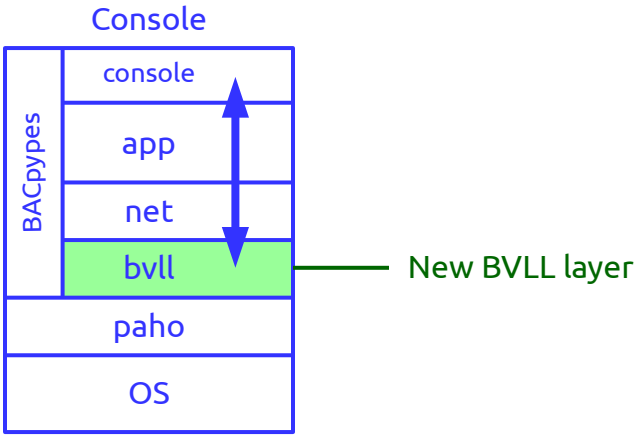
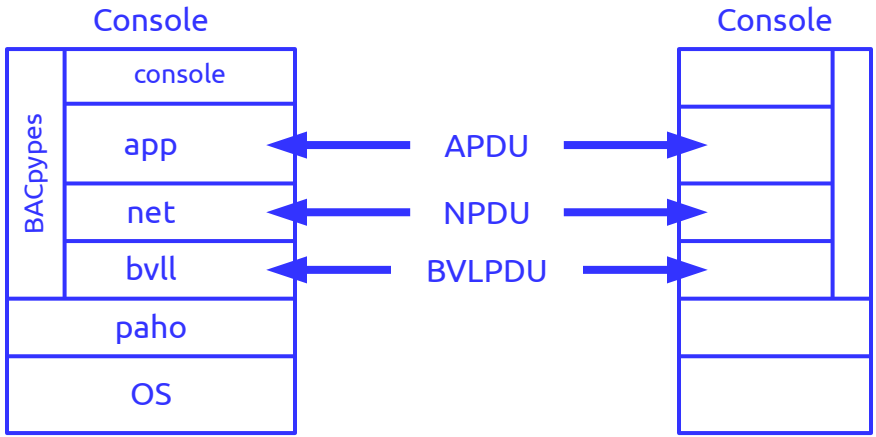
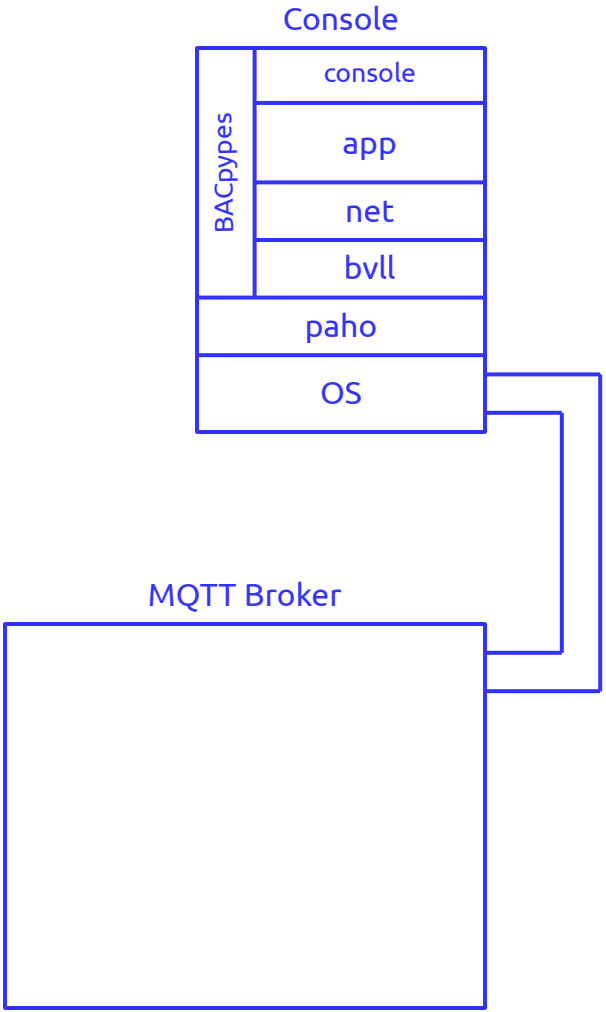


The sample console application is a Python application that uses the paho library from the Eclipse project. It has the same structure as BACnet/IP applications but rather than using UDP sockets uses MQTT for messaging.

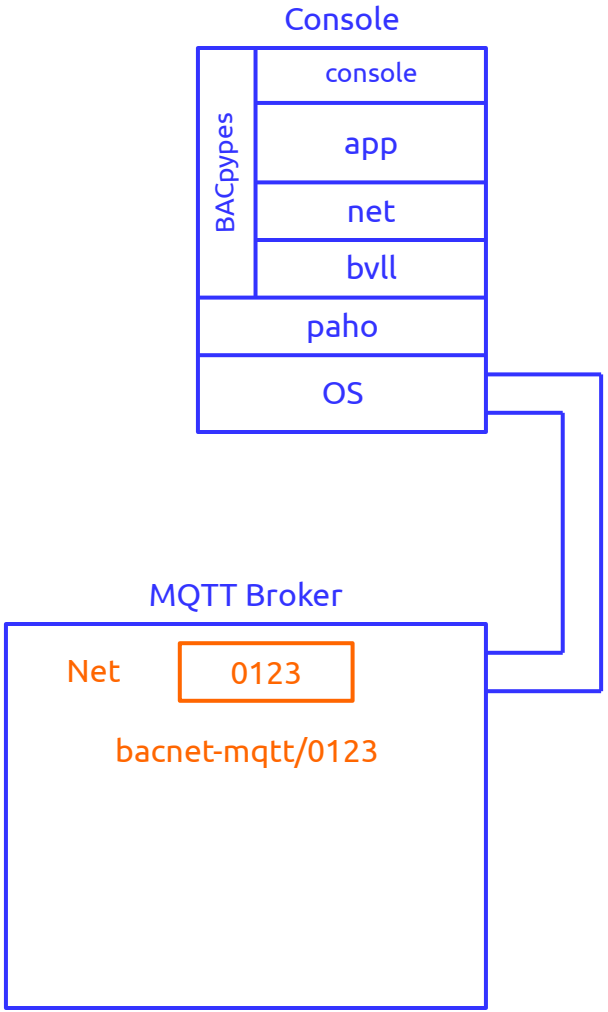


Like all BACnet applications, each of the layers in a stack is designed to exchange some content with its peers in other stacks. The application layer exchanges information about objects and properties using APDUs, network layers exchange topology information.

The BACnet Virtual Link Layer is similar to BACnet/IP.

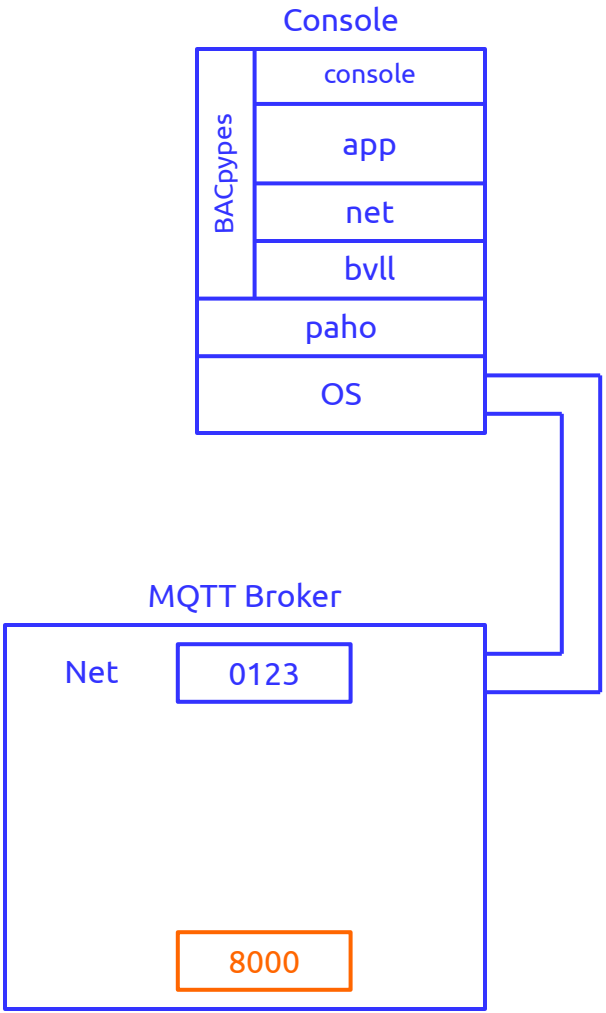


BACnet/MQTT peers are TCP clients of a **broker** which provides a **publish/subscribe** service organized into **topics**. MQTT provides few restrictions on topic names, no restriction on what clients can publish to which topics, or which clients can subscribe to topics. Some implementations provide that as an additional service, along with automatic failover and clustering.



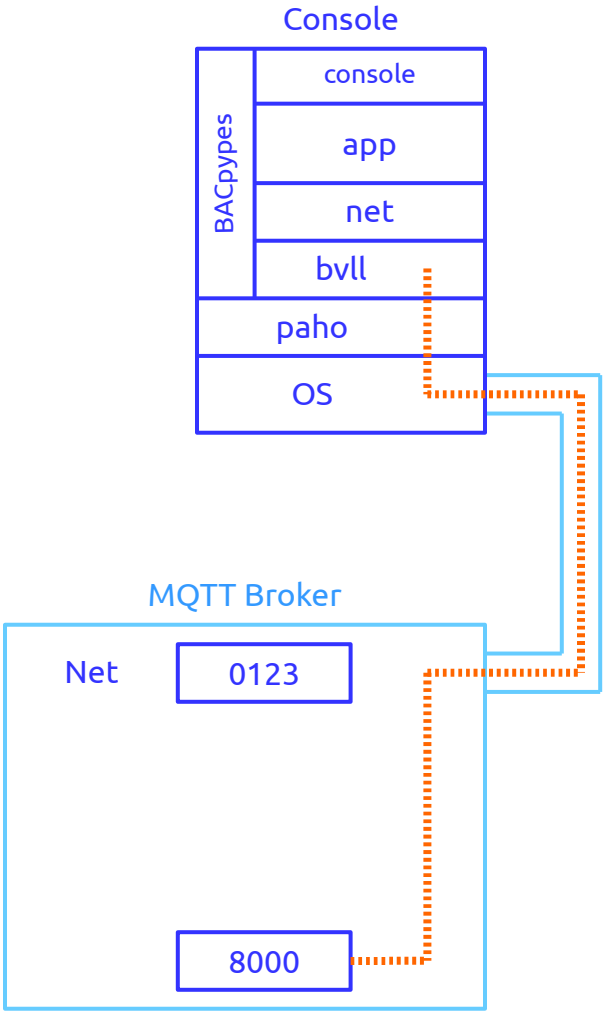
MQTT Topics are structured in a hierarchy similar to folders and files in a file system using the forward slash (/) as a delimiter. BACnet/MQTT divides that into a **network** and an **address**, where the network name is unrestricted and the address is the virtual MAC address of the client, lowercase hex encoded.

Topic names are case sensitive.

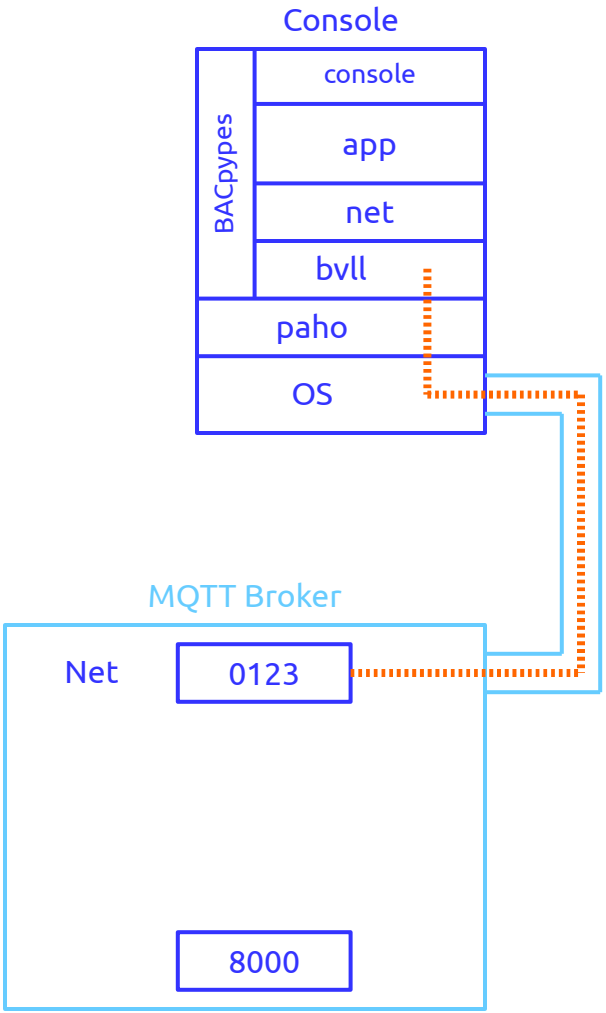


BACnet/MQTT reserves a special **broadcast topic** that is shared by all of the clients and functions as the broadcast MAC address for the virtual network.

It is encoded as X'80...' to match the broadcast address design pattern for VMACs. The address length is the same constant length for all virtual network addresses.

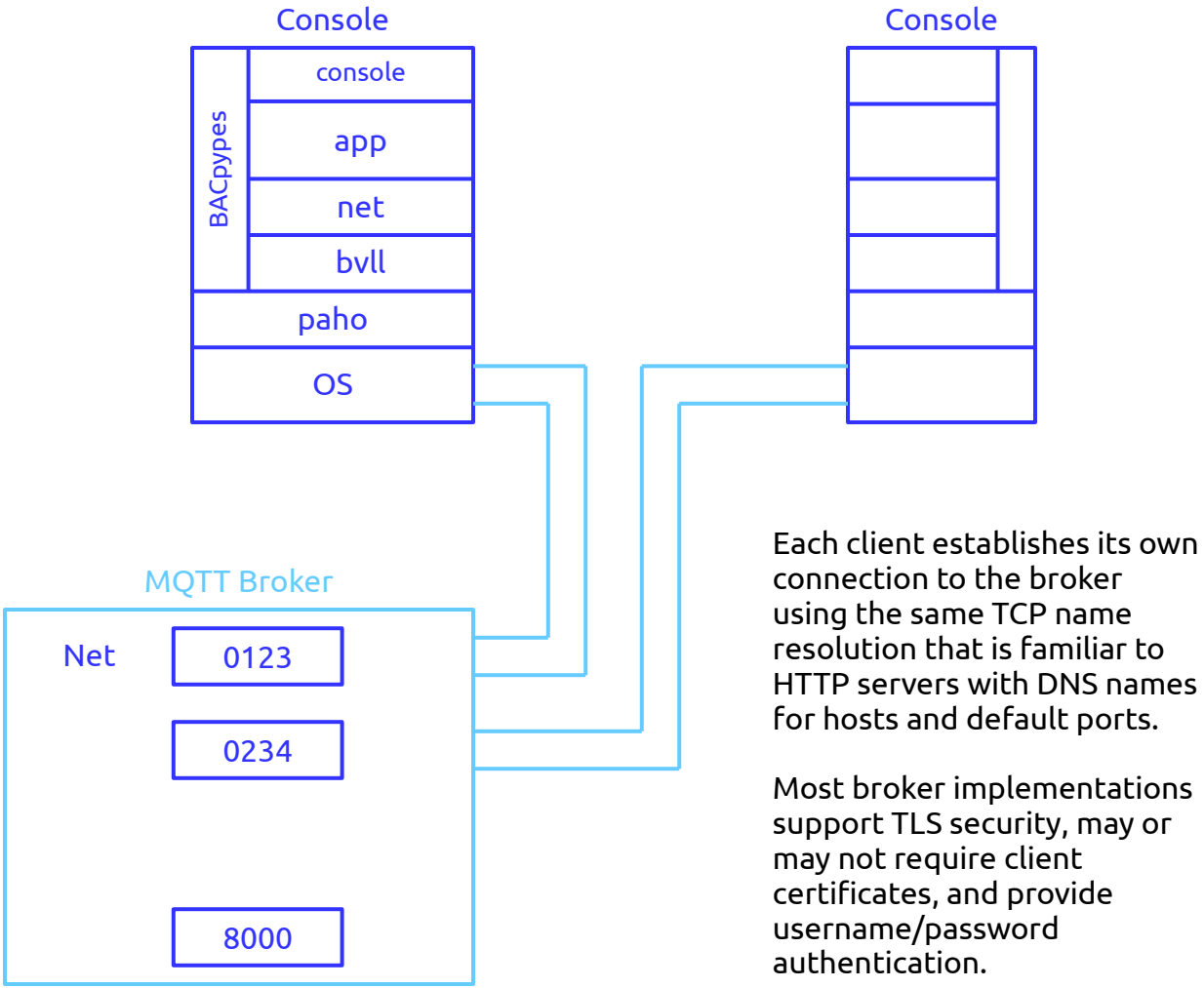


After establishing the connection, a BACnet/MQTT node subscribes to the broadcast topic. It may start receiving broadcast messages even before completing its setup process.



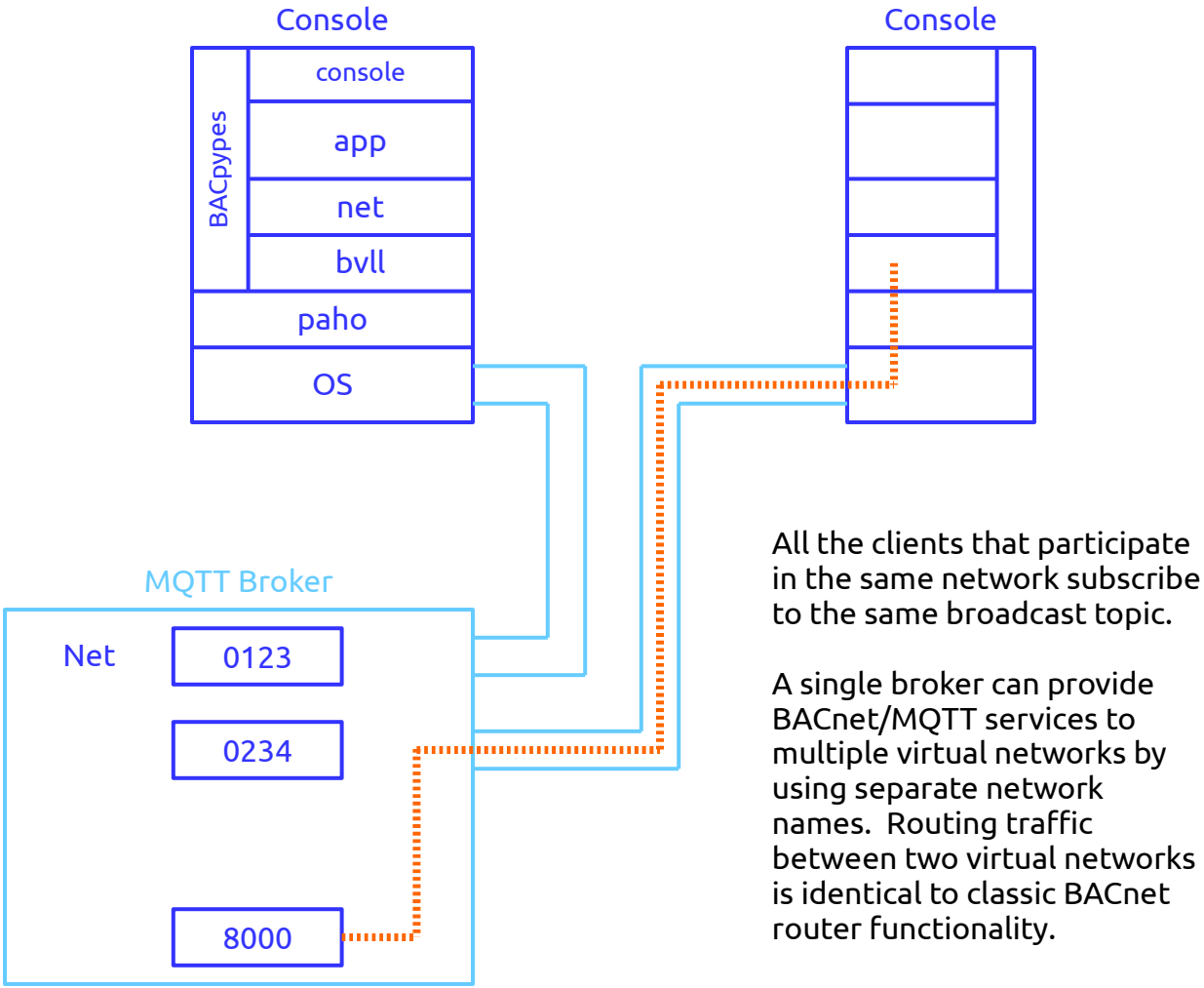
The node then subscribes to its **client topic** with a configured address. It is now ready to receive both broadcast and unicast traffic.

There is an additional service procedure and messages for assigning a randomly generated address.



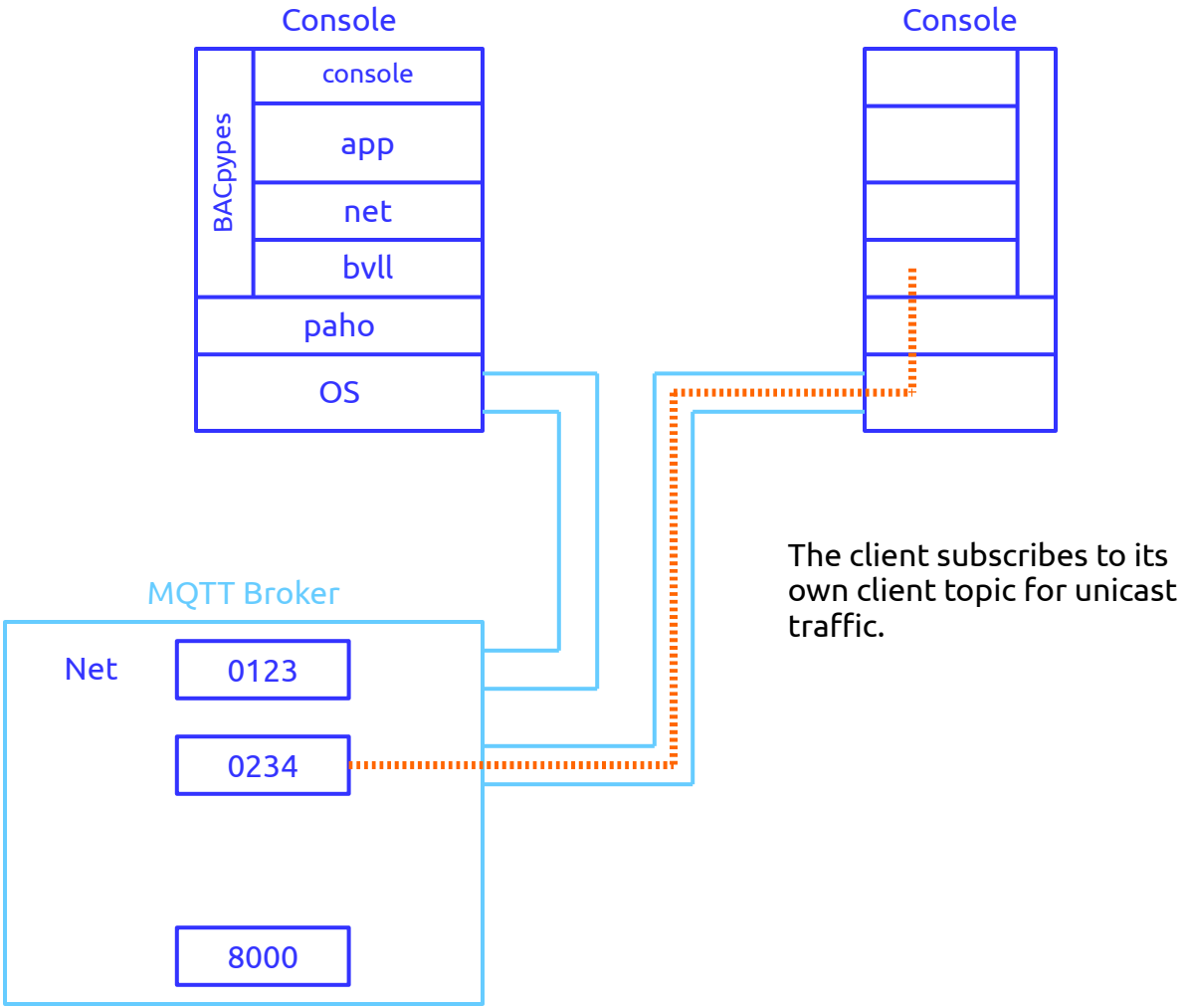
Each client establishes its own connection to the broker using the same TCP name resolution that is familiar to HTTP servers with DNS names for hosts and default ports.

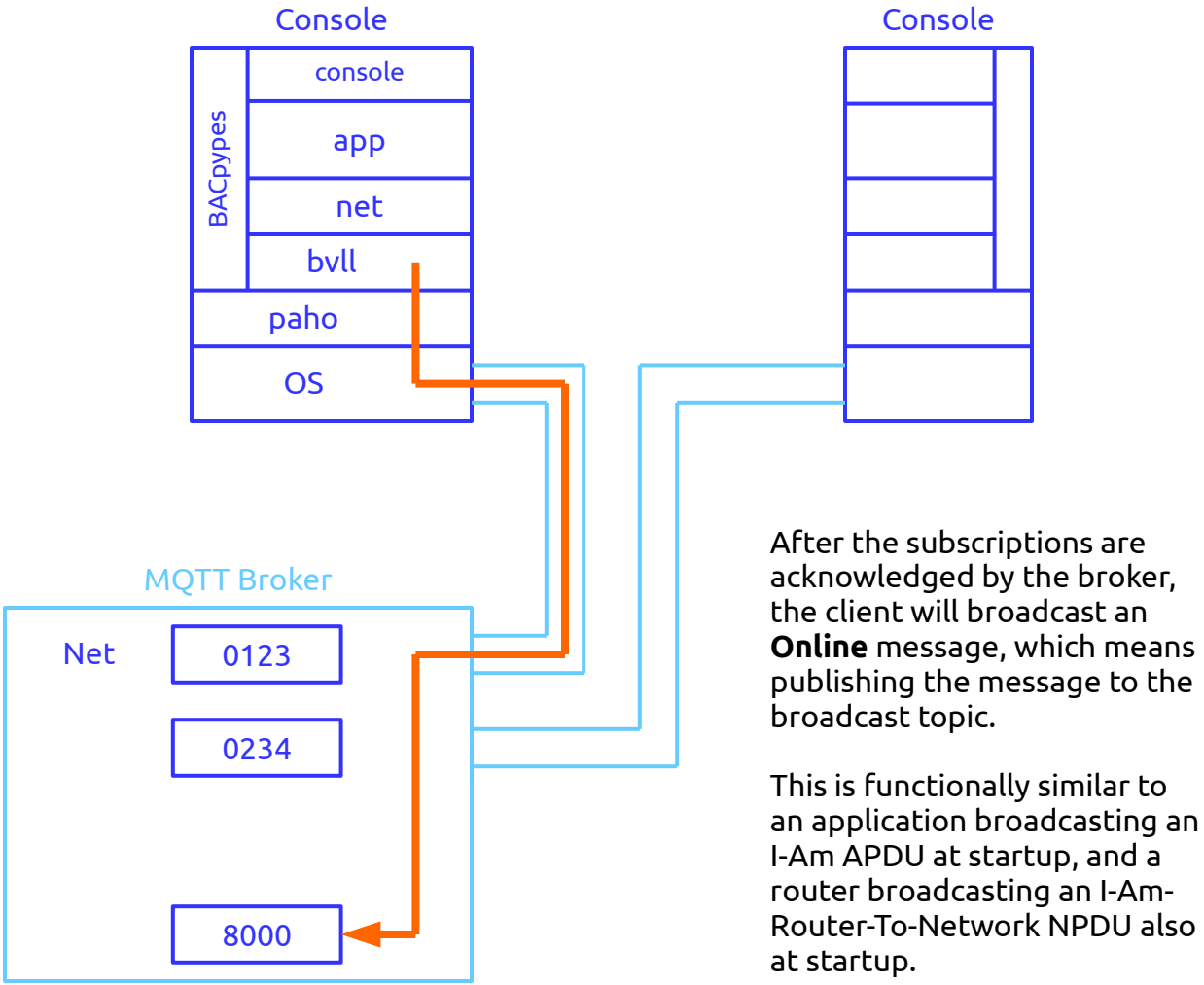
Most broker implementations support TLS security, may or may not require client certificates, and provide username/password authentication.



All the clients that participate in the same network subscribe to the same broadcast topic.

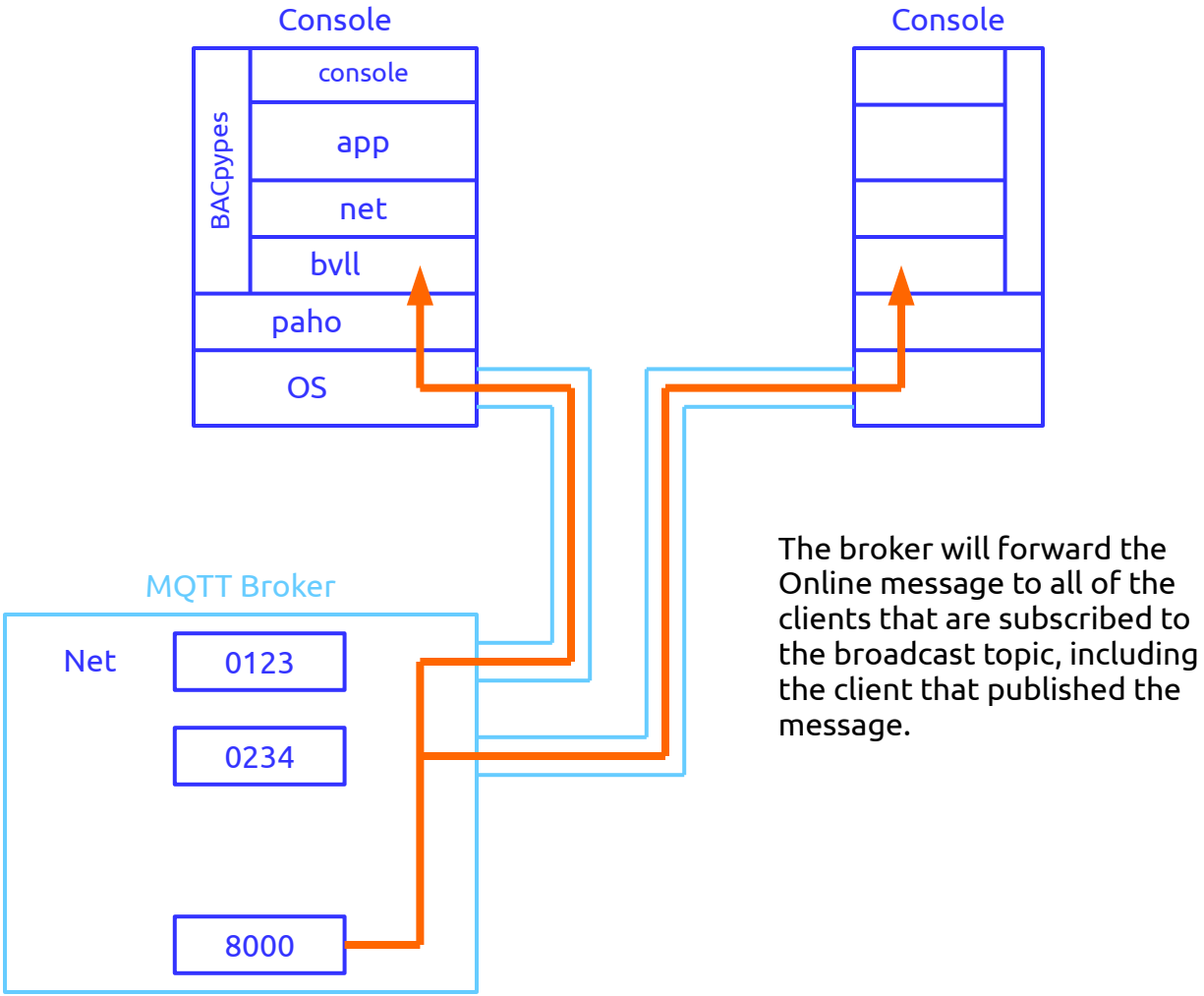
A single broker can provide BACnet/MQTT services to multiple virtual networks by using separate network names. Routing traffic between two virtual networks is identical to classic BACnet router functionality.

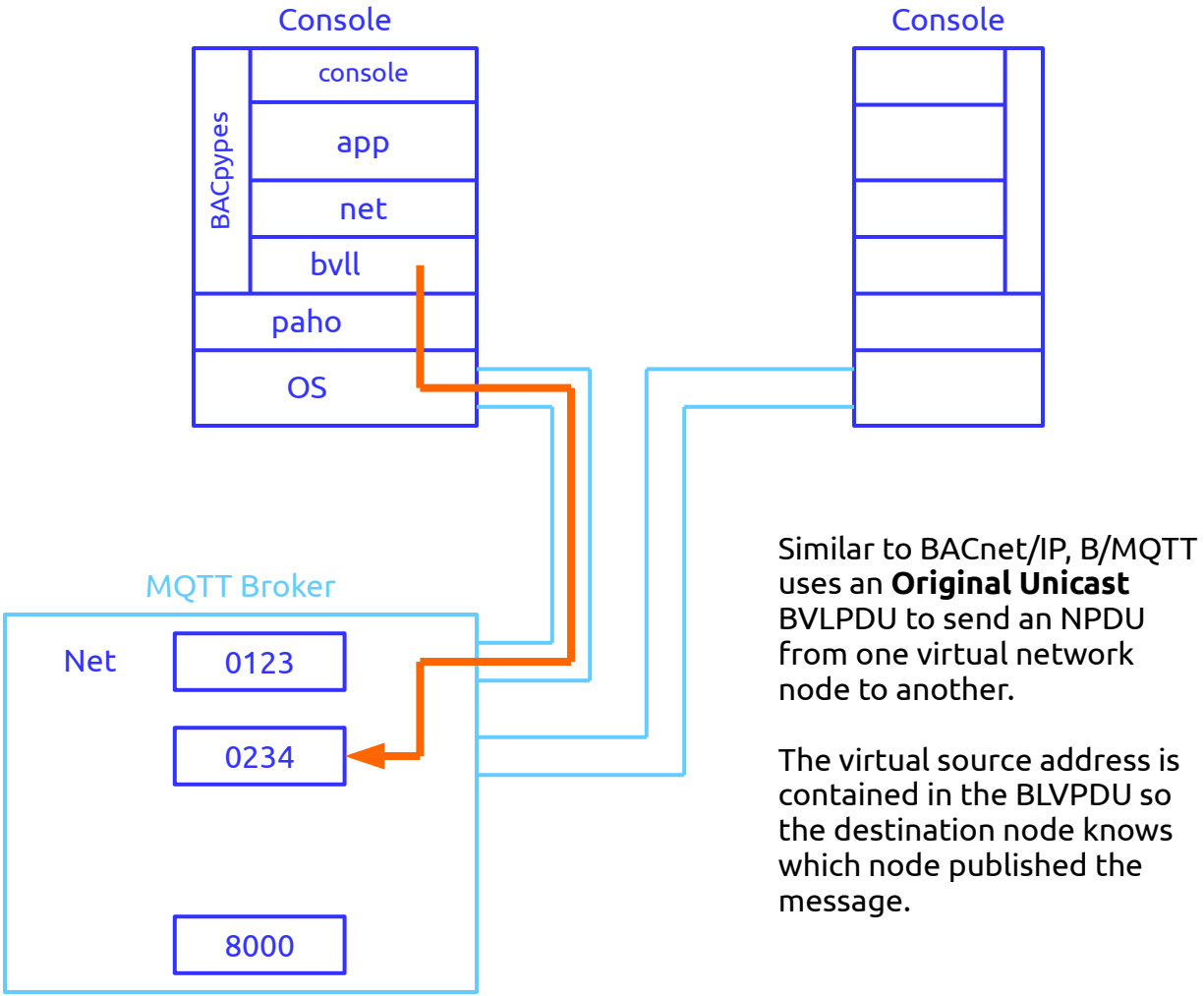

Console
console
app
net
bvll
BACpypes
paho
OS
Console
MQTT Broker
Net
0123
0234
8000
The client subscribes to its own client topic for unicast traffic.



After the subscriptions are acknowledged by the broker, the client will broadcast an **Online** message, which means publishing the message to the broadcast topic.

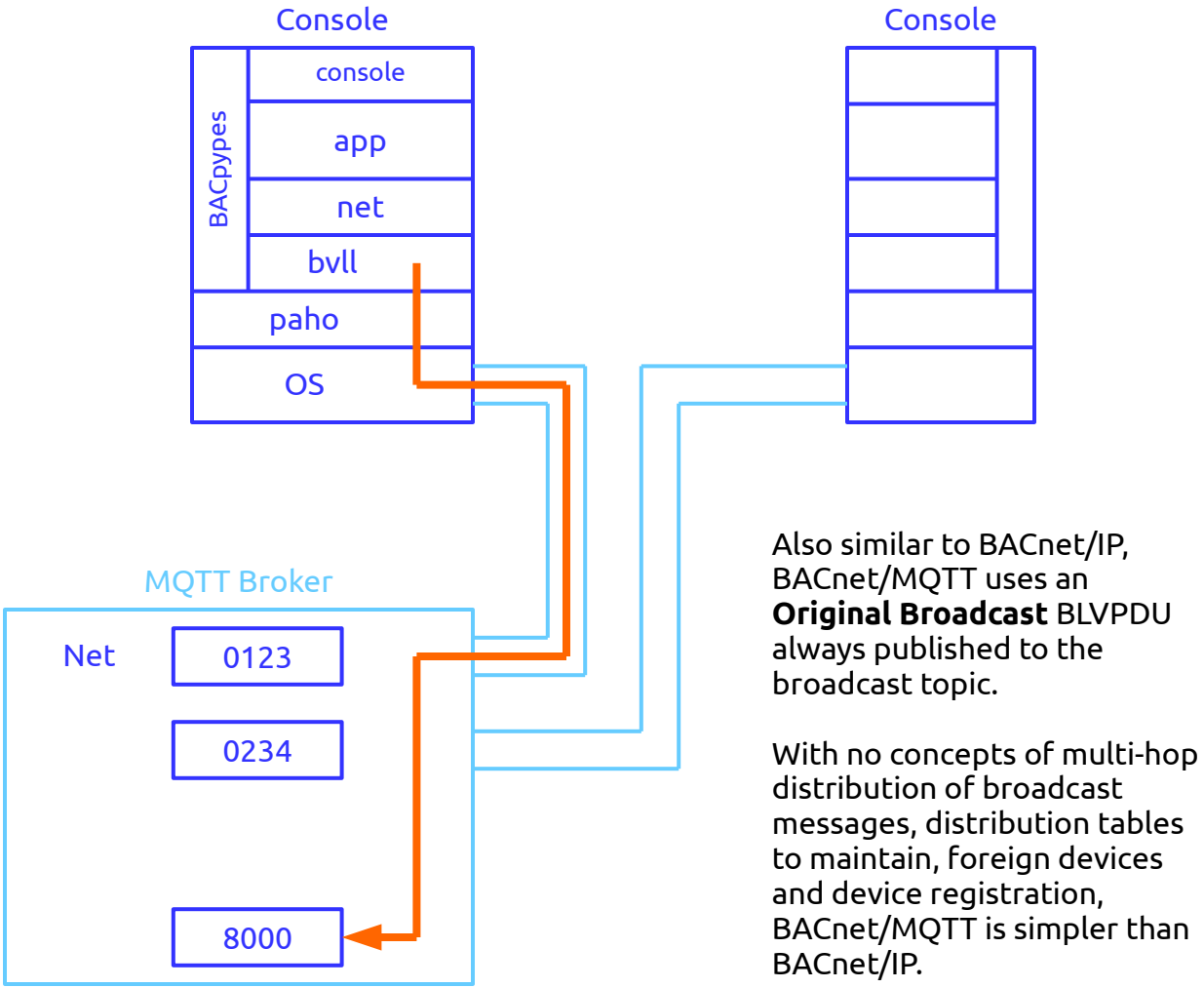
This is functionally similar to an application broadcasting an I-Am APDU at startup, and a router broadcasting an I-Am-Router-To-Network NPDU also at startup.


Console
console
app
net
bvll
BACpypes
paho
OS
Console
MQTT Broker
Net
0123
0234
8000
The broker will forward the Online message to all of the clients that are subscribed to the broadcast topic, including the client that published the message.



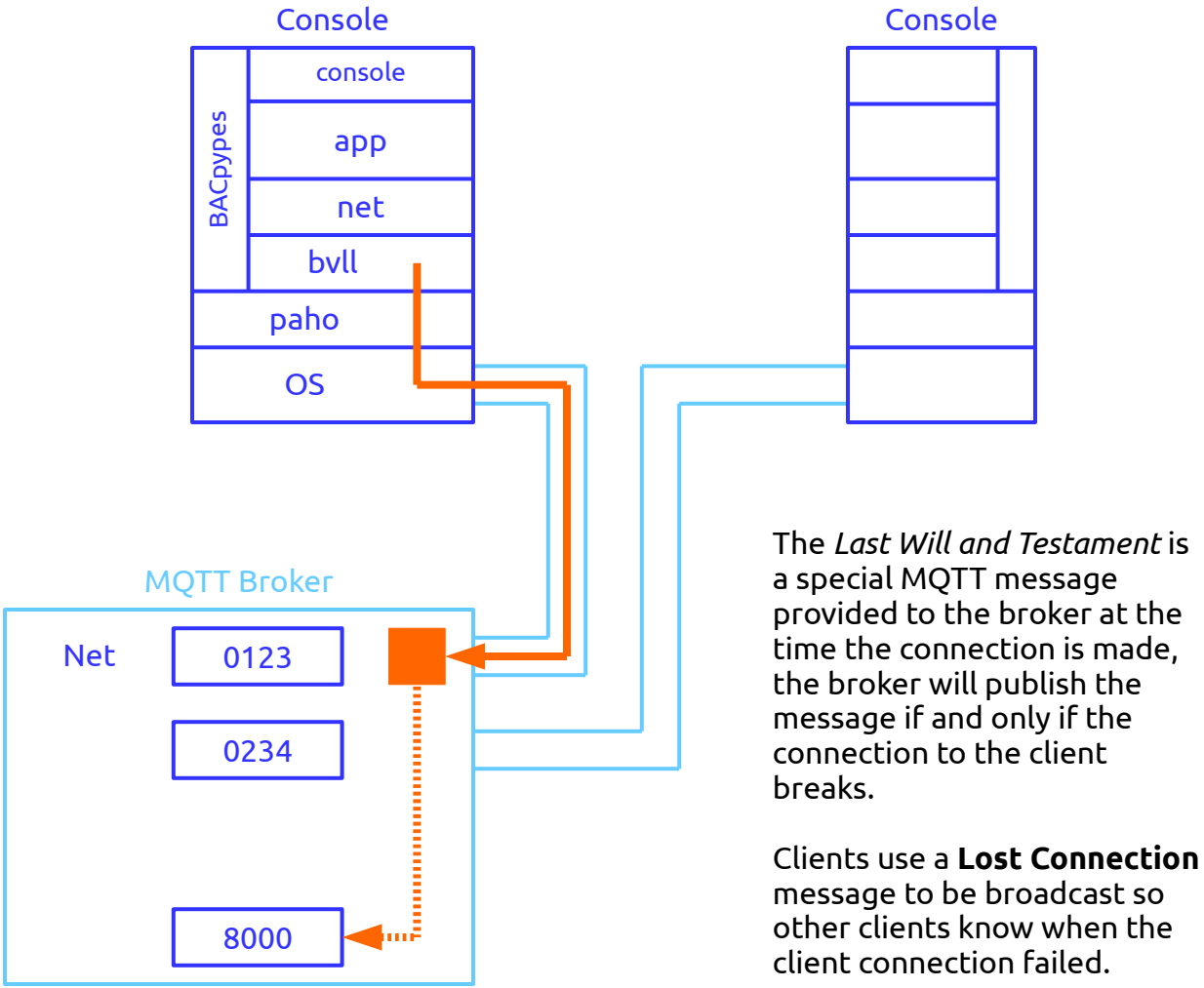
Similar to BACnet/IP, B/MQTT uses an **Original Unicast** BVLPDU to send an NPDU from one virtual network node to another.

The virtual source address is contained in the BLVPDU so the destination node knows which node published the message.



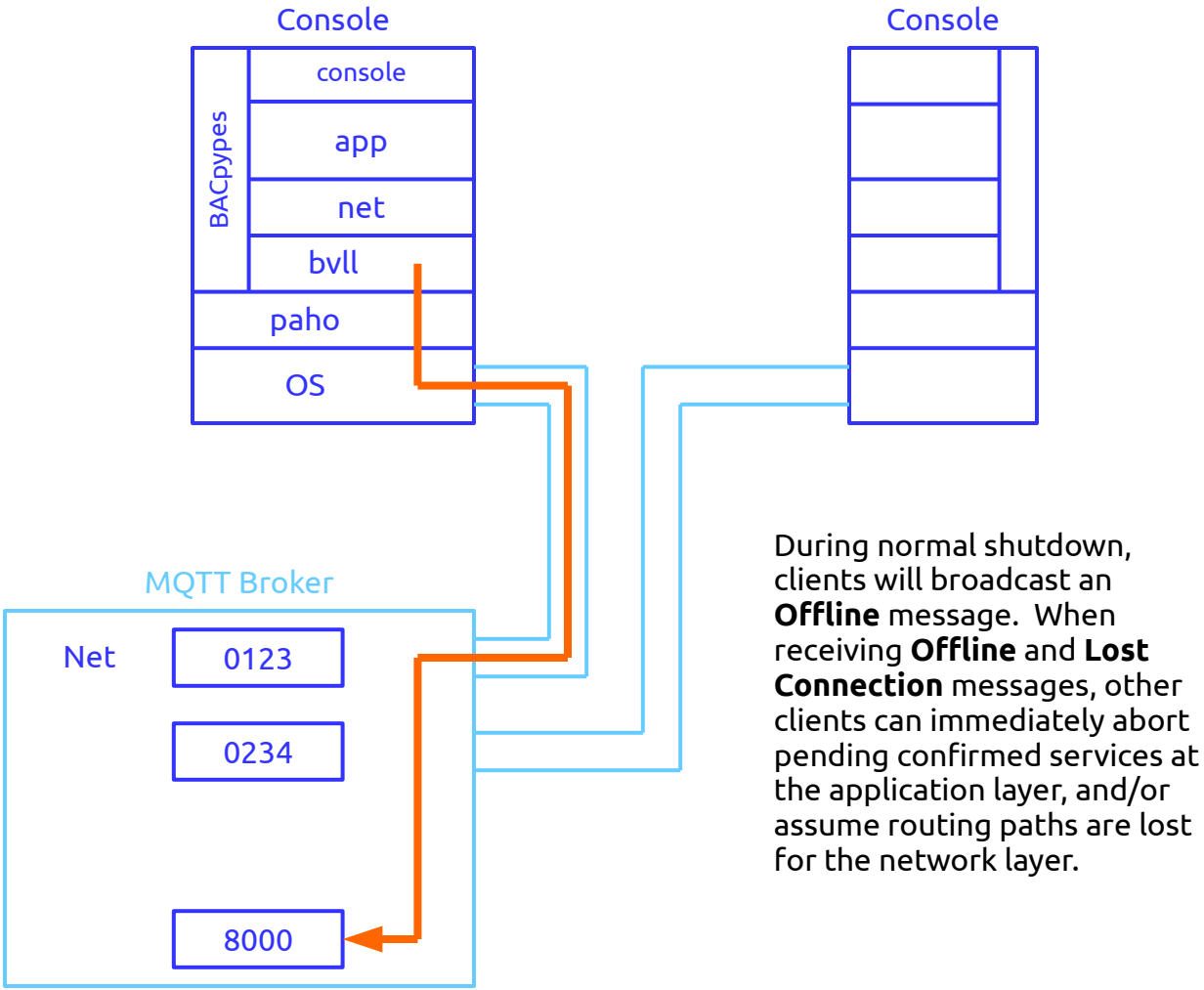
Also similar to BACnet/IP, BACnet/MQTT uses an **Original Broadcast** BLVPDU always published to the broadcast topic.

With no concepts of multi-hop distribution of broadcast messages, distribution tables to maintain, foreign devices and device registration, BACnet/MQTT is simpler than BACnet/IP.

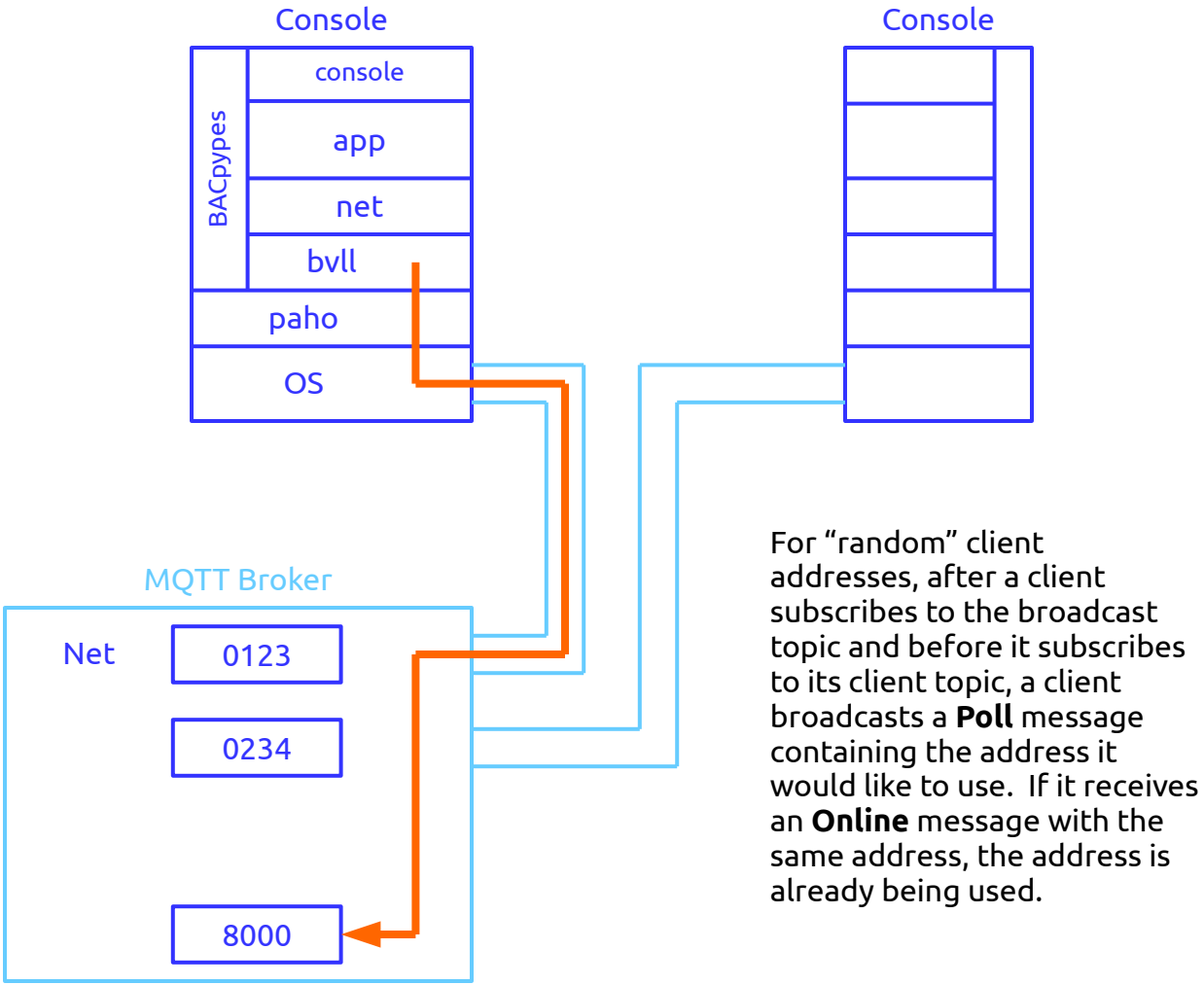


The *Last Will and Testament* is a special MQTT message provided to the broker at the time the connection is made, the broker will publish the message if and only if the connection to the client breaks.

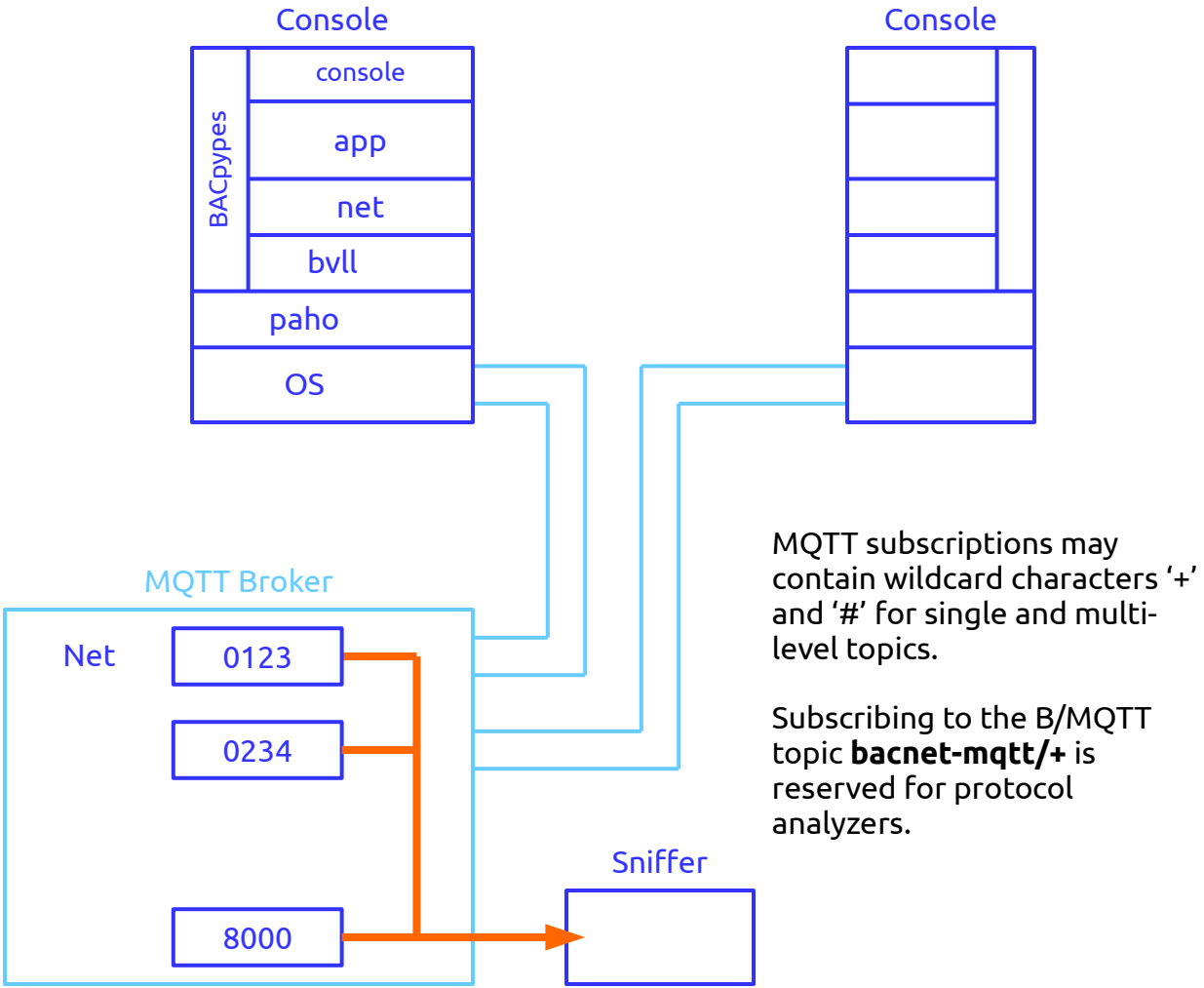
Clients use a **Lost Connection** message to be broadcast so other clients know when the client connection failed.

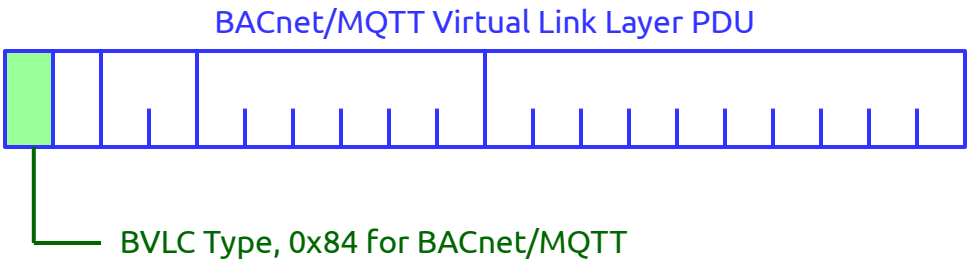


During normal shutdown, clients will broadcast an **Offline** message. When receiving **Offline** and **Lost Connection** messages, other clients can immediately abort pending confirmed services at the application layer, and/or assume routing paths are lost for the network layer.

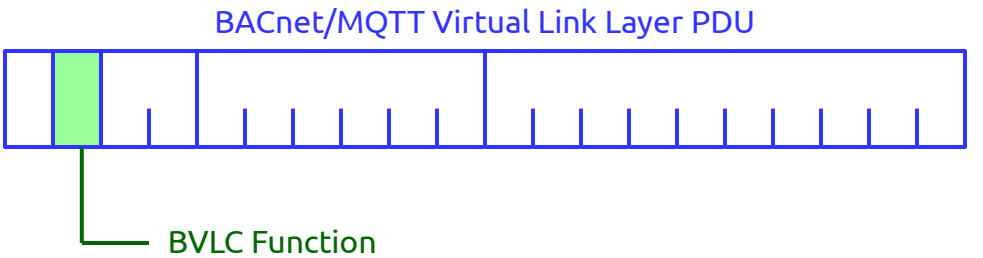


For “random” client addresses, after a client subscribes to the broadcast topic and before it subscribes to its client topic, a client broadcasts a **Poll** message containing the address it would like to use. If it receives an **Online** message with the same address, the address is already being used.


Console
console
app
net
bvll
BACpypes
paho
OS
Console
MQTT Broker
Net
0123
0234
8000
Sniffer
MQTT subscriptions may contain wildcard characters '+' and '#' for single and multi-level topics.
Subscribing to the B/MQTT topic **bacnet-mqtt/+** is reserved for protocol analyzers.

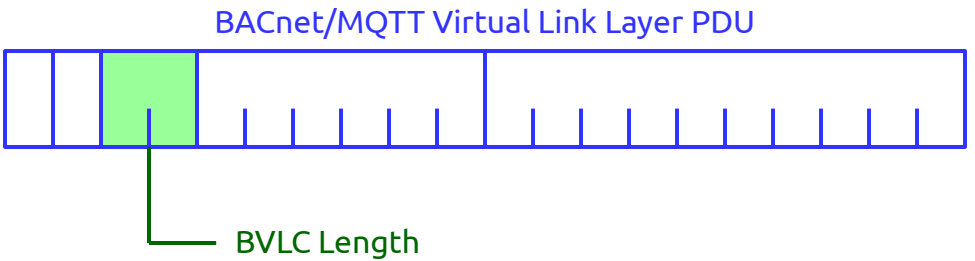


The BVLC Type field is present to be consistent with other BVLL packet formats including BACnet/IPv4, BACnet/IPv6, and BACnet/SC.

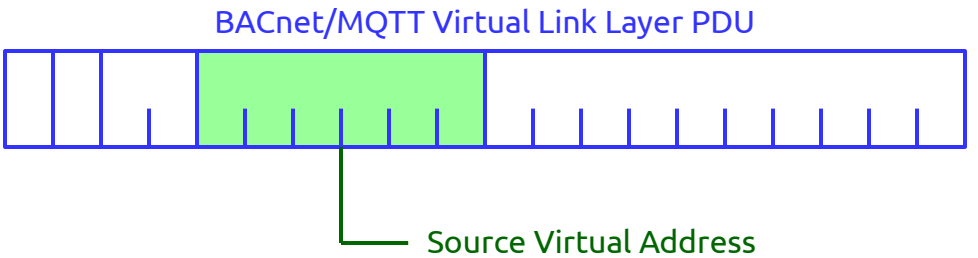


The BVLC Function code provides the message type:

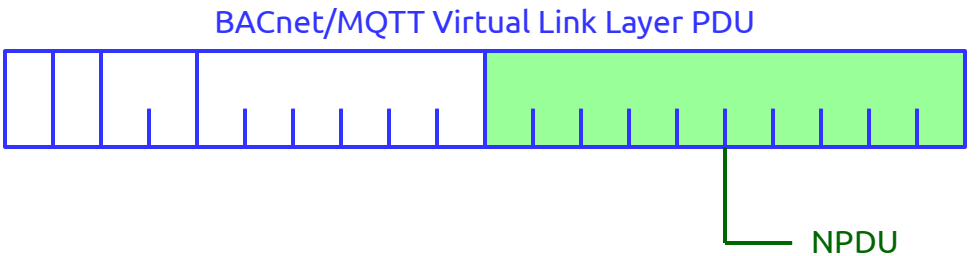
0x00	Result
0x01	Poll
0x02	Online
0x03	Offline
0x04	Lost Connection
0x05	Original Unicast
0x06	Original Broadcast



The BVLC Length is the self-included length of the entire BVLL packet, identical to the other BVLL formats.



The Source Virtual Address is the virtual MAC address for the client. It is unique for each client and cannot be the broadcast address. All virtual addresses are the same length, currently 6 octets.



The NPDU field is only used for **Original Unicast** and **Original Broadcast** messages and contains the complete NPDU beginning with the BACnet version number 0x01.