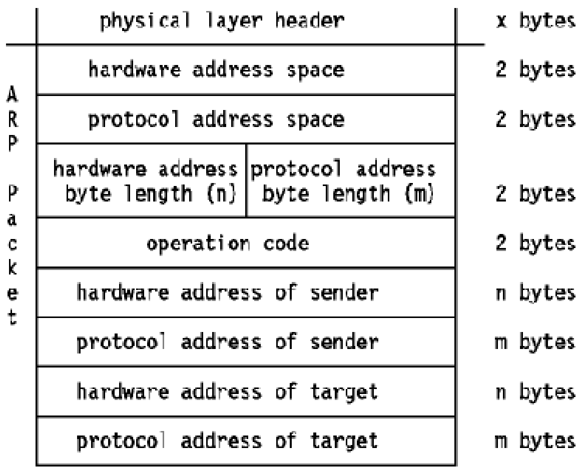
Does a Dynamic Infrastructure Need ARP for Applications?

Lori MacVittie, 2009-18-09

There's more than one way to address the rapid rate of change in infrastructure supporting a dynamic environment.

We spend a lot of time talking about how software and systems and standards are the ultimate solution to addressing the rapid rate of change in the association between applications and IP addresses in a dynamic infrastructure. But sometimes you have look *down* the stack to find a simpler, more economical and honestly, elegant, answer to the challenge of managing the problem associated with virtualized and cloud computing architectures. We need to take another look at the *link layer* protocols and specifically ARP (address resolution protocol) for some pointers on how we might address this particular challenge.


physical layer header
x bytes
ARP Packet
hardware address space
2 bytes
protocol address space
2 bytes
hardware address byte length (n)
protocol address byte length (m)
2 bytes
operation code
2 bytes
hardware address of sender
n bytes
protocol address of sender
m bytes
hardware address of target
n bytes
protocol address of target
m bytes

THE PROBLEM

In a nutshell the basic problem is that in a scalable, on-demand environment like cloud computing applications are launched and shutdown at a higher rate than is seen in traditional architectures. Because they may be (should be) launched on whatever host has the resources to meet its requirements, the associated IP addresses are highly likely to vary during any given time interval. This can cause problems for infrastructure solutions that rely on IP address to apply their peculiar functionality to the application.

That means just about everything, because all environments today are essentially IP-based.

Example: load balancing devices such as application delivery controllers manage applications via a pool of *nodes*, each of which is assumed to be capable of responding to application requests. These nodes are typically identified by IP address. Some are capable of using FQDN (Fully Qualified Domain Name) to identify nodes, but this has a price in performance as it requires a DNS lookup for the IP address *for every request*. Thus it is rare to see this usage in live environments. Changes in the composition of the overall pool of nodes requires either manual or programmatic (scripted or integrated in a third-party application) intervention. While a programmatic approach is more efficient from a process and time perspective, it still requires integration which requires API and product-specific skill sets that are not always available in any given organization. This is mostly problematic because there exist no standards for integration; every infrastructure solution with an API is different, requiring integrators to leverage multiple APIs to programmatically build automated and orchestrated systems.

A SIMPLE(R) SOLUTION?

The mappings between applications and IP addresses are tightly coupled in a way that is very similar to MAC address coupling with IP addresses. While they are in service they are tightly coupled. But change in **MAC –> IP** address mappings are not uncommon, especially in a virtualized environment. ARP (Address Resolution Protocol) resolves many of the challenges associated with mapping MAC –> IP in the network communication layers and even provides multiple methods of discovering such mappings.

What we need, then, is a way to dynamically map and discover the relationship between **APPLICATION –> IP ADDRESS** using a method similar to ARP. For purposes of this discussion, let's call it ARP7 (for layer 7, of course).

From Wikipedia:

> *In computer networking, the Address Resolution Protocol (ARP) is the method for finding a host's link layer (hardware) address when only its Internet Layer (IP) or some other Network Layer address is known.*

Now let's replace that with some application-specific language:

> *In computer networking, the Application Resolution Protocol Layer 7 (ARP7) is the method for finding an application's IP address (IP) address when only its Application Name (AN) is known.*

This is not DNS for applications or DNS with special extensions. This is intended to be an autonomic process in much the same was as ARP is autonomic in that when an application is first launched and can identify its IP address it automatically sends a broadcast message on the local network identifying itself. Conversely, it can answer broadcast queries such as "Who has application X" in much the same way a single host answers "Who has IP address X". Perhaps all that's really necessary is to implement Gratuitous ARP7 for applications, in which a broadcast message is sent on behalf of the host by another device (the hypervisor? the virtual machine?).

> *Many operating systems perform this during startup. It helps to resolve problems which would otherwise occur if, for example, a network card was recently changed (changing the IP-address-to-MAC-address mapping) and other hosts still have the old mapping in their ARP caches.*

That sounds eerily familiar to the situation in which a load balancing device maintains a list of nodes (IP addresses) on which applications reside. In an on-demand environment, those IP addresses may change at a rapid rate, making the process of manually updating them inefficient. While Infrastructure 2.0 enabled load balancing devices are capable of integration with management systems driving the provisioning and decommissioning of applications and therefore can be automatically updated, this requires integration work on the part of the provider or management system provider. Wouldn't it be nice, then, if the load balancing device – or any other infrastructure component that maps IP addresses to applications for any reason – could simply passively monitor the local network for "ARP7" updates that would indicate a change in IP address?

Example: Application delivery controllers – a load balancing device – that already maintain the ability to passively monitor application exchanges determine if the application is no longer available (for whatever reason) and can automatically remove it from the pool of available nodes. While it can also re-enable the node if it returns to service, it can't – on its own – *discover* new nodes on which the application it is managing might suddenly appear. This is currently the role of automation and orchestration and provisioning systems.

A network-layer protocol, however, that carried a unique application id in much the same way as MAC addresses are unique and was broadcast such that every component on the network could "hear" the announcement and react to it, would be much more efficient. If the device sees an "ARP7" broadcast it can grab it, examine it, and determine if the application being announced is relevant to its configuration. Because it's a broadcast message *every* device can simultaneously receive the announcement and perform whatever updates to its configuration is necessary.

PROTOCOLS ARE IMPLEMENTATION NEUTRAL

This kind of solution is applicable to solutions other than those provided as examples such as firewalls, bandwidth management, content filtering, etc… Basically any infrastructure solution that maps applications to IP addresses could take advantage of the existence of such a protocol. Management and monitoring applications, too, could benefit from the existence of such a protocol. Using a broadcast-style protocol instead of port-mirroring solutions to duplicate data required by multiple systems (as opposed to deploying inline and potentially degrading performance or introducing a single point of failure) may increase network utilization slightly, but with internal networks running at multi-gigabit speeds these days, it shouldn't be that much of an issue and it eliminates the need to mirror ***all*** traffic for some solutions.

Implementing a solution to this challenge as a protocol means it is highly non-intrusive as devices that do not support the functionality can simply ignore the packets on the wire. Yet it is a highly disruptive solution to the problem as it would radically open up a wide variety of functionality and features across multiple markets and product sets and offer an answer in general to the decades old problem of automated application discovery. It would change how we manage application-focused infrastructure in ***any*** environment.

It may be the case that because of timelines, ARP is not the right model. NDP (Neighbor Discovery Protocol) is designed for IPv6, so perhaps we should look to ***it*** to implement an ADP (Application Discovery Protocol) that still maintains the basic concepts but is updated for modern IP layer connectivity.

Not only is this an interoperable solution – the protocol would be product/vendor agnostic – but it's an elegant, real-time solution that is vastly more simple than the large integration-based solutions that have been proposed so far (even by me). Note that this is ***not*** an IPAM solution. In fact, this assumes that IPAM is already taken care of by someone/something else. This solution is specifically aimed at solving the problem of how to most efficiently share changes in IP addresses across applications being managed and delivered and secured by a variety of infrastructure solutions.

Crazy? Perhaps. But ignoring that the network layer has already solved some of the same problems we're facing now at the application layer would be folly. After all, if we're going to need to create new standards to support a dynamic infrastructure (and we are) why shouldn't one of those be a new network protocol?

- Cloud Computing Makes Servers Obsolete
- The Cloud Metastructure Hubub
- Dynamic Infrastructure: The Cloud within the Cloud
- The days of IP-based management are numbered
- Managing Virtual Infrastructure Requires an Application Centric Approach
- Server Virtualization versus Server Virtualization
- The Context-Aware Cloud

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc. Corporate Headquarters info@f5.com	F5 Networks Asia-Pacific apacinfo@f5.com	F5 Networks Ltd. Europe/Middle-East/Africa emeainfo@f5.com	F5 Networks Japan K.K. f5j-info@f5.com