 255.255.255.252
 duplex auto
 speed auto
 no shutdown
exit

interface GigabitEthernet0/0/1
 ip address 203.0.113.5 255.255.255.252
 duplex auto
 speed auto
 no shutdown
exit

! --- Static Routing ---
ip route 192.168.2.0 255.255.255.0 203.0.113.2
ip route 192.168.10.0 255.255.255.0 203.0.113.6
ip route 192.168.20.0 255.255.255.0 203.0.113.6
ip route 192.168.30.0 255.255.255.0 203.0.113.6
ip route 192.168.41.0 255.255.255.0 203.0.113.6
ip route 192.168.50.0 255.255.255.0 203.0.113.6

end
write memory
```

1.3.3 NAT Configuration and VPN Traffic Exemption

This section outlines the NAT settings applied to both edge routers. NAT allows internal devices to access external networks using public IPs. However, for the VPN to function correctly, **specific traffic must be able to bypass NAT**. This is accomplished using extended access control lists (ACLs) to define and exempt "interesting traffic" destined for the VPN tunnel.

1.3.3.1 TO-FW-EX - NAT and VPN Bypass Rules

This router handles NAT for outbound internet access and excludes VPN-bound traffic using an external ACL.

```
cli


configure terminal

! --- Standard NAT Rule for General Internet Access ---
access-list 100 permit ip 192.168.10.0 0.0.0.255 any
access-list 100 permit ip 192.168.20.0 0.0.0.255 any
access-list 100 permit ip 192.168.30.0 0.0.0.255 any
access-list 100 permit ip 192.168.41.0 0.0.0.255 any
access-list 100 permit ip 192.168.50.0 0.0.0.255 any

! --- Extended ACL to Exempt VPN Traffic from NAT ---
ip access-list extended NO_NAT_VPN_TRAFFIC_TO
 deny ip 192.168.10.0 0.0.0.255 192.168.2.0 0.0.0.255
 deny ip 192.168.20.0 0.0.0.255 192.168.2.0 0.0.0.255
 deny ip 192.168.30.0 0.0.0.255 192.168.2.0 0.0.0.255
 deny ip 192.168.41.0 0.0.0.255 192.168.2.0 0.0.0.255
 deny ip 192.168.50.0 0.0.0.255 192.168.2.0 0.0.0.255
 permit ip any any

! Apply NAT with VPN exemption ---
ip nat inside source list NO_NAT_VPN_TRAFFIC_TO interface
GigabitEthernet0/0/0 overload

end
write memory
```

1.3.3.2 NY-FW-EX - NAT and VPN Bypass Rules

The router uses a similar approach, applying NAT for general outbound traffic while exempting VPN traffic destined for Toronto subnets.

cli

```
configure terminal

! --- Standard NAT Rule for General Internet Access ---
access-list 100 permit ip 192.168.2.0 0.0.0.255 any

! --- Extended ACL to Exempt VPN Traffic from NAT ---
ip access-list extended NO_NAT_VPN_TRAFFIC
 deny ip 192.168.2.0 0.0.0.255 192.168.10.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.20.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.30.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.41.0 0.0.0.255
 deny ip 192.168.2.0 0.0.0.255 192.168.50.0 0.0.0.255
 permit ip any any

! Apply NAT with VPN exemption ---
ip nat inside source list NO_NAT_VPN_TRAFFIC interface
GigabitEthernet0/0/1 overload

end
write memory
```

1.3.4 ISAKMP (Phase 1) Policy Configuration

This section defines **ISAKMP Phase 1 security parameters** used to establish a secure tunnel between the Toronto (TO-FW-EX) and New York (NY-FW-EX) edge routers. The ISAKMP policy ensures both devices agree on foundational key exchange settings before proceeding to IPSec Phase 2 negotiations.

Key attributes configured:

- **Encryption**: AES
- **Authentication**: Pre-shared key
- **Diffie-Hellman Group**: Group 5
- **Hashing Algorithm**: SHA
- **Lifetime**: 86400 seconds

To avoid legacy misconfigurations or simulator-induced residual states, all existing ISAKMP policies and keys were explicitly cleared before applying the updated configuration.

> ***Note on Simulator Behavior:*** *During initial testing, attempts to configure* `hash sha256` *failed. See* *Section 1.4.2.1* *for details on this workaround.*

1.3.4.1 TO-FW-EX - ISAKMP Configuration

```
cli

configure terminal

! Clear any existing Phase 1 configuration
no crypto isakmp policy 10
no crypto isakmp key MCSI_SecureVPN2025! address 203.0.113.2

! Define updated ISAKMP Policy
crypto isakmp policy 10
 encryption aes
 authentication pre-share
 group 5
 hash sha
 lifetime 86400
exit

! Define pre-shared key for New York peer
crypto isakmp key MCSI_SecureVPN_2025! address 203.0.113.2
end
```

1.3.4.2 NY-FW-EX - ISAKMP Configuration

```
cli

configure terminal

! Remove any existing Phase 1 configuration
no crypto isakmp policy 10

! Define ISAKMP Policy
crypto isakmp policy 10
 encryption aes
 authentication pre-share
 group 5
 hash sha
 lifetime 86400
exit

! Define pre-shared key for Toronto peer
crypto isakmp key MCSI_SecureVPN_2025! address 203.0.113.6
end
```

Note on Simulator Behavior: *During initial testing, attempts to configure* `hash sha256` *failed. See* [*Section 1.4.2.1*] *for details on this workaround.*

1.3.5 Crypto IPSec (Phase 2) Transform-Set Configuration

This sector defines the IPSec Phase 2 settings used to secure traffic across the VPN tunnel after it is established. The transform-set specifies how packets are encrypted and authenticated across the tunnel between Toronto (TO-FW-EX) Firewall and New York (NY-FW-EX) Firewall.

During testing, the `mode tunnel` command repeatedly failed with `% Invalid input detected at '^' marker` error continued to persist when entered line-by-line in Cisco Packet Tracer. This was due to known limitations in the simulator's command retention behavior.

> ***Workaround Implementation:*** *See Section 1.4.2.1 for details on this workaround.*

1.3.5.1 TO-FW-EX - IPSec Settings

cli

```
configure terminal or conf t

! --- Clear any existing Phase 2 configuration
interface GigabitEthernet0/0/0
no crypto isakmp policy 10
no crypto isakmp key MCSI_SecureVPN2025! address 203.0.113.2

! --- Remove existing transform-set to ensure clean config ---
no crypto ipsec transform-set MCSI-Transform2025 esp-aes esp-sha-hmac

crypto ipsec transform-set MCSI-Transform2025 esp-aes esp-sha-hmac
exit

! --- Re-apply crypto map (will be done in step 5) ---
end
```

1.3.5.2 NY-FW-EX - IPSec Settings

cli

```
configure terminal or conf t

! Remove crypto map temporarily (required to modify transform-set)
interface GigabitEthernet0/0/1
 no crypto map MCSI-Map2025
exit

! Remove existing transform-set
no crypto ipsec transform-set MCSI-Transform2025 esp-aes esp-sha-hmac

! Define transform-set with explicit tunnel mode
crypto ipsec transform-set MCSI-Transform2025 esp-aes esp-sha-hmac
 mode tunnel
exit
end
```

1.3.6 Define ACL for Interesting Traffic

This section defines the Access Control Lists (ACLs) that determine which traffic should be protected by the IPSec tunnel. Only the traffic matching these rules (often referred to as "interesting traffic") will trigger tunnel negotiation and be encapsulated by IPSec.

To simplify troubleshooting and improve readability, custom-named ACLs (`VPN_TRAFFIC_TO` and `VPN_TRAFFIC`) were used instead of default numeric ACLs like 100 or 101. These ACLs match local-to-remote IP subnets on both edge routers and are later referenced in the crypto map configuration.

> ***Note:*** *For full paste consistency and to prevent simulator conflicts, ensure any legacy ACLs (e.g., numbered 100/101) are removed prior to applying the configuration.*

TO-FW-EX (Toronto Firewall)

```
cli

configure terminal or conf t

access-list extended VPN_TRAFFIC_TO
 permit ip 192.168.10.0 0.0.0.255 192.168.2.0 0.0.0.255
 permit ip 192.168.20.0 0.0.0.255 192.168.2.0 0.0.0.255
 permit ip 192.168.30.0 0.0.0.255 192.168.2.0 0.0.0.255
 permit ip 192.168.41.0 0.0.0.255 192.168.2.0 0.0.0.255
 permit ip 192.168.50.0 0.0.0.255 192.168.2.0 0.0.0.255
end
```

NY-FW-EX (New York Firewall)

```
cli

configure terminal or conf t

access-list extended VPN_TRAFFIC
 permit ip 192.168.2.0 0.0.0.255 192.168.10.0 0.0.0.255
 permit ip 192.168.2.0 0.0.0.255 192.168.20.0 0.0.0.255
 permit ip 192.168.2.0 0.0.0.255 192.168.30.0 0.0.0.255
 permit ip 192.168.2.0 0.0.0.255 192.168.41.0 0.0.0.255
 permit ip 192.168.2.0 0.0.0.255 192.168.50.0 0.0.0.255

end
```

1.3.7 Crypto Map Definition and Interface Binding

This section defines and attaches the IPSec crypto maps that bind together all previously configured elements: ISAKMP policies (Phase 1), transform-sets (Phase 2), and ACLs for interesting traffic. The crypto map is applied directly to the appropriate WAN-facing interface on each edge device to activate IPSec tunnel negotiation and enforcement.

***Note:** Before defining the crypto map, any existing crypto map entries should be removed to prevent misconfigurations.*

TO-FW-EX (Toronto Firewall)

```
cli

configure terminal or conf t

! Remove any existing crypto map instance for a clean start
no crypto map MCSI-Map2025 10 ipsec-isakmp

! Define crypto map and bind relevant components
crypto map MCSI-Map2025 10 ipsec-isakmp
 set peer 203.0.113.2
 set pfs group5
 set security-association lifetime seconds 86400
 set transform-set MCSI-Transform2025
 match address VPN_TRAFFIC_TO
exit

! Apply the crypto map to the external interface
interface GigabitEthernet0/0/0
 crypto map MCSI-Map2025

end
write memory
```

NY-FW-EX (New York Firewall)

```
cli

configure terminal or conf t

! Remove any existing crypto map instance for a clean start
no crypto map MCSI-Map2025 10 ipsec-isakmp

! Define crypto map and bind relevant components
crypto map MCSI-Map2025 10 ipsec-isakmp
 set peer 203.0.113.6
 set pfs group5
 set security-association lifetime seconds 86400
 set transform-set MCSI-Transform2025
 match address VPN_TRAFFIC
exit

! Apply the crypto map to the external interface
interface GigabitEthernet0/0/1
 crypto map MCSI-Map2025

end
write memory
```

> ***Summary:*** *Crypto maps were applied on TO-FW-EX (Gi0/0/0) and NY-FW-EX (Gi0/0/1), using transform-set* `MCSI-Transform2025` *and ACLs* `VPN_TRAFFIC_TO` / `VPN_TRAFFIC`*.*

1.4 VPN Deployment Challenges and Troubleshooting Log

During the complex implementation and validation of the IPsec tunnel in Cisco Packet Tracer, multiple challenges arose that required persistent, methodical troubleshooting. These challenges revealed the unique complexities of working within a simulated environment:

Problem 1: Simulator Limitations and Non-Standard Configuration

Problem: The biggest challenge was having to change from real-world best practices because of simulation limits. At first, the project used ASAs (Adaptive Security Appliances) for firewalling. To allow ICMP traffic, I had to create a special access list and remove NAT on the ASAs. This workaround was needed for the simulation to work. But it made the setup less realistic. Eventually, I replaced the ASAs with routers acting as firewalls to finish the configuration.

Impact: This showed the important difference between real network design and simulation compromises. It forced me to document a non-standard setup. I learned that real networks need different, more secure methods.

Problem 2: Packet Tracer CLI and Configuration Instability

Problem: The simulation environment often behaved unpredictably. I saw many “% Invalid input detected at '^' marker” errors, even with correct commands. This meant internal parser bugs were causing problems. Also, critical commands like `mode tunnel` inside the crypto transform set were accepted but did not save in the running config. The simulator dropped or misapplied these commands silently. To fix this, I stopped entering commands line-by-line. Instead, I used full configuration blocks pasted at once to make sure all settings, including `mode tunnel`, were applied.

Impact: These issues blocked the correct VPN setup. The commands not saving stopped the tunnel from working. I spent a lot of time fixing errors caused by the simulator, not by my own mistakes. This taught me to spot tool limits and find workarounds.

Problem 3: Hidden `no ip cef` Command and NAT Troubleshooting

Problem: A major roadblock was a hidden `no ip cef` command breaking NAT silently. After setting IPs and routes right, pings to public IPs worked. But packets did not translate properly. I had to dig deep into routing and config to find and remove this `no ip cef` command. That fixed the NAT problem.

Impact: This showed how a single command, even if unrelated, can cause big network problems. It was a key moment in my troubleshooting. It taught me to review configs deeply, not just surface settings.

Problem 4: Critical Crypto Parameter Non-Persistence

Problem: Another hard issue was that advanced crypto settings—like `hash sha256`, `lifetime 86400`, and certain transform sets—did not save on both routers. They seemed accepted, but vanished from running configs. I switched to full-text block inputs for all crypto settings to get them saved.

Impact: I had to lower crypto settings (e.g., to `hash sha` and `esp-sha-hmac`) to make sure commands stayed. This showed me the importance of checking command persistence and adapting security when needed in simulators.

Problem 5: VPN Negotiation Failures and Simulator Quirks

Problem: The VPN tunnel kept failing to start Phase 1. The command `show crypto isakmp sa` showed states like `MM_NO_STATE` and `ACTIVE (deleted)`. This meant a security association (SA) was formed, but then dropped immediately. The Toronto firewall sometimes showed no ISAKMP SA entries at all.

Impact: Without a stable Phase 1, no key exchange could finish. No encrypted traffic passed. This caused `show crypto ipsec sa` to report zero packets encrypted. I had to run deep diagnostics to prove my config was correct, and the simulator was giving false status outputs.

Problem 6: Unrelated Local LAN Connectivity Issues

Problem: During testing, I found a persistent Layer 2/ARP problem on the New York router's public link. To fix this, I replaced the virtual router with a fresh one in the topology.

Impact: Though unrelated to the VPN tunnel, this problem slowed my troubleshooting. It taught me to isolate issues and not let one problem block overall progress.

1.4.1 CLI Input Errors and Syntax Failures

One of the earliest and most frequent issues I ran into was the `% Invalid input detected at '^' marker` error. This happened when I included prompt context in the command, like `(config)#`, or even when entering valid commands like `terminal length 0`.

These errors pointed to a deeper issue with how Packet Tracer handles CLI input. Some commands simply wouldn't run, even when typed correctly. That slowed down progress and forced me to re-check syntax constantly.

1.4.2 Non-Persistence Configuration Across Reboots

Another major issue came from config settings that didn’t save. Even though Packet Tracer showed no errors, some commands vanished after rebooting or switching tabs. This affected critical settings like:

- `mode tunnel` in the crypto IPsec transform-set
- `hash sha256` in the ISAKMP policy

To deal with this, I had to clear and reapply configs multiple times on both the **Toronto (TO-FW-EX) Firewall** and the **New York (NY-FW-EX) Firewall**. The problem was frustrating and time-consuming.

In some cases, I even considered rebuilding parts of the topology or replacing routers entirely. This showed just how limited simulated environments can be compared to real hardware.

1.4.2.1 Workaround: Full Block Paste via CLI

To fix the problem where commands didn't apply or stick properly, I used a full copy-paste method with pre-written configuration blocks. This made sure key settings—like `mode tunnel` in the transform-set and NAT exemption rules—were saved correctly.

What I Did:

- **Routers Affected:**
  - Toronto (TO-FW-EX) Firewall
  - New York (NY-FW-EX) Firewall
- **Problem:**
  - Manual typing caused missing or dropped commands.
- **Fix**
  - I created complete configs and pasted them into the router CLI in one go.

This method helped get around Packet Tracer bugs by avoiding line-by-line input. It stopped issues like the CLI skipping prompts or not recognizing commands. The next sections show the full configs used for both routers. These can be reused or rolled back if needed.

This approach made the tunnel setup more stable across multiple test runs.

1.4.3 ISAKMP Phase 1 Negotiation Failures

One of the main problems was that Phase 1 of the VPN tunnel wouldn't complete. When I used `show crypto isakmp sa`, one router showed short-lived states like `MM_NO_STATE` and `ACTIVE (deleted)`. This meant it was trying to build the tunnel, but the connection broke right after. The other router didn't show anything at all. It wasn't even processing the tunnel requests. Without Phase 1, the routers couldn't agree on security settings, and the tunnel couldn't start.

1.4.4 IPsec Phase 2 Encapsulation Failures

Since Phase 1 kept failing, Phase 2 couldn't work either. I ran `show crypto ipsec sa` and saw that no packets were being encrypted or sent through the tunnel. This confirmed that the VPN was down and no protected traffic was moving between sites.

1.4.5 Unrelated Local LAN Connectivity Issue During Testing

While testing the tunnel, I found a separate issue: a PC with IP `192.168.2.10` couldn't ping another PC on the same network (`192.168.2.100`). The pings failed with timeouts. This wasn't related to the VPN, but it made the troubleshooting harder. I had to rule it out before continuing with the tunnel fixes.

1.4.6 Key Lessons and Future Recommendations

This troubleshooting process highlighted several important lessons:

- Cisco Packet Tracer can behave in unexpected ways, especially with complex configurations. Some commands don't apply properly unless entered in a very specific order or format.
- Using full configuration blocks instead of entering commands one by one helped avoid bugs. It also made sure settings like `mode tunnel`, and ACL rules were saved correctly.
- Both routers must have clean, matching configs for the VPN to work. If one side is missing even a small setting, the tunnel won't come up.
- Not all problems are caused by the VPN. One issue during testing came from a local PC-to-PC ping that had nothing to do with the tunnel. That added confusion and made troubleshooting harder.

Next time, using full pastes from saved configs earlier in the process could save time. Also, keeping detailed notes after each change helps avoid going in circles.

1.5 Tunnel Traffic Initiation and Final Checks

To test and activate the IPsec VPN tunnel, I sent pings (ICMP echo requests) between two computers on different networks:

- **From** TO-PC (`192.168.10.100`)**:** `ping 192.168.2.100`
- **From** NY-PC (`192.168.2.100`)**:** `ping 192.168.10.100`

At first, the pings may have failed while the VPN tunnel was being built. After that, repeated ping attempts were successful. This confirmed that the two private networks — Toronto and New York — were connected securely through the tunnel.

1.5.1 Tunnel Verification Commands

To check that the VPN tunnel was working, I ran the following command on both the Toronto (TO-FW-EX) Firewall and the New York (NY-FW-EX) Firewall:

```
cli

show crypto ipsec sa
```

This command showed details about encryption, security associations (SAs), and traffic flow between the networks. The output grouped results by source and destination subnet.

Device	Packets Encrypted	Packets Decrypted	SPI (Outbound)	SPI (Inbound)	Status
TO-FW-EX	7	7	0x1BAC53A8	0x567FA704	ACTIVE
NY-FW-EX	7	7	0x567FA704	0x1BAC53A8	ACTIVE

- **Transform Set Used:** `esp-aes 256 esp-sha-hmac`
- **Mode:** Tunnel
- **Replay Detection:** Disabled (simulator limitation)
- **Key Lifetime Remaining:** ~64,000–70,000 seconds
- This confirms **bidirectional encrypted traffic** and tunnel integrity for the primary subnet pair.

Dormant Tunnels – SAs Defined but Inactive (No Matching Traffic Yet)

These Security Associations (SAs) exist but have **not been triggered** due to a lack of interesting traffic:

Local Subnet (NY-FW-EX)	Remote Subnet (TO-FW-EX)	Packets Encrypted/Decrypted	Status
192.168.2.0/24	192.168.20.0/24	0 / 0	Inactive
192.168.2.0/24	192.168.30.0/24	0 / 0	Inactive
192.168.2.0/24	192.168.41.0/24	0 / 0	Inactive
192.168.2.0/24	192.168.50.0/24	0 / 0	Inactive

- These subnet pairs are fully defined in ACLs and crypto maps.
- As per IOS behavior, **IPsec Phase 2 tunnels are not instantiated until traffic flows** between the matched subnets.
- Initiating traffic (e.g., `ping` or `traceroute`) from those subnets would activate the respective SAs.

Summary & Operational Status

- The **Phase 2 IPsec VPN tunnel** is active and working correctly for the tested subnet pair:
  `192.168.10.0/24 ↔ 192.168.2.0/24`.
- This is confirmed by **increasing encrypt and decrypt counters** on both devices.
- Other subnet pairs are also configured properly but are currently inactive. This is expected, since no traffic has been sent from those networks.
- The current setup supports **on-demand SA creation** for all defined subnet pairs — meaning the tunnel will activate automatically when traffic is initiated.

SECTION 2.0 VPN TUNNEL SYMMETRY and DIAGNOSTICS

2.1 Critical Tunnel Matching Guidance

This section outlines the essential parameters that must match for an IPSec VPN tunnel to successfully establish between the Toronto (TO-FW-EX) Firewall and the New York (NY-FW-EX) Firewall. It also highlights the common failure points and the diagnostic checks to confirm proper tunnel alignment.

In IPSec VPN deployment, **symmetry is critical**. Even a minor configuration mismatch between Phase 1 (ISAKMP) or Phase 2 (IPSec) parameters can prevent tunnel establishment, leading to opaque failure modes and time-consuming troubleshooting.

Tunnel Peers:

- **TO-FW-EX** → Public IP: `203.0.113.6`
- **NY-FW-EX** → Public IP: `203.0.113.2`

Checklist for Tunnel Symmetry

1. **Crypto Map Name and Interface Placement**

- **Name Consistency** (e.g., `MCSI-Map2025`) is encouraged but not required.
- **Correct Interface Attachment is mandatory:**
  - `GigabitEthernet0/0/0` on Toronto (TO-FW-EX) Firewall
  - `GigabitEthernet0/0/1` on New York (NY-FW-EX) Firewall

Note: *If the crypto map is not applied to the correct interface, no negotiation will take place.*

---

2. **Matching IPSec Transform-Sets (Phase 2)**

Both devices must agree on:

- **Encryption**: `esp-aes 256`
- **Integrity**: `esp-sha-hmac`
- **Mode**: `tunnel`

Note: *A mismatch here will prevent IPSec Security Associations from forming, causing Phase 2 to fail silently.*

---

3. **Mirrored ACLs for "Interesting" Traffic**

- Each side must define **mirror-opposite ACLs**:
  - NY-FW-EX: `source 192.168.2.0/24 → destination 192.168.10.0/24`
  - TO-FW-EX: `source 192.168.10.0/24 → destination 192.168.2.0/24`

Note: *If ACLs don't match symmetrically, traffic won't be considered "interesting" and the tunnel won't trigger.*

---

4. **Pre-Shared Key (PSK) Match**

- The shared secret must be identical on both ends.

Command format:

```
cli

crypto isakmp key MCSI_SecureVPN_2025! address [peer IP]
```

Note: *A single typo or IP mismatch will cause Phase 1 failure*

---

5. **Configuration Hygiene: Remove → Replace**

Due to Packet Tracer's tendency to retain ghost configurations:

- Remove all prior crypto settings before reapplying:
    - `no crypto isakmp policy [#]`
    - `no crypto ipsec transform-set [name]`
    - `no crypto map [name]`
- Use **full block pastes** when applying a new config to ensure parameter persistence.

Note: *This strategy was essential to resolving simulator-specific issues in this lab.*

By rigorously aligning these parameters and managing config state cleanly, the tunnel can be brought up reliably — even within a constrained lab environment.

SECTION 3.0 CONCLUSION

3.1 Final VPN Status Verification and Wrap-Up

This section confirms that the IPsec site-to-site VPN tunnel between Toronto (TO-FW-EX) and New York (NY-FW-EX) was successfully deployed. While some problems came up during testing, and some of the Packet Tracer tools gave confusing or incorrect results, I was able to confirm the tunnel works using two reliable methods:

- A successful Layer 3 traceroute
- Matched outputs from `show crypto` commands on both devices

3.1.1 Traceroute Verification

The first clear sign of tunnel success came from a traceroute test. I ran this test from TO-PC1 and aimed it at the New York gateway (NY-FW-EX). The traceroute showed that packets were being routed correctly through the encrypted tunnel. It also skipped over the ISP router, which proved that the tunnel was active and doing its job.

> ***Note**: This live routing result helped override any confusion from earlier simulator issues. It confirmed the tunnel was working.*

3.1.2 Validation via `show crypto` Commands

Next, I ran the following CLI commands on both the Toronto and New York firewalls:

- `show crypto isakmp sa`
- `show crypto ipsec sa`

Toronto (TO-FW-EX) Firewall Results

- *Outgoing traffic was being encrypted.*
- *Tunnel status showed as **ACTIVE**.*
- *Matching transform-sets and SPI values were confirmed.*

New York (NY-FW-EX) Firewall Results

- *Inbound traffic was being decrypted properly.*
- *Packet counters showed traffic was being received and processed securely.*
- *Security Associations (SAs) matched the Toronto firewall's output.*

> ***Note**: Both firewalls showed nearly identical counters, matching settings, and active tunnel status—confirming full end-to-end encryption and decryption were working.*

Final Reflection

This project showed how important it is to test a network setup using more than one tool. Even in a lab with limited features, I was able to confirm VPN functionality using simple commands and step-by-step checks. Using traceroute to test packet flow and then checking encryption status with `show crypto`, gave me strong proof that the tunnel worked.

> **Takeaway**: Don't depend on just one method. Use both command-line checks and live network testing together to make sure your IPSec tunnel is up and working.
