

# Secure DX UIM Communications to Meet Emerging Requirements for IT Operators

## TABLE OF CONTENTS

---

### DX UIM Architecture Overview

### Hub Communications in a Non-tunneled Environment

### Hub Communications in a Tunneled Environment

### Create a Tunneled Environment from a Non-tunneled Environment

### Verify Hub Tunnels Are Properly Configured

### Correct an Improper Configuration

### Conclusion

Security has become one of the most daunting challenges facing IT operations, and it has become increasingly more important as the volume of data flowing across networks worldwide increases. Enterprises, government agencies, and managed service providers handle data of all sorts, from personally identifiable information to quality of service metrics, and IT is often required to secure communications between the infrastructures within their control.

Quality of service metrics are at the heart of infrastructure monitoring tools. DX Unified Infrastructure Management (DX UIM) by Broadcom® Software is a monitoring tool that ensures communications between IT components are secure. Within the DX UIM architecture, one of the most important components is the hub probe. The hub probe is at the core of the DX UIM infrastructure, and is responsible for routing traffic within an environment. Large environments can have multiple hubs. Securing the channel between the DX UIM hubs is now a requirement for many IT operators. This can be done efficiently and securely with hub tunnels.

This paper explains the intricacies of hub tunnels, their uses, and how to configure them correctly. DX UIM users will understand how to secure DX UIM component communications, how to ensure tunnels are properly configured, and how to both validate configurations and correct invalid configurations.

## DX UIM Architecture Overview

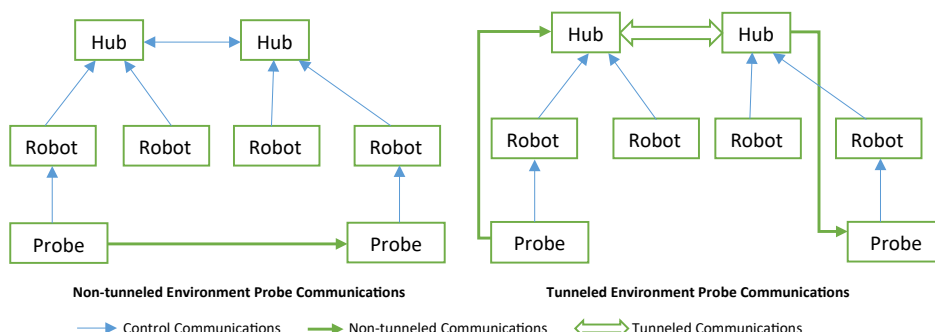
A DX UIM hub probe is one of the most powerful and important components in the DX UIM environment. The hub probe provides security, transports QOS metrics to the database, resolves DX UIM addresses to the correct IP and port, and manages robots.

To create a secure VPN-like connection for communication through firewalls and between hubs, the functionality of hub tunnels is used. Hub tunnels are end-to-end secure channels between hubs, and they communicate via TLS. The tunnel server is the hub that acts as a certifying authority for the tunnel clients; certificates are created here for tunnel clients. The tunnel client is the hub that connects to the tunnel server. It uses the certificate provided by the tunnel server to make the secure connection.

## TO CREATE A SECURE VPN-LIKE CONNECTION FOR COMMUNICATION THROUGH FIREWALLS AND BETWEEN HUBS, THE FUNCTIONALITY OF HUB TUNNELS IS USED

Once tunnels are enabled, all inter-hub communications occur through hub tunnels. All requests are directed to the local hub, and tunnels in the local hub securely transport requests between the hubs. Similarly, incoming requests to probes will be redirected to the local hub, and the local hub will pass the request on to the correct destination probe.

Figure 1: Probe Communications in Tunneled and Non-tunneled Environments



Tunneled communications do have pros and cons:

### Pros

- Tunnels provide secure transport between hubs.
- There are hubs which have to be exposed to outside networks, but not all hubs and robots need to be exposed to outside networks. A tunnel-configured hub can act as an interface to the set of the robots under it; these robots can be behind a firewall where access to them is provided only through the tunnel hub.
- Access within the DX UIM infrastructure can be restricted and managed through tunnels. Configure the access control list in the tunnel configuration; traffic from a particular source or toward a particular destination can be allowed or denied based on ACL rules.
- UDP hubup broadcasts are no longer needed, which reduces the network usage for DX UIM traffic.

### Cons

- Security comes at a potential cost.
  - Using TLS/SSL at the transport level and transporting DX UIM data over multiple hops can impact performance in comparison to a non-tunneled environment, where a direct connection on a socket is created.
  - Depending on traffic volume, the overhead of encrypting and decrypting SSL traffic can have an impact on the CPU usage of the hub process; higher levels of encryption and stronger ciphers can increase this impact.

## TUNNELS PROVIDE SECURE TRANSPORT BETWEEN HUBS

## A DX UIM HUB PROBE IS ONE OF THE MOST POWERFUL AND IMPORTANT COMPONENTS IN THE DX UIM ENVIRONMENT

### Hub Communications in a Non-tunneled Environment

To determine whether a tunneled or non-tunneled environment is preferable, it's important to understand how each environment functions. In a non-tunneled environment, after the hub comes up it reads the hubs.sds file in the hub installation directory and stores it in the process cache called hubhash. The hub holds associated robots and probe names, as well as the port numbers running on all the robots under the hub in the hub's local cache.

The hubhash contains the details of all the hubs that are connected to that hub. The hubhash is populated with the UDP broadcasts (received on hub UDP port 48002) using the hub updates received from its peer hubs; these updates are called hubups. The hubups received from the peer hubs contain information about the sending hub, including hubname, hubdomain, hubip, and hubport.

Along with the sending hub details, the sender hub also propagates its own hubhash. The hubhash in every hub holds the details of all the hubs present in that domain.

For every hub entry in hubhash, a parameter called proximity is updated. Proximity defines how far a particular hub is from the current hub. The proximity of the hub determines the route for a hub; the hub with the least proximity (the closest hub) will be chosen as the next hop.

The hubhash that is created by the hubups is used to resolve the DX UIM addresses to a valid IP and port (name-to-IP resolution) for the communication between the probes. This resolved IP and port are used to create the direct connection between the two probes.

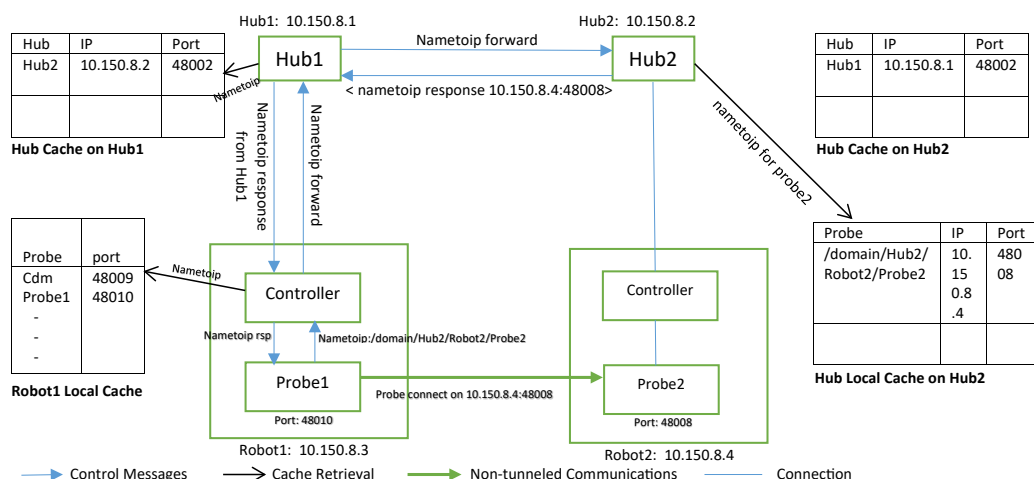
The following example demonstrates how a probe communicates with a probe that is present on a robot under a destination hub, using non-tunneled communications:

1. The probe asks the local controller for the IP address of the destination probe. Since the destination probe is not present on the local machine, the request is redirected to the source hub.
2. The source hub checks the hub name in the request. Since the source hub is not the same as the destination hub, the request is forwarded to the destination hub. This happens in single or multiple hops, based on proximity.
3. The destination hub checks the probes in that hub's local cache, which holds both the probe names and the ports of all the robots under that hub, and gives the IP and port of the probe to the source hub.
4. The source hub passes on the IP and port information to the source probe, which uses that information to create the connection to the destination probe.
5. The connection between the source probe and the destination probe is a direct connection on the destination robot IP address and destination probe port.

In the process explained above, the actual name-to-IP resolution happens on

the destination hub where the destination hub's local cache is used to resolve the IP address of the destination probe. Once the resolution happens, the information is sent to the source probe's hub, which in turn passes it on to the source probe. Then source probe can directly connect to the destination probe.

**Figure 2: Communication between Two Probes on Different Hubs in a Non-tunneled Environment**



## Hub Communications in a Tunneled Environment

Communication between probes in a tunneled environment is not direct; it happens in three steps:

1. The source probe communicates with the source hub.
2. The source hub communicates with the destination probe's hub.
3. The destination probe's hub communicates directly with the destination probe.

When tunnels are configured in the hub, for every tunnel connection toward another tunnel hub, a dedicated server thread is created to listen for the local probes and for the probes running on other robots under that hub. This dedicated thread creates, on-demand, sender and receiver threads on which the data is passed. These sender and receiver threads do the actual communication between hubs. These threads communicate over TLS/SSL, hence the connection is secured at the transport level.

In tunneled environments, a hub is configured either as a tunnel server or a tunnel client. A tunnel client communicates only with its tunnel server. The tunnel server acts as a central point for communication; when traffic from the source tunnel client reaches its tunnel server, it is passed to the destination tunnel client. Since communication happens only between the tunnel client and tunnel server, the dedicated threads on tunnel clients are configured only toward their respective tunnel server. The tunnel server's dedicated threads are created for every tunnel client connected to that tunnel server.

**THREADS COMMUNICATE  
OVER TLS/SSL,  
ENSURING SECURE  
COMMUNICATIONS AT THE  
TRANSPORT LEVEL**

## HIGHER-PERFORMING HOSTS SHOULD BE USED FOR TUNNEL SERVERS

Any communication toward a particular tunnel hub uses these dedicated threads to communicate with the hub. Tunnel clients are only connected to tunnel servers, and all outgoing traffic routes through its tunnel server. This means a tunnel server with many clients is much busier than a normal tunnel client hub. Therefore, higher-performing hosts should be used for tunnel servers. This ensures that the tunnel server is not configured on high-volume hubs or hubs that can become busy. Configuration should not be assigned to the primary hub.

The following example details communications between tunnel clients Hub1 and Hub2, which are connected to the tunnel server. The traffic flows from a source probe Robot1/Probe1 running on tunnel client Hub1 toward destination probe Robot2/Probe2 running on tunnel client Hub2. Traffic between the tunnel server and tunnel clients is over TLS/SSL.

1. Traffic from source probe Probe1, running on Robot1, is sent to source tunnel client hub, Hub1, for communication with the tunnel server. The source tunnel client hub will handle transport toward the tunnel server.
2. In the tunnel server, traffic is redirected to the thread created for communication between the tunnel server and the destination tunnel client hub, Hub2.
3. The destination tunnel client hub, Hub2, redirects the traffic to the destination probe, Probe2, running on Robot2.

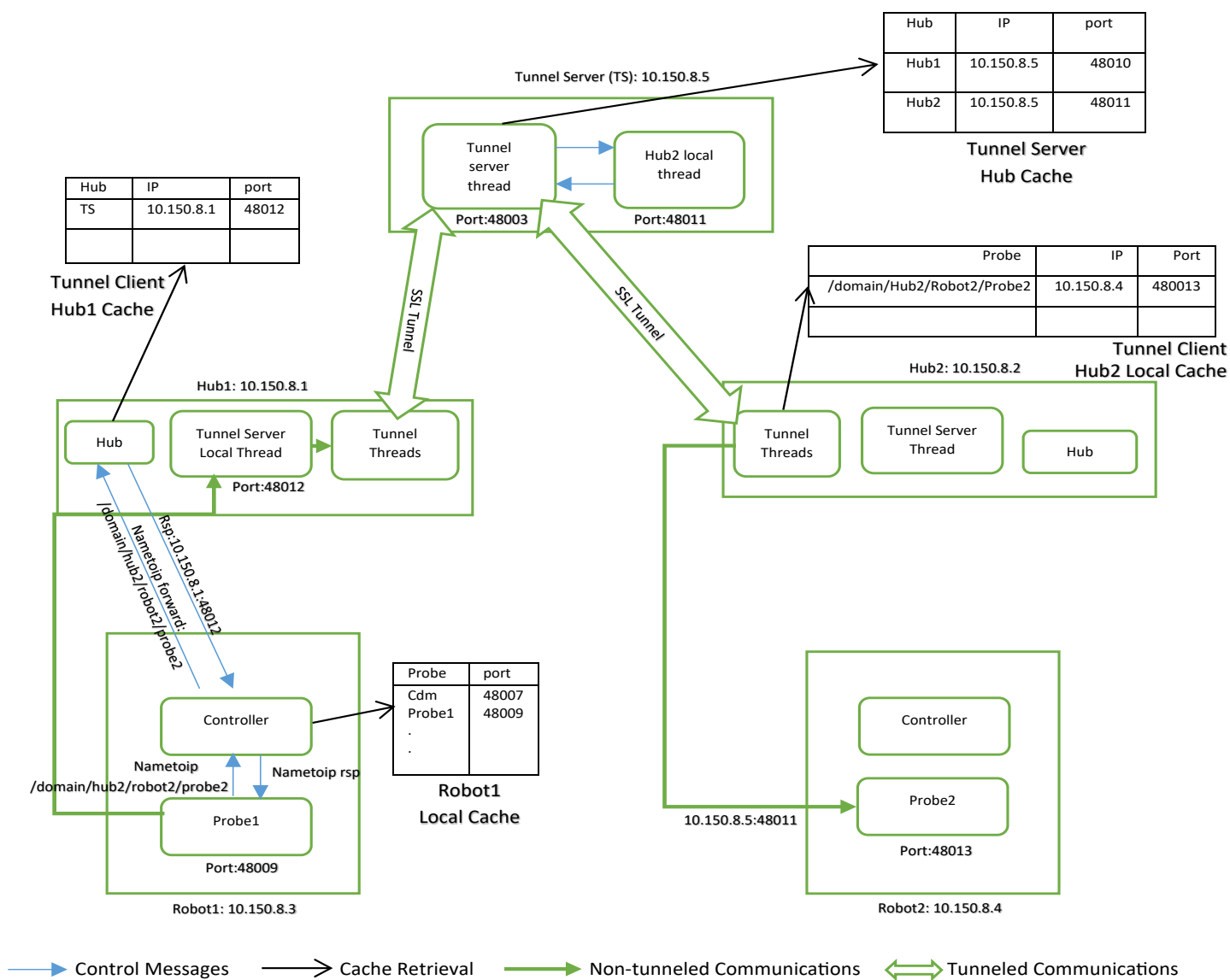
In both tunneled and non-tunneled environments, hubhash is created so that hubups can be transmitted between the hubs. In tunneled environments, however, the broadcasts are not UDP; they are TCP unicast hubups. These hubups form the hubhash in the tunneled environment. This hubhash is used during the resolution for the given DX UIM IP address. Unlike in the non-tunneled hubs, hubups in the tunneled hubs holds the port numbers of the dedicated tunnel threads of the local hub designated for a particular destination hub.

Consider the following: a probe with the DX UIM address /Domain/Hub1/Robot1/Probe1 running on IP 10.150.8.3 wants to communicate with a probe /Domain/Hub2/Robot2/Probe2 running on IP 10.150.8.4, when a name-to-IP resolution is requested for Probe2 from Probe1. Below is the name-to-IP resolution for tunneled and non-tunneled environments:

- Non-tunneled: 10.150.8.4:48013 (Probe2 actual IP and port)
- Tunneled: 10.150.8.1:48012 (local hub IP and port of tunnel thread created for tunnel server on Hub1)

In a tunneled environment, the resolution happens to the local hub IP and the tunnel port instead of the actual Probe2 IP and port, because name-to-IP resolutions in a tunneled environment is not forwarded to the destination Hub2.

Figure 3: Communication in a Tunneled Environment between Probes on Different Tunnel Clients





## REVERTING A FAILED TUNNEL ENVIRONMENT MAY INVOLVE DOWNTIME

## Create a Tunneled Environment from a Non-tunneled Environment

You can convert a non-tunneled environment to a tunneled environment. If this conversion is not completed correctly, the tunnels will not work properly and may not be usable. Reverting from a failed tunneled environment back to a non-tunneled environment can be a very tedious task, and may involve downtime for the entire environment. Follow these steps to correctly configure a tunneled environment:

1. Configure the tunnels as detailed in the Broadcom Knowledge Base: <https://knowledge.broadcom.com/external/article/12105/how-to-create-tunnels-between-two-hubs-a.html>
2. Ensure that the **Tunnel Status** tab in the hub Infrastructure Management GUI reflects the configured tunnel and that there is a flow of traffic toward the destination of the tunnel. This shows data is being transferred over the tunnels, which means a connection is established over hub tunnels.

Figure 4: Tunnel Status Tab in the Infrastructure Management GUI

General

Hubs

Robots

Name Services

Queues

Tunnels


Status




Subscribers/Queues

Subjects

Tunnel Status

Tunnel Statistics

Peer Hub	Started	Last	Connection Stats(ms)	Connections	Traffic In/Out
 bltest001315 [10.89.245.1...	09/15/22 11:53:04	11:53:05	10 / 38 / 93	6	946 KB/33 KB

State	Started	Last	In	Out	Address	Command
 active	09/15/22 11:53:04	09/15/22 11:58:59	938 KB	22 KB	/bltest002564_dom...	heartbeat
 idle	09/15/22 11:53:04	09/15/22 11:57:22	2.0 KB	6.4 KB	/bltest002564_dom...	hubup
 idle	09/15/22 11:53:04	09/15/22 11:57:23	5.5 KB	3.4 KB	/bltest002564_dom...	hubup

OK

Cancel

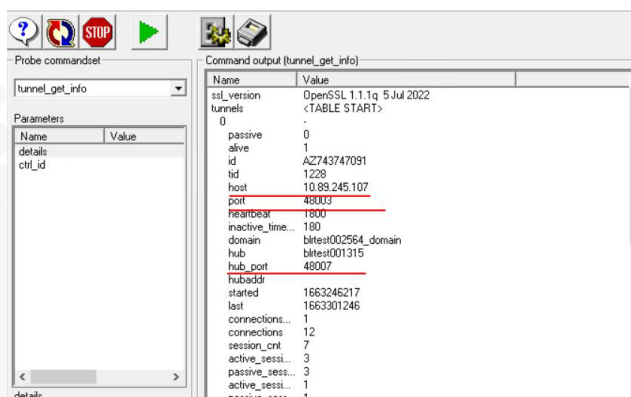
Apply

Refresh

Help

3. To further ensure connectivity, verify that the dedicated tunnel port is created locally for a destination tunnel hub. Check this using the hub callback `tunnel_get_info`. Figure 5 shows the output of the `tunnel_get_info` callback on a tunnel client hub, which illustrates that to reach the tunnel server hub running on IP 10.89.245.107 and port 48003, local hub\_port 48007 is used.

Figure 5: Verify the Dedicated Tunnel Port Is Created Locally

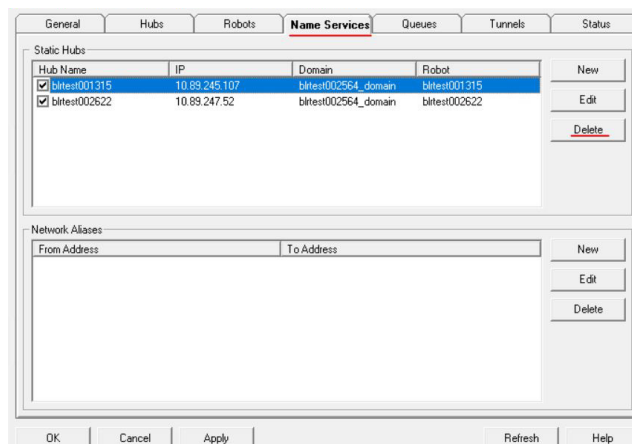


Name	Value
ssl_version	OpenSSL 1.1.1q 5 Jul 2022
tunnels	<TABLE START>
0	
passive	0
alive	1
id	AZ743747091
tid	1228
host	10.89.245.107
port	48003
heartbeat	18000
inactive_time...	180
domain	bltest002564_domain
hub	bltest001315
hub_port	48007
hubaddr	
started	1663246217
last	1663301246
connections...	1
connections	12
session_cnt	7
active_sessi...	3
passive_sess...	3
active_sessi...	1
passive_sess...	1

## AFTER ENSURING THE TUNNELS ARE CONFIGURED AND WORKING, STOP THE UDP BROADCASTS THAT PROPAGATE THE NON-TUNNEL HUB PORTS.

4. After ensuring that the tunnels are configured and working, it is important to stop the UDP broadcasts that propagate the non-tunnel hub ports. Set parameter **broadcast\_on** to **no** in the **hub.cfg**, under the **<hub>** section. By default this parameter is set to **yes**, so if it not set to **no**, this will pollute the hub cache to give non-tunnel IPs and ports when the name-to-IP is inquired on the hubs
5. Remove the hubs in the **Name Services** tab of the hub configuration. These static routes bypass the tunnels and create connectivity in a non-tunneled manner, even if the tunnels are configured.

Figure 6: Remove a Hub from the Name Services Tab



6. Remove the **hubs.sds** file from the hub's installation directory. This file holds the old hub cache with non-tunneled IPs and ports, so removing it helps the hub create a new cache with the tunnel ports.
7. Restart the hub probe (DX UIM restart) to implement the above changes on the hub.
8. Perform the above steps for every hub running tunnels.



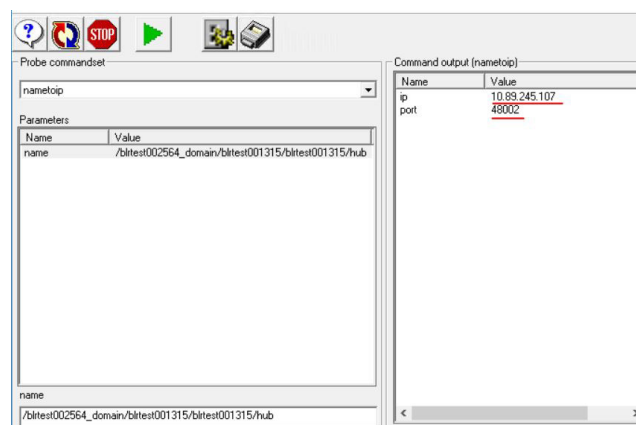
## Verify Hub Tunnels Are Properly Configured

After the tunnel configuration is complete, it is very important to ensure the hub tunnels are enabled for communications. Unlike in non-tunneled environments, where the communication toward the destination probe happens on its IP and port, in tunneled environments, if the DX UIM communication is toward the destination probes residing under the destination hub, it should first reach the source probe's local hub where tunnels are configured. Tunnels between the source tunnel client hub and the destination tunnel client hub pass the traffic to the destination probes.

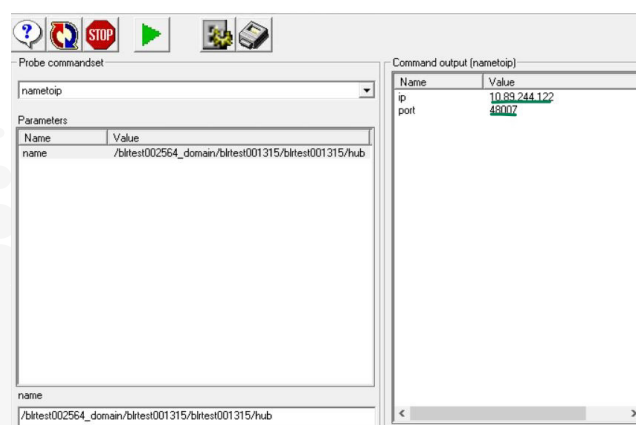
Every DX UIM request obtains the destination IP and port using the hub callback name-to-IP, which receives the DX UIM address as an input and returns its IP and port as the result. To ensure that communication is occurring on the hub tunnels, trigger a name-to-IP callback on the source hub with input for the destination hub. If the result is the source tunnel hub IP and port xref, then that means hub tunnels are in use. If the result is the destination probe's IP and port, then that means the hub tunnels are not in use, and communications are not secure.

**TO ENSURE  
COMMUNICATION IS  
OCCURRING, TRIGGER A  
NAME-TO-IP CALLBACK**

**Figure 7: Tunnels are Not in Use: The Name-to-IP of the Destination Hub is the Destination Hub IP (10.89.245.107) and Destination Hub Port 48002**



**Figure 8: Tunnels are in Use: The Name-to-IP of the Destination Hub is the Source Hub IP and the Hub Tunnel Port**



## Correct an Improper Configuration

An improper configuration happens when the hub cache doesn't hold the tunnel's IP and ports for a particular destination hub in spite of enabling tunnels, and instead holds the probe's actual IP and port. The hub cache is used by the hub name-to-IP callback to resolve the IP address to a given DX UIM address during communication. So, when a name-to-IP is requested for a hub, it will give the actual IP and probe's port instead of local IP and tunnel's port. Because of this improper configuration, instead of tunnel communications, direct communication toward the destination probe will be in use. This means tunnels are bypassed.

Perform the following steps to remediate the hub cache issue:

1. Stop the tunnels on all the hubs; this will stop the hubhash exchange between the hubs.
2. Repeat the steps in the [Create a Tunneled Environment from a Non-tunneled Environment](#) section.
3. After performing the above steps, check if the issue was resolved and tunnels are in use toward the destination hubs; use the hub callback name-to-IP to verify that tunnels are in place.

## Conclusion

Configuring DX UIM hub tunnels for secure communications is advantageous for the security requirements now imposed on IT operators in enterprise, government agencies, and managed service providers. It is important to configure hub tunnels properly for secure transport. If hub tunnels are not configured correctly, non-tunnel communications can override what were intended to be tunnel communications. Best practices include ensuring tunnels are actually in use after the configuration to avoid rework and downtime.

## LEVERAGING DX UIM HUB TUNNELS FOR SECURE COMMUNICATIONS IS ADVANTAGEOUS

For more information, refer to the [Broadcom Knowledge Base](#):

<https://knowledge.broadcom.com/external/article/12105/how-to-create-tunnels-between-two-hubs-a.html>



### About Broadcom Software

Broadcom Software is a world leader in business-critical software that modernizes, optimizes, and protects the world's most complex hybrid environments. With its engineering-centered culture, Broadcom Software has an extensive portfolio of industry-leading infrastructure and security software, including AI/ops, Cybersecurity, Value Stream Management, DevOps, Mainframe, and Payment Security. Our software portfolio enables scalability, agility, and security for the largest global companies in the world.

For more information, visit our website at: [software.broadcom.com](https://software.broadcom.com)

Copyright © 2023 Broadcom. All Rights Reserved. The term "Broadcom" refers to Broadcom Inc. and/or its subsidiaries. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

DXIUM-WP100 February 9, 2023