Troubleshoot Proxy on Cisco Secure Firewall Management Center (FMC)

Contents

Introduction

This document describes configuring a proxy on FMC to allow users to connect to the Internet through an intermediary server, enhancing security and sometimes improving performance. This article guides you through the steps to configure a proxy on FMC and provide troubleshooting tips for common issues.

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco Secure Firewall Management Center (FMC)
- Proxy

Components Used

The information in this document is based on these software and hardware versions:

- FMC 7.4.x

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Configuration

Configure Network http-proxy on FMC GUI:

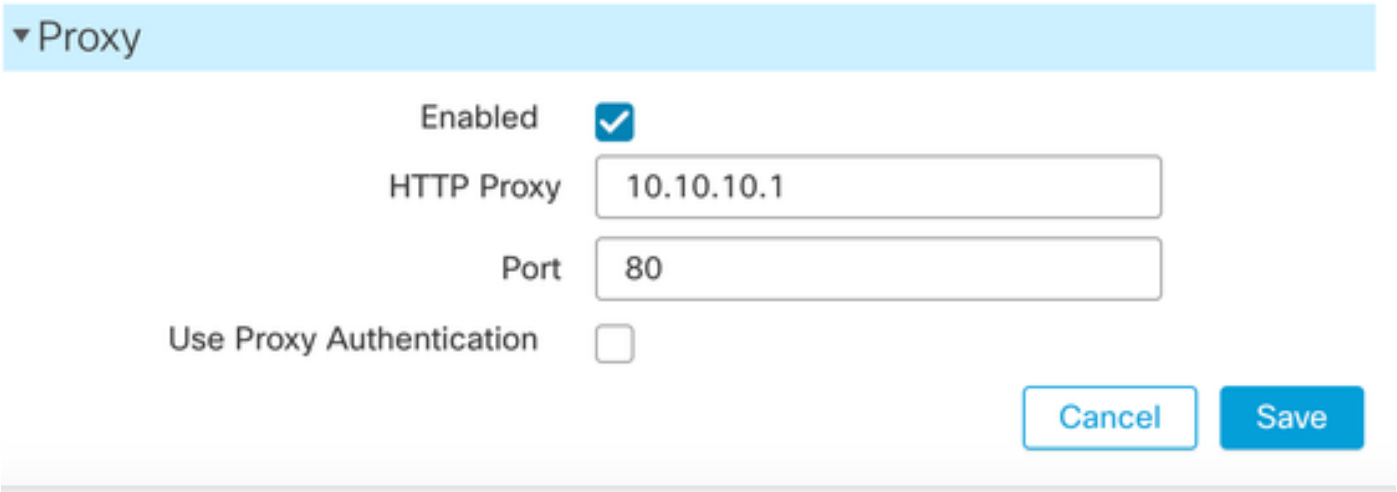
Login **FMC GUI** > Choose **System** > **Configuration**, and then choose **Management Interfaces**.

Note: Proxies that use NT LAN Manager (NTLM) authentication are not supported. If you use Smart Licensing, the proxy FQDN cannot have more than 64 characters.

In the **Proxy** area, configure HTTP proxy settings.

The management center is configured to directly connect to the Internet on ports TCP/443 (HTTPS) and TCP/80 (HTTP). You might use a proxy server, to which you might authenticate via HTTP Digest.

- Check the**Enabled**check box.
- In the**HTTP Proxy**field, enter the IP address or fully-qualified domain name of your proxy server.
- In the**Port**field, enter a port number.
- Supply authentication credentials by choosing**Use Proxy Authentication**, and then provide a**User Name**and**Password**.
- Click **Save**.



Note : For the proxy password you can use A-Z, a-z, and 0-9 and special characters.

Troubleshoot

Get access to **FMC CLI** and expert mode, then verify **iprep_proxy.conf** to ensure proxy settings are correct:

<#root>

admin@fmc:~$

cat /etc/sf/iprep_proxy.conf

```
iprep_proxy {
PROXY_HOST 10.10.10.1;
PROXY_PORT 80;
}
```

Check the active connections to verify the active proxy connection:

<#root>

admin@fmc:~$

netstat -na | grep 10.10.10.1

```
tcp 0 0 10.40.40.1:40220 10.10.10.1:80
```

ESTABLISHED

Using the **curl** command, verify both the request details and the response from the proxy. If you receive the response: **HTTP/1.1 200 Connection established**, this indicates that the FMC is successfully sending and receiving traffic through the proxy.

<#root>

admin@fmc:~$

curl -x http://10.10.10.1:80 -I https://tools.cisco.com

HTTP/1.1 200 Connection established

If you have configured the username and password for the proxy, verify the authentication and proxy response:

```
curl -u proxyuser:proxypass --proxy http://proxy.example.com:80 https://example.com
```

Verification

Successful Connection Establishment via Proxy

When running a **curl** command with a proxy, such as **curl -x http://proxy:80 -I https://tools.cisco.com**, a

series of expected network interactions occur, which can be observed through packet capture (tcpdump). This is a high-level overview of the process, enriched with real **tcpdump** outputs:

TCP Handshake Initiation:
The client (FMC) initiates a TCP connection to the proxy server on port 80 by sending a SYN packet. The proxy responds with a SYN-ACK, and the client completes the handshake with an ACK. This establishes the TCP session over which HTTP communication proceeds.

Example tcpdump output:

```
10:20:58.987654 IP client.54321 > proxy.80: Flags [S], seq 0, win 64240, options [mss 1460], length 0
10:20:58.987700 IP proxy.80 > client.54321: Flags [S.], seq 0, ack 1, win 65160, options [mss 1460], le
10:20:58.987734 IP client.54321 > proxy.80: Flags [.], ack 1, win 64240, length 0
```

HTTP CONNECT Request:
Once the TCP connection is established, the client sends an HTTP CONNECT request to the proxy, instructing it to create a tunnel to the target HTTPS server (tools.cisco.com:443). This request allows the client to negotiate an end-to-end TLS session through the proxy.

Example tcpdump (decoded HTTP):

```
CONNECT tools.cisco.com:443 HTTP/1.1
Host: tools.cisco.com:443
User-Agent: curl/8.5.0
Proxy-Connection: Keep-Alive
```

Connection Establishment Acknowledgment:
The proxy replies with an HTTP/1.1 200 Connection established response, indicating that the tunnel to the target server has been successfully created. This means the proxy is now acting as a relay, forwarding encrypted traffic between the client and tools.cisco.com.

Example tcpdump:

<#root>

```
HTTP/1.1
```

200

```
 Connection established
```

HTTPS Communication via Tunnel:
Following the successful CONNECT response, the client initiates the SSL/TLS handshake directly with tools.cisco.com over the established tunnel. Since this traffic is encrypted, the contents are not visible in the tcpdump, but packet lengths and timings are observable, including TLS Client Hello and Server Hello

packets.

Example tcpdump:

```
10:20:59.123456 IP client.54321 > proxy.80: Flags [P.], length 517 (Client Hello)
10:20:59.123789 IP proxy.80 > client.54321: Flags [P.], length 1514 (Server Hello)
```

Handling of HTTP Redirect (302 Found):
As part of the HTTPS communication, the client requests the resource from tools.cisco.com. The server responds with an HTTP/1.1 302 Found redirect to another URL (https://tools.cisco.com/healthcheck), which the client can follow depending on the curl parameters and purpose of the request. Although this redirect occurs within the encrypted TLS session and is not directly visible, it is expected behavior and can be observed if TLS traffic is decrypted.

The encrypted redirect traffic would appear like this:

```
10:21:00.123000 IP client.54321 > proxy.80: Flags [P.], length 517 (Encrypted Application Data)
10:21:00.123045 IP proxy.80 > client.54321: Flags [P.], length 317 (Encrypted Application Data)
```

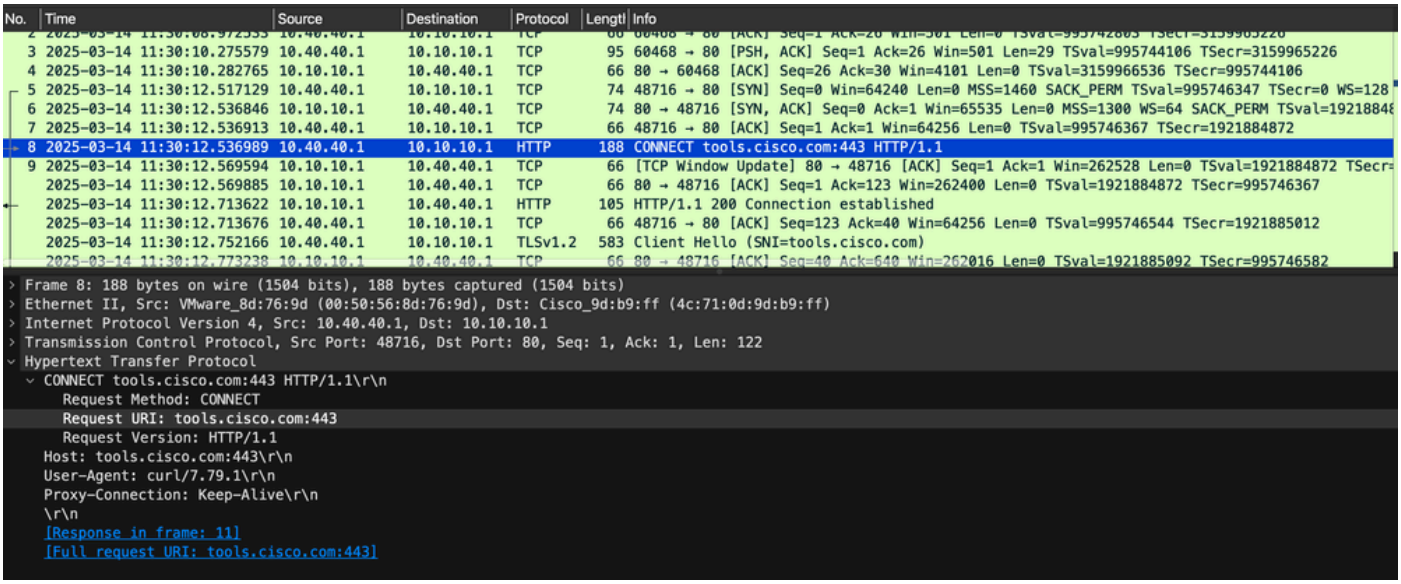
Connection Teardown:
Once the exchange is complete, both client and proxy gracefully close the TCP connection by exchanging FIN and ACK packets, ensuring proper session termination.

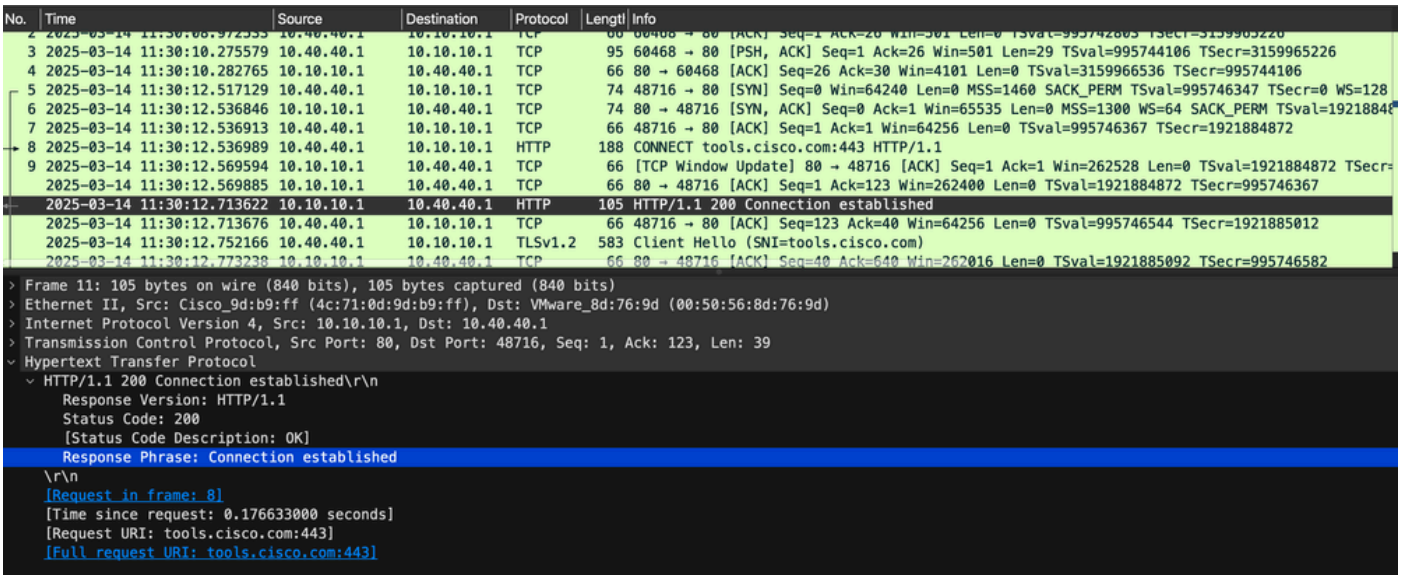
Example tcpdump output:

```
10:21:05.000111 IP client.54321 > proxy.80: Flags [F.], seq 1234, ack 5678, length 0
10:21:05.000120 IP proxy.80 > client.54321: Flags [F.], seq 5678, ack 1235, length 0
10:21:05.000125 IP client.54321 > proxy.80: Flags [.], ack 5679, length 0
```

Tip: By analyzing the tcpdump output, you can verify that the HTTPS request through the explicit proxy follows the expected flow: TCP handshake, CONNECT request, tunnel establishment, TLS handshake, encrypted communication (including possible redirects), and graceful connection closure. This confirms the proxy and client interaction is working as designed and helps identify any issues in the flow, such as failures in tunneling or SSL negotiation.

The FMC (10.40.40.1) establishes a successful TCP handshake with the Proxy (10.10.10.1) on port 80, followed by an HTTP CONNECT to the server (72.163.4.161) on port 443. The server replies with an **HTTP 200 Connection established** message. The TLS handshake completes, and data flows properly. Finally, the TCP connection terminates gracefully (FIN).





Known Issues

Proxy ACL Restrictions

If there is a permission issue (like an access list on the proxy), you can observe that through packet capture (tcpdump). This is a high-level explanation of the failure scenario, along with example tcpdump outputs:

TCP Handshake Initiation:
The client (Firepower) starts by establishing a TCP connection to the proxy on port 80. The TCP handshake (SYN, SYN-ACK, ACK) completes successfully, which means the proxy is reachable.

Example tcpdump output:

```
10:20:58.987654 IP client.54321 > proxy.80: Flags [S], seq 0, win 64240, options [mss 1460], length 0
10:20:58.987700 IP proxy.80 > client.54321: Flags [S.], seq 0, ack 1, win 65160, options [mss 1460], le
10:20:58.987734 IP client.54321 > proxy.80: Flags [.], ack 1, win 64240, length 0
```

HTTP CONNECT Request:
Once connected, the client sends an HTTP CONNECT request to the proxy, asking it to create a tunnel to tools.cisco.com:443.

Example tcpdump (decoded HTTP):

```
CONNECT tools.cisco.com:443 HTTP/1.1
Host: tools.cisco.com:443
User-Agent: curl/8.5.0
Proxy-Connection: Keep-Alive
```

Error Response from Proxy:

Instead of allowing the tunnel, the proxy denies the request, likely because of an access list (ACL) that doesn't permit this traffic. The proxy responds with an error like 403 Forbidden or 502 Bad Gateway.

Example tcpdump output showing error:

<#root>

```
HTTP/1.1
```

403

```
 Forbidden
Content-Type: text/html
Content-Length: 123
Connection: close
```

Connection Teardown:
After sending the error message, the proxy closes the connection, and both sides exchange FIN/ACK packets.

Example tcpdump output:

```
10:21:05.000111 IP client.54321 > proxy.80: Flags [F.], seq 1234, ack 5678, length 0
10:21:05.000120 IP proxy.80 > client.54321: Flags [F.], seq 5678, ack 1235, length 0
10:21:05.000125 IP client.54321 > proxy.80: Flags [.], ack 5679, length 0
```

Tip: From the tcpdump, you can see that although the TCP connection and HTTP CONNECT request were successful, the proxy denied the tunnel setup. This usually indicates that the proxy has an ACL or permission restriction preventing the traffic from passing.

Proxy Fails Download (Timeout/Incomplete Transfer)

In this scenario, FMC successfully connects to the proxy and starts the file download, but the transfer times out or fails to complete. This is typically due to proxy inspection, timeouts, or file size limits on the proxy.

TCP Handshake Initiation
FMC initiates a TCP connection to the proxy on port 80, and the handshake completes successfully.
Example tcpdump output:

```
10:20:58.987654 IP fmc.54321 > proxy.80: Flags [S], seq 0, win 64240, options [mss 1460], length 0
10:20:58.987700 IP proxy.80 > fmc.54321: Flags [S.], seq 0, ack 1, win 65160, options [mss 1460], lengt
10:20:58.987734 IP fmc.54321 > proxy.80: Flags [.], ack 1, win 64240, length 0
```

HTTP CONNECT Request
FMC sends an HTTP CONNECT request to the proxy to reach the external target.
Example tcpdump (decoded HTTP):

```
CONNECT tools.cisco.com:443 HTTP/1.1
Host: tools.cisco.com:443
User-Agent: FMC-Agent
Proxy-Connection: Keep-Alive
```

Tunnel Establishment and TLS Handshake
Proxy responds with HTTP/1.1 200 Connection established, allowing the TLS handshake to begin.

Example tcpdump output:

<#root>

```
HTTP/1.1
```

200

```
 Connection established
10:20:59.123456 IP fmc.54321 > proxy.80: Flags [P.], length 517 (Client Hello)
10:20:59.123789 IP proxy.80 > fmc.54321: Flags [P.], length 1514 (Server Hello)
```

Timeout or Incomplete Download
After initiating the file transfer, the download stalls or does not complete, leading to a timeout. The connection remains idle.
Possible reasons include:

- Proxy inspection delays or filtering.
- Proxy timeouts for long transfers.
- File size limits imposed by the proxy.

Example tcpdump showing inactivity:

<#root>

```
10:21:00.456000 IP fmc.54321 > proxy.80: Flags [P.], length 1440
```

 # FMC sending data

 # No response from proxy, connection goes idle...

 # After a while, FMC may close the connection or retry.

Tip: FMC initiates the download but fails to complete due to timeouts or incomplete transfers, often caused by proxy filtering or file size restrictions.

Proxy Fails File Download (MTU Issue)

In this case, FMC connects to the proxy and starts downloading files, but the session fails due to MTU issues. These issues cause packet fragmentation or dropped packets, especially with large files or SSL/TLS handshakes.

TCP Handshake Initiation
FMC initiates TCP handshake with the proxy, which succeeds.
Example tcpdump output:

```
10:20:58.987654 IP fmc.54321 > proxy.80: Flags [S], seq 0, win 64240, options [mss 1460], length 0
10:20:58.987700 IP proxy.80 > fmc.54321: Flags [S.], seq 0, ack 1, win 65160, options [mss 1460], lengt
10:20:58.987734 IP fmc.54321 > proxy.80: Flags [.], ack 1, win 64240, length 0
```

HTTP CONNECT Request and Tunnel Establishment
FMC sends an HTTP CONNECT request, and the proxy responds, allowing the tunnel to be established.
Example tcpdump (decoded HTTP):

```
CONNECT tools.cisco.com:443 HTTP/1.1
Host: tools.cisco.com:443
User-Agent: FMC-Agent
Proxy-Connection: Keep-Alive
```

TLS Handshake Begins
FMC and tools.cisco.com start negotiating SSL/TLS, and the initial packets are exchanged.
Example tcpdump output:

<#root>

```
HTTP/1.1

200

 Connection established
10:20:59.123456 IP fmc.54321 > proxy.80: Flags [P.], length 517 (Client Hello)
10:20:59.123789 IP proxy.80 > fmc.54321: Flags [P.], length 1514 (Server Hello)
```

Packet Fragmentation or Drop Due to MTU
When FMC or the server attempts to send large packets, MTU issues cause packet fragmentation or packet drops, resulting in file transfer or TLS negotiation failures.
This typically occurs when the MTU between FMC and the proxy (or between the proxy and the Internet) is incorrectly set or too small.
Example tcpdump showing fragmentation attempt:

<#root>

```
10:21:00.456000 IP fmc.54321 > proxy.80: Flags [P.], length 1440

# Large packet


10:21:00.456123 IP proxy.80 > fmc.54321: Flags [R], seq X, win 0, length 0

# Proxy resets connection due to MTU issue
```

Tip: MTU issue results in dropped or fragmented packets, which disrupt the TLS handshake or cause file downloads to fail. This is commonly seen when SSL inspection or packet fragmentation occurs due to incorrect MTU settings.

In a failure scenario, FMC gets CONNECT without HTTP 200, with retransmissions and FINs confirming no TLS/data exchange, possibly due to MTU issues or a proxy/upstream problem.

When using curl, you can encounter various **HTTP response codes** indicating server-side issues or authentication errors. This is a list of the most common error codes and their meanings:

HTTP Code	Meaning	Cause
400	Bad Request	Incorrect request syntax
401	Unauthorized	Missing or incorrect credentials
403	Forbidden	Access denied
404	Not Found	Resource not found
500	Internal Error	Server error
502	Bad Gateway	Server miscommunication
503	Service Unavailable	Server overload or maintenance
504	Gateway Timeout	Timeout between servers

HTTP Code	Meaning	Cause

References

Cisco Secure Firewall Threat Defense Release Notes, Version 7.4.x