October 2021

ENHANCED HOST DISCOVERY IN SDN/FABRIC-BASED NETWORKS

Eric Levy-Abegnoli

Pascal Thubert

Patrick Wetterwald

Ratko Kovacina

ENHANCED HOST DISCOVERY IN SDN/FABRIC-BASED NETWORKS

AUTHORS:
Eric Levy-Abegnoli
Pascal Thubert
Patrick Wetterwald
Ratko Kovacina

ABSTRACT

Various solutions are provided herein to facilitate the efficient discovery of hosts in large network environments, such as software-defined networking (SDN) or fabric-based networks, utilizing several techniques. A first technique supports the ability to efficiently manage silent ports and silent media access control (MAC) addresses. This technique involves applying a novel heuristic to ports and MAC addresses, classifying such entities (as silent, quiet, and noisy), and intelligently polling such entities. A second technique supports a Multicast Listener Discovery (MLD)-based host discovery approach that is applicable to Internet Protocol (IP) version 4 (IPv4) and involves a host creating an IP version 6 (IPv6) address that embeds its IPv4 address, the addition of a well-known first byte to the three bytes in a Solicited-Node multicast address (SNMA), and the use of a form of unicast ping to confirm whether a host formed a derived address. A third technique involves using a service lookup for deterministic host discovery that involves the use of upper-layer discovery services to cause a host to expose its addresses in the replies to multicast discoveries.

DETAILED DESCRIPTION

In a traditional local area network (LAN), when two end nodes that share the same IP subnet – e.g., IP phones, servers, routers, etc. – need to communicate together for the first time, they broadcast a search packet (e.g., an Address Resolution Protocol (ARP) request in IP version 4 (IPv4) or a Neighbor Solicitation message in IP version 6 (IPv6)) to inform the target of their search and provide it with an opportunity to respond. Once the target has responded (in unicast) the communication may be established.

As LANs (which may also be referred to as broadcast domains, bridge domains, virtual LANs (VLAN), layer 2 segments, etc.) have grown in size to include a very large number of end nodes, the approach that was described above reached its limit. That is, too many broadcast searches can negatively impact the central processing unit (CPU) of each node and the bandwidth of the connecting links. With the inclusion of wireless paradigms, such broadcasts have become very harmful.

Additionally, another more serious issue may be realized when a broadcast search is seen by every node on a LAN, which makes it very easy for a malicious entity to obtain a map of a network. Such an entity can also respond to searches that are intended for others and, thus, can impersonate them, as illustrated below in Figure 1.

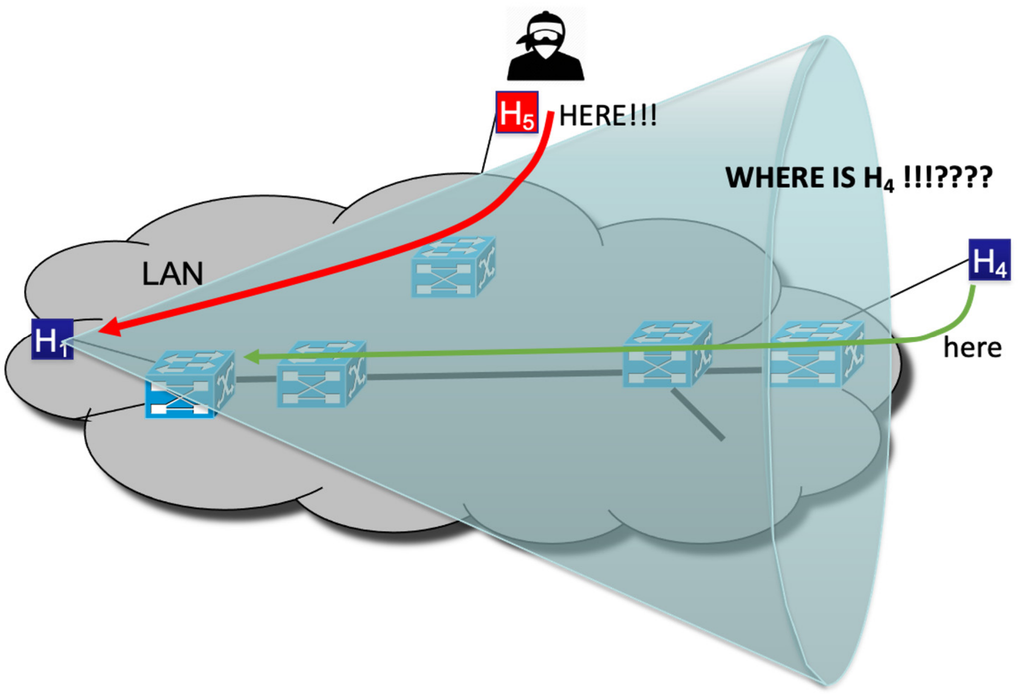


Figure 1: Exemplary Malicious Entity

The issue noted above is one of the main reasons why fabric-based networks have moved to a different model in which end nodes' IP addresses are stored in a host table and this table is used to stop searches at the edge of the fabric (i.e., a fabric edge or "FE"). By successfully searching the host table, the FE may respond with the found location or the

MAC address (i.e., a proxy operation) or it can transport the search to a destination in a unicast transmission (i.e., a relay operation). Most modern network deployments (such as, for example a software-defined access (which may be referred to herein as "SDA") environment, an Ethernet VPN (EVPN), an application-centric infrastructure (which may be referred to herein as "ACI"), etc.) employ such a technique to, in theory, prevent any layer 2 broadcast (IPv4) or multicast (IPv6) from circulating inside the fabric.

Figure 2, below, depicts elements of an exemplary environment as described above.

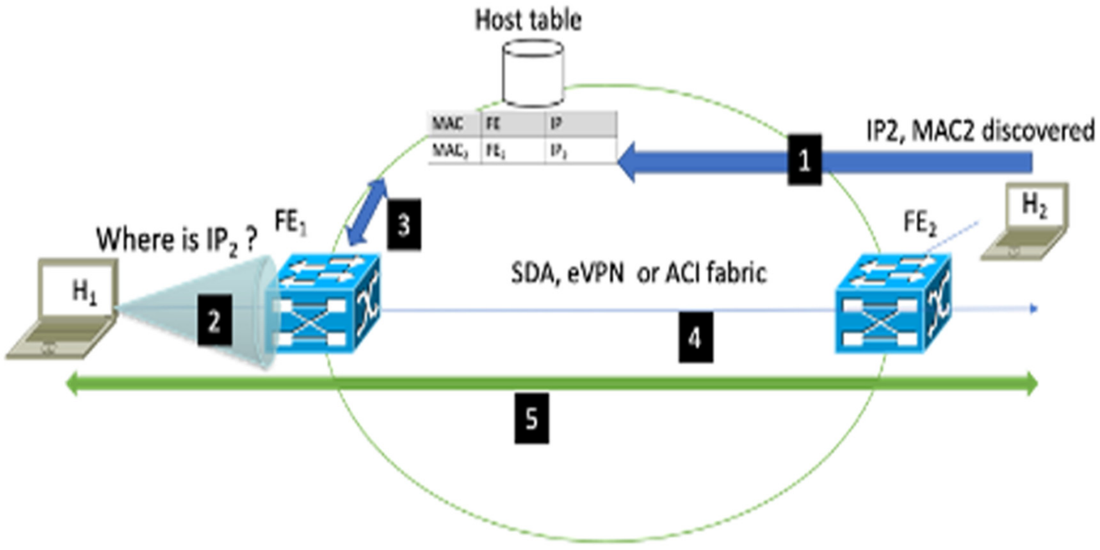


Figure 2: Illustrative Host Table Environment

Figure 2, above, identifies five exemplary steps (labeled 1 through 5), as follows:

1. Host H_2 is discovered and the binding <IP_2, MAC_2> is pushed into the host table.
2. Host H_1 searches for host H_2.
3. The search is intercepted on the fabric edge FE_1 and it is resolved through the host table.
4. The search is responded to by the fabric edge FE_1 or it is relayed in a unicast transmission to host H_2.
5. A session is established.

The main difference between the different fabric types (e.g., an ACI in a datacenter, an eVPN in a datacenter and on a campus, or an SDA for a campus) lies in the location of the host table. Such a table may be distributed (e.g., in an ACI setting) with a subset of it on each fabric edge, it may be replicated (e.g., in an eVPN setting) through Border Gateway Protocol (BGP) on each fabric edge, or it may be centralized (e.g., in an SDA setting) in a map server or map resolver.

The host table-based approach has been a significant improvement over the previous "always broadcast" approach. However, it has its limits, which can make it very fragile. Discovering a host's bindings (e.g., an IP address and a MAC address) is the keystone of the whole solution. If a binding is not discovered, a search (as depicted by Step 2 in Figure 2, above) will fail, and a session will not be established. Alternatively, in the case of a search failure the network will fall back to a broadcast approach.

Falling back to a broadcast approach may appear to be a corner case that does not affect the overall advantages of the host table approach, but, in reality, this reasoning is flawed. Any time that a search ends up as a broadcast it becomes an opportunity for a malicious node to impersonate the target. At the same time, as soon as broadcasting becomes an option in the fabric a malicious node can take advantage of it to overwhelm the network by sending a large number of fake searches (e.g., to targets that do not exist) that will all be broadcast. These issues have been taken seriously enough that some fabrics (e.g., an SDA setting) prefer to disallow a link-level broadcast completely, at the expense of failing to connect peers that are not known in the host table.

It is important to consider the reasons why a host discovery might fail. The discovery is essentially based on snooping traffic – comprising control traffic (such as, for example, Address Resolution Protocol (ARP) communications, neighbor discovery (ND) communications, and Dynamic Host Configuration Protocol (DHCP) communications) or data traffic – that is sent by hosts. Such snooping depends entirely on the good will of the host to show a "sign of life" from each of its IP addresses.

In reality, a host could be silent and only respond to solicitations. However, that will not occur because such solicitations in turn depend upon the IP address being known in the host table.

Alternatively, a host could be quiet. For example, it may take some time before a host shows signs of life, which will delay the moment where it is discovered and becomes reachable. In such a case, it slows down dramatically the establishment of sessions between peers (to the moment of host discovery) and it is often interpreted as the network being broken.

Another kind of issue can be seen with IPv6. In IPv6, there are multiple addresses per host, each address having a different scope. Some of the addresses (e.g., link-local) are heavily used on a LAN while other addresses (e.g., global) tend to be quieter. A host table may therefore have the link-local addresses but not the global addresses, which are precisely the addresses that a remote peer would want to reach.

Finally, states tend to be forgotten, including host table entries. For example, entries could age, devices storing the host table could reboot, a host could move to a location where the host table is not accessible, etc. A host that showed a sign of life, perhaps when the entry was first assigned, will eventually remain quiet after that until it is searched. If the corresponding host table entry goes away it may take a long time before the network is able to rediscover the host.

A Multicast Listener Discovery (MLD)-based approach may be employed to strengthen various of the host discovery issues that were described above. IPv6 requires that a host that forms an IPv6 address send a MLD report and later respond to MLD queries for the Solicited-Node multicast address (SNMA) that derives from the IPv6 address. The SNMA has the last three bytes of the IPv6 address encoded and operates at the scope of a link (e.g., FF02: ... last three bytes). Usually there is a one-to-one relationship between a SNMA and unicast addresses due to the rarity of collision of the last three octets (e.g., the birthday paradox).

This is 1) effectively implemented by the protocol stacks and 2) mostly useless since there is no layer 2 multicast operation associated to the multicast group, and MLD snooping is usually not used for link-scoped multicast communications for scalability reasons. However, it is there and aspects of the techniques presented herein, which will be described and illustrated in the narrative that is presented below, repurpose it to discover silent nodes.

Aspects of the techniques presented herein place MLD capabilities in a first hop security context (e.g., in a switch integrated security feature (which for convenience may be referred to herein as “SISF”) component that is in charge of address discovery and tracking) whereby a SISF listens to MLD reports. A SISF may act as yet another snooper of MLD activity besides the classic MLD snooping function. It may also generate queries if it acts as a layer 3 switch or if it can impersonate a router. In such a case the query may be sent in MAC-unicast form to observe a particular MAC address.

According to aspects of the techniques presented herein, a SISF matches the SNMA that it observes from a MAC address and the bindings that it has for that MAC address. An SNMA with no associated binding indicates a silent node, but at that point it is not known which address is missing (just the last three bytes of the missing address). In that very rare case, a SISF may inject a forged unicast address in the fabric (e.g., a mapped address that is recognizable as encoding the SNMA and yet can be presented as a unicast MAC or IP mapping in the fabric overlay).

Upon an address lookup for which there is no match in the overlay (e.g., in eVPN BGP tables), a SISF performs a second lookup that searches for a mapped address that has the indicated last three bytes. If such an address is found, the lookup is sent in unicast form to the matching MAC address.

Figure 3, below, depicts elements of an exemplary enhanced environment as described above.

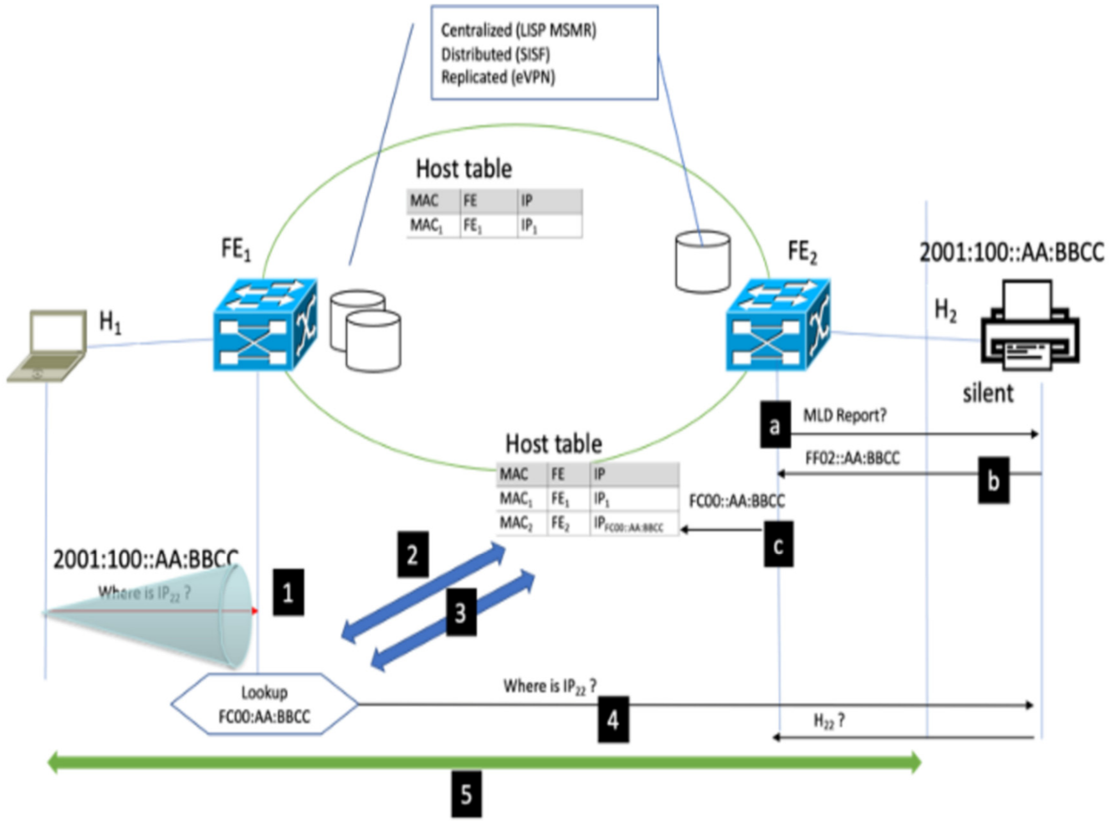


Figure 3: Illustrative Enhanced Environment

In the environment that is depicted in Figure 3, above, a host address has not, for any number of reasons (including, for example, a host being silent, quiet, forgotten, etc.), been discovered. The network proceeds to proactively discover it through the three exemplary activities that are labeled 'a', 'b', and 'c' in Figure 3, above. During activity 'a', the network FE_2 issues a request for an MLD report. During activity 'b', host H_2 provides a list of SNMA values. Finally, during activity 'c', the SNMA is pushed to the host table

Subsequently, when host H_1 needs to establish a session with host H_2 the five exemplary steps that are labeled 1 through 5 in Figure 3, above, may be performed as follows:

1. Host H_1 sends a search for the address IP_2 which is intercepted by its point of attachment (i.e., the fabric edge FE_1).
2. The fabric edge FE_1 first checks in the host table for the address that is being requested (i.e., IP_2, 2001:100::AA:BBCC).

3. The lookup fails so the fabric edge FE_2 then searches the host table instead for the corresponding SNMA (i.e., FC00::AA:BBCC). Since this information was proactively learnt during activities a through c (as described above), the host table returns the fabric edge FE_2.
4. The search for the address IP_2 (that was received in Step 1, as described above) is forwarded to MAC_2 which is attached to FE_2.
5. The session is established.

To address the types of challenges that were described above, various solutions are provided herein through several techniques. Each of the techniques may operate within a software-defined networking (SDN)- or fabric-based network and each of the techniques will be discussed and illustrated in the narrative that is presented below. A first technique supports the ability to efficiently managing silent ports and silent MAC addresses. A second technique supports an MLD-based host discovery that is applicable to IPv4. A third technique involves the use of a service lookup for deterministic host discovery.

Turning to the first technique, aspects of this technique support the ability to efficiently managing silent ports and silent MAC addresses in an SDN- or fabric-based network.

Frequently probing on all of the ports with an MLD query may overwhelm the CPU of the fabric edge. Additionally, when the port is a wireless tunnel, or simply a trunk interface that is going to a set of wireless devices, the probing, sent as a layer 2 multicast (i.e., the layer 2 destination is 33:33:0:0:0:1) packet, can be very harmful over the air. Aspects of the first technique address such a problem.

Aspects of the first technique classify silent and noisy ports, as well as silent and noisy devices. In support of such a classification process, the following definitions are important:

- A silent port is a wired port of the FE that is up, but which has no known MAC or IP address bound to it in the host table. Note that an SNMA address is not treated as an IP address in this definition.
- A quiet port is a wired port where a mismatch was detected between the list of known IP addresses that are behind it and the list of SNMA groups.

- A noisy port is a wired port where there is a perfect match between the known IP addresses and the known SNMA groups that are behind it.
- A silent MAC is a known device MAC (in the host table) with no known IP addresses bound to it (sometimes referred to as a "standalone MAC"). Note that an SNMA address is not treated as an IP address in this definition.
- A quiet MAC is a device MAC where a mismatch was detected between the list of known IP addresses that are behind it and the list of SNMA groups.
- A noisy MAC is a device MAC where there is a perfect match between the known IP addresses that are bound to it and the known SNMA groups that are behind it.

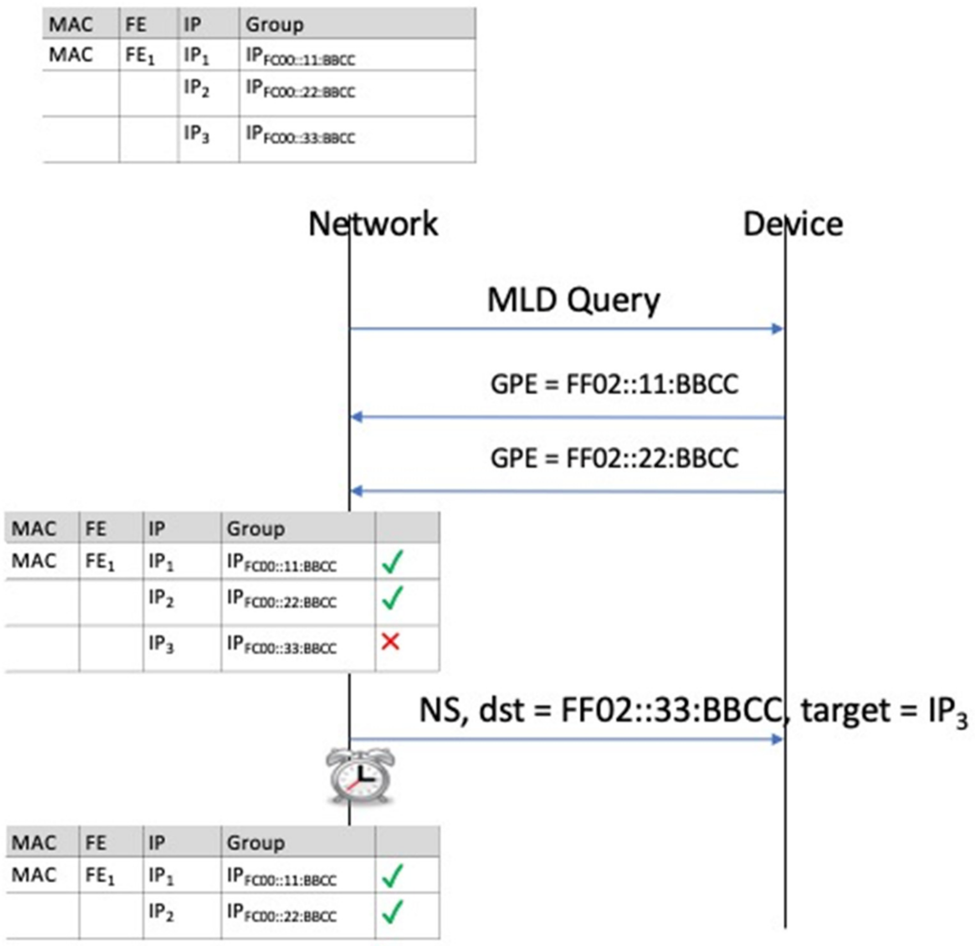
According to aspects of the first technique, a suite of heuristics may be applied to the above definitions. For example:

- As soon as an access port (e.g., device facing) comes up, it is added to the list of silent ports.
- As soon as a wireless device is associated, it is added to the list of silent MAC addresses.
- As soon as a wired IP address is discovered (e.g., discovery happens as part of the normal address discovery mechanisms), the port that it is bound to is removed from the list of silent port and it is added to the list of quiet ports.
- As soon as a wireless IP address is discovered, the MAC address that it is bound to is removed from the list of silent MAC addresses and it is added to the list of quiet MAC addresses.
- Whenever the received MLD reports (always from a given MAC address, and over either an access port or a wireless access tunnel) become consistent with the host table for the instant MAC address or port (i.e., the entry "#groups #known IP" in a host table), the MAC address or port are removed from the list of silent MAC addresses or ports and added to the list of noisy MAC addresses or ports.
- MLD reports are aggressively probed (e.g., every 10 seconds) on silent ports or silent MAC addresses. The probing period may be increased on quiet MAC

addresses or ports and it may be made even longer (e.g., every minute) on noisy MAC addresses or ports.

As long as the number of IP addresses behind a MAC address or an access port is from the number of groups, the MAC address or port are considered to be quiet (or silent if no IP address has been discovered). When the numbers become the same, and there is a match between each IP address and each group (i.e., the same last three digits), the MAC address or port are considered to be noisy.

When there is an IP address that does not have a group match, a specific algorithm may be exercised. For example, the IP address may be queried using a name server (NS) lookup (that is sent to the group address) and if no response is received the address may be removed from the host table. Such an approach is illustrated in Figure 4, below.



MAC	FE	IP	Group
MAC	FE_1	IP_1	$IP_{FC00::11:BBCC}$
		IP_2	$IP_{FC00::22:BBCC}$
		IP_3	$IP_{FC00::33:BBCC}$

MAC	FE	IP	Group	
MAC	FE_1	IP_1	$IP_{FC00::11:BBCC}$	✓
		IP_2	$IP_{FC00::22:BBCC}$	✓
		IP_3	$IP_{FC00::33:BBCC}$	✗

MAC	FE	IP	Group	
MAC	FE_1	IP_1	$IP_{FC00::11:BBCC}$	✓
		IP_2	$IP_{FC00::22:BBCC}$	✓

Figure 4: Exemplary Host Table Management

Turning to the second technique, as referenced above, aspects of the second technique support an MLD-based host discovery that is applicable to IPv4 in an SDN- or fabric-based network.

The MLD procedure that was described above applies to IPv6 but there is no such multicast operation in IPv4. With IPv4, this is typically less of a problem since a stack usually creates a single address through DHCP and that may be observed in a relay. However, there are cases in which the node will dynamically create sub-interfaces or secondary addresses that may be missed.

Aspects of the second technique employ a variation of 464XLAT (see the Internet Engineering Task Force (IETF) Request for Comments (RFC) 6877) or other forms of IPv4-mapped IPv6 address to bridge the gap. It is important to note that the Customer-side transLATor (CLAT) function is generally available on most operating systems (OSs), typically to access Wireless wide area network (WWAN) interfaces. This is the case, for instance, with the Windows 10 Creators update. However, equipment manufacturers seem to resist this trend. The CLAT-like operation that is required under aspects of the second technique is not the full CLAT in 464XLAT, since the client does not perform network address translation (NAT) on the IPv4 traffic to yield IPv6 artifacts. Rather, the client creates an IPv6 address that embeds or somehow matches the IPv4 address and uses that IPv6 address as a source for its IPv6 traffic, if desired.

Aspects of the second technique support a function that automatically configures an IPv6 address on the interface of the client, such that the IPv6 address embeds the IPv4 address. The simplest way that this may be accomplished is to use a method in the art where the last four bytes of the IPv6 address are the IPv4 address. The /96 that embeds the IPv4 address can be well-known such as, for example, all zeroes. It can also be a /96 that is provided by the network in router advertisements, Authentication, Authorization and Accounting (AAA), or DHCP. Either way, it is expected that the /96 is known to the network such that an IPv6 address can be recognized as potentially embedding an IPv4 address.

Once the IPv6 address is installed on the interface, it can be observed using aspects of the procedure that was described above. The presence of an address that has the last three bytes of the IPv4 address is known. What is not known is whether this is an IPv4-embedded address or a native IPv6 address.

Whether the client employs a private or a public address, the first octet in the IPv4 address is typically well known (e.g., 10 or 192 for a private address, 9 if the client is within IBM’s network and the address is taken from IBM's class A, etc.) and there can be only a very limited number of such first octets in a network.

If 1) the IPv4 address is embedded as the last four octets of the IPv6 address (as most IPv4 embedding techniques realize) and 2) the first octet is well-known as discussed above, then the last three octets of the SNMA combined with the well-known first octet provide a full indication of what the IPv4 address is (or what the few IPv4 addresses can be). As opposed to IPv6, the possible IPv4 address(es) can be fully recovered by the concatenation of the well-known first octet and the three octets in the SNMA.

To determine if the SNMA concerns an IPv4 address, aspects of the second technique add a step where a router may ping the address that is generated by the concatenation of the well-known first octet and the last three bytes in the SNMA. The ping may be of type MAC-unicast to the MAC address of the node that sent the report. The ping is not performed if the node is found to expose an IPv6 address that matches the last three bytes of the SNMA but not the /96 that is used for CLAT operation. If the ping responds, then the IPv4 address is discovered and the unknown SNMA address may be resolved so that it can be removed from the list of unknown values.

Turning to the third technique, as referenced above, aspects of this technique support the use of a service lookup for deterministic host discovery in an SDN- or fabric-based network.

A number of upper-layer services respond to well-known multicast addresses. For example, the Simple Service Discovery Protocol (SSDP) for Universal Plug and Play (UPnP) responds to FF0X:0:0:0:0:0:0:C, the Unified Fabric Management Protocol (UFMP) responds to FF0X:0:0:0:0:0:0:15F, the Service Location Protocol (SVRLOC) uses FF0X:0:0:0:0:0:0:116, and the Multicast Domain Name System (mDNS) responds to FF0X:0:0:0:0:0:0:FB. Note that in the previous examples the value “X” identifies a scope.

For IPv6 the complete range may be found in the “IPv6 Multicast Address Space Registry” from the Internet Assigned Numbers Authority (IANA).

Services like mDNS are typically provided by silent servers such as a printer, services like UPnP are typically provided by different types of hosts, services like UFMP can be used to discover data center network interface controller (NIC) cards, and services like SRVLOC have a wide range of applications.

In order to discover an address of a given scope, aspects of the third technique leverage the source address selection rules in IPv6 that cause a source to match the scope or a request in a response. In particular, aspects of the third technique suggest that the first hop router send MAC layer unicast IPv6 multicast queries using one of the IPv6 multicast discovery protocols. The scope of a query X is chosen based on the scope of the address that is expected. If there is at least one link-local address that is currently known, a Unique Local Address (ULA) or a Global Unicast Address (GUA) is expected so a scope that is greater than two (i.e., larger than the link-local count) is employed.

For UPnP, to be found by a network search a device must send a unicast User Datagram Protocol (UDP) response to the source IP address and port that sent the request to the multicast address. Devices respond if the search target (ST) header field of the M-SEARCH request is "ssdp:all," "upnp:rootdevice," and "uuid:" followed by a Universal Unique Identifier (UUID) that exactly matches the one advertised by the device or if the M-SEARCH request matches a device type or service type that is supported by the device. Thus, if the UUID is already known from a link-local exchange it can be used with a larger scope for a global multicast. If it is not known, then a series of typical services may be queried.

Aspects of the third technique employ heuristics in the art to detect the type of device (e.g., based on link-local activity, based on the fact that it is fully silent (like a printer would be but a sensor would not), etc.). Multiple protocols may be tried, depending upon how precisely the type of device is known. The order in which the protocols are tried may be chosen based on the probability of determining the type of a device.

What is important to note is that whichever discovery protocol is employed, the device will answer from a unicast address at which point the address is discovered. When using aspects of the third technique a first hop router may listen for a particular protocol (such as, for example, UPnP) and send a sequence of (IP multicast) discovery requests to the target device as MAC unicast communications. If the target device replies, the address may be discovered.

Note that aspects of the third technique work for any address family since the trigger for the discovery is an upper-layer protocol. For example, the IPv4 equivalent of the UPnP request that was described above is an IPv4 multicast M-SEARCH, using the message format that is shown below where the elements that are in italics are placeholders for actual values:

`M-SEARCH * HTTP/1.1 HOST: 239.255.255.250:1900 MAN: "ssdp:discover" MX:` *`seconds to delay response`* `ST:` *`search target`* `USER-AGENT:` *`OS/version`* `UPnP/1.1` *`product/version`*

In summary, in support of the efficient discovery of hosts in large software-defined networking (SDN)- or fabric-based network environments, various solutions have been provided herein through several techniques. A first technique supports, among other things, efficiently managing silent ports and silent MAC addresses and encompasses, for example, applying a novel heuristic to ports and MAC addresses, classifying such entities (as, for example, silent, quiet, and noisy), intelligently polling such entities, etc. A second technique supports, among other things, an MLD-based host discovery that is applicable to IPv4 and encompasses, for example, a host creating an IPv6 address that embeds its IPv4 address, the addition of a well-known first byte to the three bytes in a SNMA, the use of a form of unicast ping to confirm whether a host formed a derived address, etc. A third technique supports, among other things, the use of a service lookup for deterministic host discovery and encompasses, for example, the use of upper-layer discovery services to cause a host to expose its addresses in the replies to multicast discoveries.