            Address Resolution Problems in Large Data Center Networks

Abstract

   This document examines address resolution issues related to the
   scaling of data centers with a very large number of hosts. The scope
   of this document is relatively narrow, focusing on address resolution
   (the Address Resolution Protocol (ARP) in IPv4 and Neighbor Discovery
   (ND) in IPv6) within a data center.

Copyright Notice

Table of Contents

1. Introduction

   This document examines issues related to the scaling of large data
   centers. Specifically, this document focuses on address resolution
   (ARP in IPv4 and Neighbor Discovery in IPv6) within the data center.
   Although strictly speaking the scope of address resolution is
   confined to a single L2 broadcast domain (i.e., ARP runs at the L2
   layer below IP), the issue is complicated by routers having many
   interfaces on which address resolution must be performed or with the
   presence of IEEE 802.1Q domains, where individual VLANs effectively
   form their own L2 broadcast domains. Thus, the scope of address
   resolution spans both the L2 link and the devices attached to those
   links.

   This document identifies potential issues associated with address
   resolution in data centers with a large number of hosts. The scope
   of this document is intentionally relatively narrow, as it mirrors
   the Address Resolution for Massive numbers of hosts in the Data
   center (ARMD) WG charter. This document lists "pain points" that are
   being experienced in current data centers. The goal of this document
   is to focus on address resolution issues and not other broader issues
   that might arise in data centers.

2. Terminology

   Address Resolution: The process of determining the link-layer
      address corresponding to a given IP address. In IPv4, address
      resolution is performed by ARP [RFC0826]; in IPv6, it is provided
      by Neighbor Discovery (ND) [RFC4861].

   Application: Software that runs on either a physical or virtual
      machine, providing a service (e.g., web server, database server,
      etc.).

   L2 Broadcast Domain: The set of all links, repeaters, and switches
      that are traversed to reach all nodes that are members of a given
      L2 broadcast domain. In IEEE 802.1Q networks, a broadcast domain
      corresponds to a single VLAN.

   Host (or server): A computer system on the network.

   Hypervisor: Software running on a host that allows multiple VMs to
      run on the same host.

   Virtual Machine (VM): A software implementation of a physical
      machine that runs programs as if they were executing on a
      physical, non-virtualized machine. Applications (generally) do
      not know they are running on a VM as opposed to running on a

"bare" host or server, though some systems provide a
paravirtualization environment that allows an operating system or
application to be aware of the presence of virtualization for
optimization purposes.

ToR: Top-of-Rack Switch. A switch placed in a single rack to
aggregate network connectivity to and from hosts in that rack.

EoR: End-of-Row Switch. A switch used to aggregate network
connectivity from multiple racks. EoR switches are the next level
of switching above ToR switches.

3. Background

Large, flat L2 networks have long been known to have scaling
problems. As the size of an L2 broadcast domain increases, the level
of broadcast traffic from protocols like ARP increases. Large
amounts of broadcast traffic pose a particular burden because every
device (switch, host, and router) must process and possibly act on
such traffic. In extreme cases, "broadcast storms" can occur where
the quantity of broadcast traffic reaches a level that effectively
brings down part or all of a network. For example, poor
implementations of loop detection and prevention or misconfiguration
errors can create conditions that lead to broadcast storms as network
conditions change. The conventional wisdom for addressing such
problems has been to say "don't do that". That is, split large L2
networks into multiple smaller L2 networks, each operating as its own
L3/IP subnet. Numerous data center networks have been designed with
this principle, e.g., with each rack placed within its own L3 IP
subnet. By doing so, the broadcast domain (and address resolution)
is confined to one ToR switch, which works well from a scaling
perspective. Unfortunately, this conflicts in some ways with the
current trend towards dynamic workload shifting in data centers and
increased virtualization, as discussed below.

Workload placement has become a challenging task within data centers.
Ideally, it is desirable to be able to dynamically reassign workloads
within a data center in order to optimize server utilization, add
more servers in response to increased demand, etc. However, servers
are often pre-configured to run with a given set of IP addresses.
Placement of such servers is then subject to constraints of the IP
addressing restrictions of the data center. For example, servers
configured with addresses from a particular subnet could only be
placed where they connect to the IP subnet corresponding to their IP
addresses. If each ToR switch is acting as a gateway for its own
subnet, a server can only be connected to the one ToR switch. This
gateway switch represents the L2/L3 boundary. A similar constraint
occurs in virtualized environments, as discussed next.

Server virtualization is fast becoming the norm in data centers.
With server virtualization, each physical server supports multiple
virtual machines, each running its own operating system, middleware,
and applications. Virtualization is a key enabler of workload
agility, i.e., allowing any server to host any application (on its
own VM) and providing the flexibility of adding, shrinking, or moving
VMs within the physical infrastructure. Server virtualization
provides numerous benefits, including higher utilization, increased
data security, reduced user downtime, and even significant power
conservation, along with the promise of a more flexible and dynamic
computing environment.

The discussion below focuses on VM placement and migration. Keep in
mind, however, that even in a non-virtualized environment, many of
the same issues apply to individual workloads running on standalone
machines. For example, when increasing the number of servers running
a particular workload to meet demand, placement of those workloads
may be constrained by IP subnet numbering considerations, as
discussed earlier.

The greatest flexibility in VM and workload management occurs when it
is possible to place a VM (or workload) anywhere in the data center
regardless of what IP addresses the VM uses and how the physical
network is laid out. In practice, movement of VMs within a data
center is easiest when VM placement and movement do not conflict with
the IP subnet boundaries of the data center's network, so that the
VM's IP address need not be changed to reflect its actual point of
attachment on the network from an L3/IP perspective. In contrast, if
a VM moves to a new IP subnet, its address must change, and clients
will need to be made aware of that change. From a VM management
perspective, management is simplified if all servers are on a single
large L2 network.

With virtualization, it is not uncommon to have a single physical
server host ten or more VMs, each having its own IP (and Media Access
Control (MAC)) addresses. Consequently, the number of addresses per
machine (and hence per subnet) is increasing, even when the number of
physical machines stays constant. In a few years, the numbers will
likely be even higher.

In the past, applications were static in the sense that they tended
to stay in one physical place. An application installed on a
physical machine would stay on that machine because the cost of
moving an application elsewhere was generally high. Moreover,
physical servers hosting applications would tend to be placed in such
a way as to facilitate communication locality. That is, applications
running on servers would be physically located near the servers
hosting the applications they communicated with most heavily. The

network traffic patterns in such environments could thus be
optimized, in some cases keeping significant traffic local to one
network segment. In these more static and carefully managed
environments, it was possible to build networks that approached
scaling limitations but did not actually cross the threshold.

Today, with the proliferation of VMs, traffic patterns are becoming
more diverse and less predictable. In particular, there can easily
be less locality of network traffic as VMs hosting applications are
moved for such reasons as reducing overall power usage (by
consolidating VMs and powering off idle machines) or moving a VM to a
physical server with more capacity or a lower load. In today's
changing environments, it is becoming more difficult to engineer
networks as traffic patterns continually shift as VMs move around.

In summary, both the size and density of L2 networks are increasing.
In addition, increasingly dynamic workloads and the increased usage
of VMs are creating pressure for ever-larger L2 networks. Today,
there are already data centers with over 100,000 physical machines
and many times that number of VMs. This number will only increase
going forward. In addition, traffic patterns within a data center
are also constantly changing. Ultimately, the issues described in
this document might be observed at any scale, depending on the
particular design of the data center.

4. Address Resolution in IPv4

In IPv4 over Ethernet, ARP provides the function of address
resolution. To determine the link-layer address of a given IP
address, a node broadcasts an ARP Request. The request is delivered
to all portions of the L2 network, and the node with the requested IP
address responds with an ARP Reply. ARP is an old protocol and, by
current standards, is sparsely documented. For example, there are no
clear requirements for retransmitting ARP Requests in the absence of
replies. Consequently, implementations vary in the details of what
they actually implement [RFC0826][RFC1122].

From a scaling perspective, there are a number of problems with ARP.
First, it uses broadcast, and any network with a large number of
attached hosts will see a correspondingly large amount of broadcast
ARP traffic. The second problem is that it is not feasible to change
host implementations of ARP -- current implementations are too widely
entrenched, and any changes to host implementations of ARP would take
years to become sufficiently deployed to matter. That said, it may
be possible to change ARP implementations in hypervisors, L2/L3
boundary routers, and/or ToR access switches, to leverage such
techniques as Proxy ARP. Finally, ARP implementations need to take
steps to flush out stale or otherwise invalid entries.

Unfortunately, existing standards do not provide clear implementation
guidelines for how to do this. Consequently, implementations vary
significantly, and some implementations are "chatty" in that they
just periodically flush caches every few minutes and send new ARP
queries.

5. Address Resolution in IPv6

Broadly speaking, from the perspective of address resolution, IPv6's
Neighbor Discovery (ND) behaves much like ARP, with a few notable
differences. First, ARP uses broadcast, whereas ND uses multicast.
When querying for a target IP address, ND maps the target address
into an IPv6 Solicited Node multicast address. Using multicast
rather than broadcast has the benefit that the multicast frames do
not necessarily need to be sent to all parts of the network, i.e.,
the frames can be sent only to segments where listeners for the
Solicited Node multicast address reside. In the case where multicast
frames are delivered to all parts of the network, sending to a
multicast address still has the advantage that most (if not all)
nodes will filter out the (unwanted) multicast query via filters
installed in the Network Interface Card (NIC) rather than burdening
host software with the need to process such packets. Thus, whereas
all nodes must process every ARP query, ND queries are processed only
by the nodes to which they are intended. In cases where multicast
filtering can't effectively be implemented in the NIC (e.g., as on
hypervisors supporting virtualization), filtering would need to be
done in software (e.g., in the hypervisor's vSwitch).

6. Generalized Data Center Design

There are many different ways in which data center networks might be
designed. The designs are usually engineered to suit the particular
workloads that are being deployed in the data center. For example, a
large web server farm might be engineered in a very different way
than a general-purpose multi-tenant cloud hosting service. However,
in most cases the designs can be abstracted into a typical three-
layer model consisting of an access layer, an aggregation layer, and
the Core. The access layer generally refers to the switches that are
closest to the physical or virtual servers; the aggregation layer
serves to interconnect multiple access-layer devices. The Core
switches connect the aggregation switches to the larger network core.

Figure 1 shows a generalized data center design, which captures the
essential elements of various alternatives.

```
      +-----+-----+     +-----+-----+
      |   Core0   |     |   Core1   |        Core
      +-----+-----+     +-----+-----+
           /   \         /     /
          /      \---------\   /
         /    /---------/    \ /
      +-------+          +------+
    +/------+ |        +/-----+ |
    | Aggr11| + --------|AggrN1| +        Aggregation Layer
    +---+---+/          +------+/
       /   \             /    \
      /     \           /      \
   +---+   +---+     +---+    +---+
   |T11|...|T1x|     |TN1|    |TNy|     Access Layer
   +---+   +---+     +---+    +---+
   | |     | |       | |      | |
   +---+   +---+     +---+    +---+
   | |...  | |       | |      | |
   +---+   +---+     +---+    +---+     Server Racks
   | |...  | |       | |      | |
   +---+   +---+     +---+    +---+
   | |...  | |       | |      | |
   +---+   +---+     +---+    +---+
```
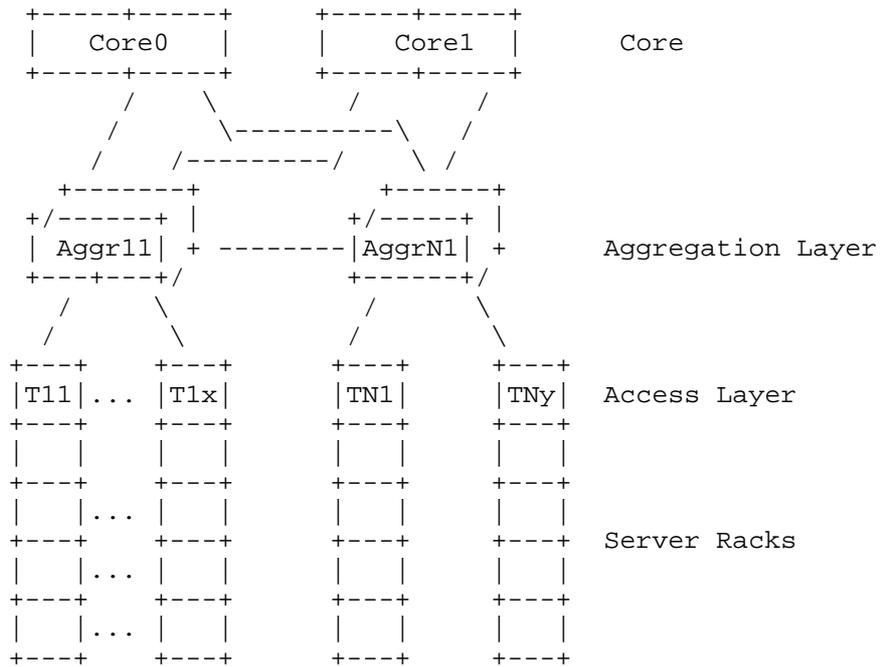
            Typical Layered Architecture in a Data Center

                               Figure 1

6.1. Access Layer

   The access switches provide connectivity directly to/from physical
   and virtual servers. The access layer may be implemented by wiring
   the servers within a rack to a ToR switch or, less commonly, the
   servers could be wired directly to an EoR switch. A server rack may
   have a single uplink to one access switch or may have dual uplinks to
   two different access switches.

6.2. Aggregation Layer

   In a typical data center, aggregation switches interconnect many ToR
   switches. Usually, there are multiple parallel aggregation switches,
   serving the same group of ToRs to achieve load sharing. It is no
   longer uncommon to see aggregation switches interconnecting hundreds
   of ToR switches in large data centers.

6.3. Core

   Core switches provide connectivity between aggregation switches and
   the main data center network. Core switches interconnect different
   sets of racks and provide connectivity to data center gateways
   leading to external networks.

6.4. L3/L2 Topological Variations

6.4.1. L3 to Access Switches

   In this scenario, the L3 domain is extended all the way from the core
   network to the access switches. Each rack enclosure consists of a
   single L2 domain, which is confined to the rack. In general, there
   are no significant ARP/ND scaling issues in this scenario, as the L2
   domain cannot grow very large. Such a topology has benefits in
   scenarios where servers attached to a particular access switch
   generally run VMs that are confined to using a single subnet. These
   VMs and the applications they host aren't moved (migrated) to other
   racks that might be attached to different access switches (and
   different IP subnets). A small server farm or very static compute
   cluster might be well served via this design.

6.4.2. L3 to Aggregation Switches

   When the L3 domain extends only to aggregation switches, hosts in any
   of the IP subnets configured on the aggregation switches can be
   reachable via L2 through any access switches if access switches
   enable all the VLANs. Such a topology allows a greater level of
   flexibility, as servers attached to any access switch can run any VMs
   that have been provisioned with IP addresses configured on the
   aggregation switches. In such an environment, VMs can migrate
   between racks without IP address changes. The drawback of this
   design, however, is that multiple VLANs have to be enabled on all
   access switches and all access-facing ports on aggregation switches.
   Even though L2 traffic is still partitioned by VLANs, the fact that
   all VLANs are enabled on all ports can lead to broadcast traffic on
   all VLANs that traverse all links and ports, which has the same
   effect as one big L2 domain on the access-facing side of the
   aggregation switch. In addition, the internal traffic itself might
   have to cross different L2 boundaries, resulting in significant
   ARP/ND load at the aggregation switches. This design provides a good
   tradeoff between flexibility and L2 domain size. A moderate-sized
   data center might utilize this approach to provide high-availability
   services at a single location.

6.4.3. L3 in the Core Only

   In some cases, where a wider range of VM mobility is desired (i.e., a
   greater number of racks among which VMs can move without IP address
   changes), the L3 routed domain might be terminated at the core
   routers themselves. In this case, VLANs can span multiple groups of
   aggregation switches, which allows hosts to be moved among a greater
   number of server racks without IP address changes. This scenario
   results in the largest ARP/ND performance impact, as explained later.
   A data center with very rapid workload shifting may consider this
   kind of design.

6.4.4. Overlays

   There are several approaches where overlay networks can be used to
   build very large L2 networks to enable VM mobility. Overlay networks
   using various L2 or L3 mechanisms allow interior switches/routers to
   mask host addresses. In addition, L3 overlays can help the data
   center designer control the size of the L2 domain and also enhance
   the ability to provide multi-tenancy in data center networks.
   However, the use of overlays does not eliminate traffic associated
   with address resolution; it simply moves it to regular data traffic.
   That is, address resolution is implemented in the overlay and is not
   directly visible to the switches of the data center network.

   A potential problem that arises in a large data center is that when a
   large number of hosts communicate with their peers in different
   subnets, all these hosts send (and receive) data packets to their
   respective L2/L3 boundary nodes, as the traffic flows are generally
   bidirectional. This has the potential to further highlight any
   scaling problems. These L2/L3 boundary nodes have to process ARP/ND
   requests sent from originating subnets and resolve physical (MAC)
   addresses in the target subnets for what are generally bidirectional
   flows. Therefore, for maximum flexibility in managing the data
   center workload, it is often desirable to use overlays to place
   related groups of hosts in the same topological subnet to avoid the
   L2/L3 boundary translation. The use of overlays in the data center
   network can be a useful design mechanism to help manage a potential
   bottleneck at the L2/L3 boundary by redefining where that boundary
   exists.

6.5. Factors That Affect Data Center Design

6.5.1. Traffic Patterns

   Expected traffic patterns play an important role in designing
   appropriately sized access, aggregation, and core networks. Traffic
   patterns also vary based on the expected use of the data center.

   Broadly speaking, it is desirable to keep as much traffic as possible
   on the access layer in order to minimize the bandwidth usage at the
   aggregation layer. If the expected use of the data center is to
   serve as a large web server farm, where thousands of nodes are doing
   similar things and the traffic pattern is largely in and out of a
   large data center, an access layer with EoR switches might be used,
   as it minimizes complexity, allows for servers and databases to be
   located in the same L2 domain, and provides for maximum density.

   A data center that is expected to host a multi-tenant cloud hosting
   service might have some completely unique requirements. In order to
   isolate inter-customer traffic, smaller L2 domains might be
   preferred, and though the size of the overall data center might be
   comparable to the previous example, the multi-tenant nature of the
   cloud hosting application requires a smaller and more
   compartmentalized access layer. A multi-tenant environment might
   also require the use of L3 all the way to the access-layer ToR
   switch.

   Yet another example of a workload with a unique traffic pattern is a
   high-performance compute cluster, where most of the traffic is
   expected to stay within the cluster but at the same time there is a
   high degree of crosstalk between the nodes. This would once again
   call for a large access layer in order to minimize the requirements
   at the aggregation layer.

6.5.2. Virtualization

   Using virtualization in the data center further serves to increase
   the possible densities that can be achieved. However, virtualization
   also further complicates the requirements on the access layer, as
   virtualization restricts the scope of server placement in the event
   of server failover resulting from hardware failures or server
   migration for load balancing or other reasons.

   Virtualization also can place additional requirements on the
   aggregation switches in terms of address resolution table size and
   the scalability of any address-learning protocols that might be used
   on those switches. The use of virtualization often also requires the
   use of additional VLANs for high-availability beaconing, which would

need to span the entire virtualized infrastructure. This would
require the access layer to also span the entire virtualized
infrastructure.

6.5.3. Summary

The designs described in this section have a number of tradeoffs.
The "L3 to access switches" design described in Section 6.4.1 is the
only design that constrains L2 domain size in a fashion that avoids
ARP/ND scaling problems. However, that design has limitations and
does not address some of the other requirements that lead to
configurations that make use of larger L2 domains. Consequently,
ARP/ND scaling issues are a real problem in practice.

7. Problem Itemization

This section articulates some specific problems or "pain points" that
are related to large data centers.

7.1. ARP Processing on Routers

One pain point with large L2 broadcast domains is that the routers
connected to the L2 domain may need to process a significant amount
of ARP traffic in some cases. In particular, environments where the
aggregate level of ARP traffic is very large may lead to a heavy ARP
load on routers. Even though the vast majority of ARP traffic may
not be aimed at that router, the router still has to process enough
of the ARP Request to determine whether it can safely be ignored.
The ARP algorithm specifies that a recipient must update its ARP
cache if it receives an ARP query from a source for which it has an
entry [RFC0826].

ARP processing in routers is commonly handled in a "slow path"
software processor, rather than directly by a hardware Application-
Specific Integrated Circuit (ASIC) as is the case when forwarding
packets. Such a design significantly limits the rate at which ARP
traffic can be processed compared to the rate at which ASICs can
forward traffic. Current implementations at the time of this writing
can support ARP processing in the low thousands of ARP packets per
second. In some deployments, limitations on the rate of ARP
processing have been cited as being a problem.

To further reduce the ARP load, some routers have implemented
additional optimizations in their forwarding ASIC paths. For
example, some routers can be configured to discard ARP Requests for
target addresses other than those assigned to the router. That way,
the router's software processor only receives ARP Requests for

addresses it owns and must respond to. This can significantly reduce
the number of ARP Requests that must be processed by the router.

Another optimization concerns reducing the number of ARP queries
targeted at routers, whether for address resolution or to validate
existing cache entries. Some routers can be configured to broadcast
periodic gratuitous ARPs [RFC5227]. Upon receipt of a gratuitous
ARP, implementations mark the associated entry as "fresh", resetting
the aging timer to its maximum setting. Consequently, sending out
periodic gratuitous ARPs can effectively prevent nodes from needing
to send ARP Requests intended to revalidate stale entries for a
router. The net result is an overall reduction in the number of ARP
queries routers receive. Gratuitous ARPs, broadcast to all nodes in
the L2 broadcast domain, may in some cases also pre-populate ARP
caches on neighboring devices, further reducing ARP traffic. But it
is not believed that pre-population of ARP entries is supported by
most implementations, as the ARP specification [RFC0826] recommends
only that pre-existing ARP entries be updated upon receipt of ARP
messages; it does not call for the creation of new entries when none
already exist.

Finally, another area concerns the overhead of processing IP packets
for which no ARP entry exists. Existing standards specify that one
or more IP packets for which no ARP entries exist should be queued
pending successful completion of the address resolution process
[RFC1122] [RFC1812]. Once an ARP query has been resolved, any queued
packets can be forwarded on. Again, the processing of such packets
is handled in the "slow path", effectively limiting the rate at which
a router can process ARP "cache misses", and is viewed as a problem
in some deployments today. Additionally, if no response is received,
the router may send the ARP/ND query multiple times. If no response
is received after a number of ARP/ND requests, the router needs to
drop any queued data packets and may send an ICMP destination
unreachable message as well [RFC0792]. This entire process can be
CPU intensive.

Although address resolution traffic remains local to one L2 network,
some data center designs terminate L2 domains at individual
aggregation switches/routers (e.g., see Section 6.4.2). Such routers
can be connected to a large number of interfaces (e.g., 100 or more).
While the address resolution traffic on any one interface may be
manageable, the aggregate address resolution traffic across all
interfaces can become problematic.

Another variant of the above issue has individual routers servicing a
relatively small number of interfaces, with the individual interfaces
themselves serving very large subnets. Once again, it is the
aggregate quantity of ARP traffic seen across all of the router's

interfaces that can be problematic. This pain point is essentially
the same as the one discussed above, the only difference being
whether a given number of hosts are spread across a few large IP
subnets or many smaller ones.

When hosts in two different subnets under the same L2/L3 boundary
router need to communicate with each other, the L2/L3 router not only
has to initiate ARP/ND requests to the target's subnet, it also has
to process the ARP/ND requests from the originating subnet. This
process further adds to the overall ARP processing load.

7.2. IPv6 Neighbor Discovery

Though IPv6's Neighbor Discovery behaves much like ARP, there are
several notable differences that result in a different set of
potential issues. From an L2 perspective, an important difference is
that ND address resolution requests are sent via multicast, which
results in ND queries only being processed by the nodes for which
they are intended. Compared with broadcast ARPs, this reduces the
total number of ND packets that an implementation will receive.

Another key difference concerns revalidating stale ND entries. ND
requires that nodes periodically revalidate any entries they are
using, to ensure that bad entries are timed out quickly enough that
TCP does not terminate a connection. Consequently, some
implementations will send out "probe" ND queries to validate in-use
ND entries as frequently as every 35 seconds [RFC4861]. Such probes
are sent via unicast (unlike in the case of ARP). However, on larger
networks, such probes can result in routers receiving many such
queries (i.e., many more than with ARP, which does not specify such
behavior). Unfortunately, the IPv4 mitigation technique of sending
gratuitous ARPs (as described in Section 7.1) does not work in IPv6.
The ND specification specifically states that gratuitous ND "updates"
cannot cause an ND entry to be marked "valid". Rather, such entries
are marked "probe", which causes the receiving node to (eventually)
generate a probe back to the sender, which in this case is precisely
the behavior that the router is trying to prevent!

Routers implementing Neighbor Unreachability Discovery (NUD) (for
neighboring destinations) will need to process neighbor cache state
changes such as transitioning entries from REACHABLE to STALE. How
this capability is implemented may impact the scalability of ND on a
router. For example, one possible implementation is to have the
forwarding operation detect when an ND entry is referenced that needs
to transition from REACHABLE to STALE, by signaling an event that
would need to be processed by the software processor. Such an
implementation could increase the load on the service processor in

   much the same way that high rates of ARP requests have led to
   problems on some routers.

   It should be noted that ND does not require the sending of probes in
   all cases. Section 7.3.1 of [RFC4861] describes a technique whereby
   hints from TCP can be used to verify that an existing ND entry is
   working fine and does not need to be revalidated.

   Finally, IPv6 and IPv4 are often run simultaneously and in parallel
   on the same network, i.e., in dual-stack mode. In such environments,
   the IPv4 and IPv6 issues enumerated above compound each other.

7.3. MAC Address Table Size Limitations in Switches

   L2 switches maintain L2 MAC address forwarding tables for all sources
   and destinations traversing the switch. These tables are populated
   through learning and are used to forward L2 frames to their correct
   destination. The larger the L2 domain, the larger the tables have to
   be. While in theory a switch only needs to keep track of addresses
   it is actively using (sometimes called "conversational learning"),
   switches flood broadcast frames (e.g., from ARP), multicast frames
   (e.g., from Neighbor Discovery), and unicast frames to unknown
   destinations. Switches add entries for the source addresses of such
   flooded frames to their forwarding tables. Consequently, MAC address
   table size can become a problem as the size of the L2 domain
   increases. The table size problem is made worse with VMs, where a
   single physical machine now hosts many VMs (in the 10's today, but
   growing rapidly as the number of cores per CPU increases), since each
   VM has its own MAC address that is visible to switches.

   When L3 extends all the way to access switches (see Section 6.4.1),
   the size of MAC address tables in switches is not generally a
   problem. When L3 extends only to aggregation switches (see
   Section 6.4.2), however, MAC table size limitations can be a real
   issue.

8. Summary

   This document has outlined a number of issues related to address
   resolution in large data centers. In particular, this document has
   described different scenarios where such issues might arise and what
   these potential issues are, along with outlining fundamental factors
   that cause them. It is hoped that describing specific pain points
   will facilitate a discussion as to whether they should be addressed
   and how best to address them.

9. Acknowledgments

   This document has been significantly improved by comments from Manav
   Bhatia, David Black, Stewart Bryant, Ralph Droms, Linda Dunbar,
   Donald Eastlake, Wesley Eddy, Anoop Ghanwani, Joel Halpern, Sue
   Hares, Pete Resnick, Benson Schliesser, T. Sridhar, and Lucy Yong.
   Igor Gashinsky deserves additional credit for highlighting some of
   the ARP-related pain points and for clarifying the difference between
   what the standards require and what some router vendors have actually
   implemented in response to operator requests.

10. Security Considerations

   This document does not create any security implications nor does it
   have any security implications. The security vulnerabilities in ARP
   are well known, and this document does not change or mitigate them in
   any way. Security considerations for Neighbor Discovery are
   discussed in [RFC4861] and [RFC6583].

11. Informative References

   [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5,
              RFC 792, September 1981.

   [RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or
              converting network protocol addresses to 48.bit Ethernet
              address for transmission on Ethernet hardware", STD 37,
              RFC 826, November 1982.

   [RFC1122] Braden, R., "Requirements for Internet Hosts -
              Communication Layers", STD 3, RFC 1122, October 1989.

   [RFC1812] Baker, F., "Requirements for IP Version 4 Routers",
              RFC 1812, June 1995.

   [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
              "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
              September 2007.

   [RFC5227] Cheshire, S., "IPv4 Address Conflict Detection", RFC 5227,
              July 2008.

   [RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational
              Neighbor Discovery Problems", RFC 6583, March 2012.

Authors' Addresses

    Thomas Narten
    IBM Corporation
    3039 Cornwallis Ave.
    PO Box 12195
    Research Triangle Park, NC   27709-2195
    USA

    EMail: narten@us.ibm.com

    Manish Karir
    Merit Network Inc.

    EMail: mkarir@merit.edu

    Ian Foo
    Huawei Technologies

    EMail: Ian.Foo@huawei.com