Doing More with Less: End-to-End Consistent IPv4 Address Sharing

Manish Karir, Eric Wustrow, Jim Rees
Networking Research and Development
Merit Network Inc.
Ann Arbor, MI 48104 USA
{mkarir, ewust, rees}@merit.edu

***Abstract*—The Internet is rapidly nearing IPv4 address space exhaustion. Current projections predict that within the next two years, all IPv4 address blocks will have been assigned. IPv6 adoption on the other hand has been slower than anticipated. It is becoming increasingly clear that there will be an extended period of during which both protocols will coexist as services and applications are slowly migrated to IPv6. As we transition from an Internet built on an abundance of IPv4 addresses to one of scarcity, innovative techniques that allow us to do more with less will become increasingly important. One such class of emerging techniques attempts to utilize unused port ranges to implement IP address sharing. We call this class of approaches *Port Scavenging*. In this paper we present a unique approach that allows multiple end hosts to share a single IPv4 address by relying on a modified device address resolution protocol. Our approach is fundamentally different from other techniques, as it does not require that packets from the end-host be modified at the network layer by an intermediate entity as they transit the network. Each end-host can use a valid routable public IP address. We have implemented our ideas in a modified Linux kernel to demonstrate the feasibility of our approach. Though not suitable for all environments, this technique can be a valuable addition to the IPv6 transition toolchest.**

I. Introduction

Current projections indicate that within the next two years, all available IPv4 address blocks will have been assigned [1]. The resulting shortage of IP addresses would severely limit innovation on the Internet and even its continued spread throughout the world. While the use of IPv6 would expand the available number of Internet addresses, adoption has been slow, as it requires not only new hardware and software to be pervasively deployed, but also requires network operators to transition entire networks and user bases to this new set of networking protocols. This has greatly slowed adoption.

One way to stretch the supply of IPv4 addresses is to use *Port Scavenging* techniques, which leverage unused port numbers to implement address sharing among a group of hosts. Over the past 10 years, one such technique, Network Address Translation (NAT), has emerged as the primary approach for sharing existing IPv4 addresses [2]. This approach relies on devices that modify IP addresses and port numbers in IP packets as they are transmitted between the network end-host and the Internet. While to an outside observer, the hosts appear to have the same IP address, they in fact have distinct private IP addresses internally. Such transparent modifications of packets in transit between source and destination cause multiple problems at all layers from network on up to application [3].

One of the primary difficulties of the NAT approach is that a host behind a NAT does not know its own Internet address, so it is unable to advertise services to the rest of the world without some help from the NAT device, which must involve manual configuration or an add-on protocol such as UPNP [4]. NAT also breaks any protocol that depends on embedded addresses or port numbers, for example FTP, peer-to-peer, H.323, SNMP, or file system protocols that depend on callbacks [5]. Some of these protocols can be patched up with an *Application-Level Gateway* but others can not. For example there is no way to pass an IPSec Authentication Header through a NAT without making arrangements ahead of time to allow for the NAT traversal [6]. This presents a fundamental problem for any security protocol that attempts to guarantee the integrity of the IP header, since the header must be modified by a device (the NAT) that is outside the security perimeter. For those protocols that can traverse a NAT, the NAT device must maintain per-connection state and recompute header checksums to accommodate the header translation. The NAT approach is fundamentally in conflict with the end-to-end principle of network design, which has been the cornerstone of the development of the Internet. Some other examples of limitations introduced by NAT include inability of uniquely identify (for traffic engineering or other purposes) flows in the network that originate from a specific host located behind a NAT as well as the inability to support an entire class of applications that require external hosts to initiate connections to NAT clients.

The impending IPv4 address scarcity has brought renewed attention to the fundamental problem of IPv4 address sharing. A new class of Port Scavenging techniques have recently emerged which leverage unused port numbers to implement IPv4 address sharing [7] without some of the disadvantages of NAT. In general, these techniques assign a unique set of TCP or UDP port numbers to each host sharing a single IP address, and use the port number as part of the unique end-point address identifier to deliver packets to the correct destination. The techniques vary in the method used to deliver the packets and the location within the network at which the port-based routing takes place.

We have developed a new Port Scavenging approach to

IPv4 address sharing called *Port Enhanced ARP* (PE-ARP) that offers important benefits when compared to NAT. PE-ARP is based on three fundamental observations. The first is that each network enabled end device already has a unique identifier (such as the MAC address). The second is that very few source ports are actually used simultaneously by an end host. The third is that a network end point does not necessarily require a unique IP address. It is the applications that run on that end point that require a unique termination point which is usually the IP address, protocol, and port combination. The IP address is simply used to route packets to the host and then the protocol and port numbers are used to identify the correct recipient application.

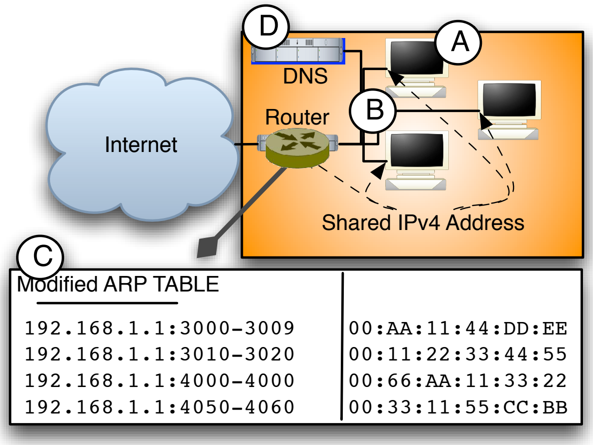
PE-ARP assigns each host a unique, contiguous range of TCP and UDP port numbers. The combination of IP address and port range is used to uniquely identify each host. A modified version of the Address Resolution Protocol (ARP) [8] is used to direct incoming packets to the correct end host. Existing service location mechanisms built into the DNS protocol allow external hosts to locate various services running on the end hosts. No IP layer address translation is done and packets are not modified in transit. No Application-Level Gateway is required to correct the packet header translations of a NAT, and no per-connection state is required on the PE-ARP gateway.

II. End-to-End Consistent IPv4 Address Sharing

Figure 1 describes the overall architecture of address sharing with PE-ARP. Labels A-D indicate changes that PE-ARP introduces into the network. It is important to note that some network scenarios might not require all of them to be implemented. Together A-D enable end hosts to use the entire set of end-to-end networking capabilities, including operation in environments that are typically challenging for NAT based solutions such as networks that host servers. Label A references changes to end hosts to enable port range allocation and management via DHCP. Labels B and C refer to a modified ARP table and protocol while Label D indicates the use of existing DNS protocol in support of PE-ARP. In the subsections below we provide details about how each of these can be implemented.

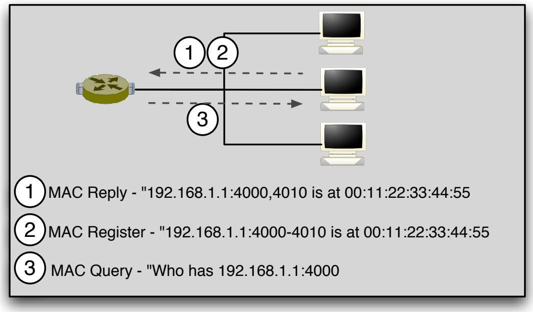
A. End-Host Source Port Range Management

The first component of the PE-ARP system is the end-host port range management module. Each application on a network end-host requests and obtains local port values from the operating system. These values can either be explicitly requested by the application or randomly assigned by the Operating System. To enable IPv4 address sharing, the range of local port values that is used by each end-host must be limited. The purpose of this local port range management agent is to intercept all local port requests from applications and to respond with values from fixed ranges. The ability to limit the range of values from which ports can be used is essential and allows us to reuse the remaining portions on another end-host. This functionality is already available on most popular Operating Systems.



Modified ARP TABLE	
192.168.1.1:3000-3009	00:AA:11:44:DD:EE
192.168.1.1:3010-3020	00:11:22:33:44:55
192.168.1.1:4000-4000	00:66:AA:11:33:22
192.168.1.1:4050-4060	00:33:11:55:CC:BB

Fig. 1. End-to-End Consistent IPv4 Address Sharing



1 MAC Reply - "192.168.1.1:4000,4010 is at 00:11:22:33:44:55

2 MAC Register - "192.168.1.1:4000-4010 is at 00:11:22:33:44:55

3 MAC Query - "Who has 192.168.1.1:4000

Fig. 2. Network Communication Protocol for IP Address and Port Information

B. Network Communication Protocol for IP Address and Port Information

Once an end host has been allocated a particular port range it must be able to send and receive packets for only that port range. There needs to be a mechanism by which the end host can inform the local router or switch the port range for which it will be the valid recipient. We use an extended version of the ARP protocol that is enhanced to include port information. When the local router needs to know what host is using a particular port, the router sends out a broadcast request for the information (PE-ARP_REQUEST). The correct host responds (PE-ARP_REPLY), providing its hardware address and full local port range.

C. Mapping IPv4 packets to network end-hosts

The third component of the overall PE-ARP architecture is the extension to the local ARP table itself. When a packet arrives from the Internet to the local router, the router has to determine the packet's end-host destination. In current networks, the local router stores a mapping between IP addresses and physical MAC addresses. However, we now allow multiple end-hosts to share the same IPv4 address, so this information is no longer sufficient to uniquely identify the destination end-host for a packet. Instead, a modified table is used that employs both IP and port information to determine the MAC address of the end-host for which the packets are intended. The structure of this table is shown in Figure 1. In addition to the IP address, this table includes the range of ports associated with a given host. This table is populated by the modified ARP protocol described in the previous subsection.

D. Enhanced DNS Look-Up Service - Dissemination of IP and Port Information via DNS

The operation of services on well-known ports is a challenge in an environment where the single unique IP address per end-host restriction has been eliminated. Providing services from behind a NAT requires that the NAT be configured to translate the public service port to the end-host's service port. With PE-ARP no port translation is done, but each host can only provide services on those ports which it has been assigned. If the well-known port of the service is not in that host's range, the host cannot provide the service at that port and must use a different port within its range. There is no reason a given service can not be provided on any available port so long as the client can discover the port number. DNS SRV records provide just this capability [9].

An SRV record maps a domain name and a service name to a canonical domain name and a port number. A PE-ARP service host can publish its service ports in this way. One problem with this approach is that not all client applications are capable of using SRV records in place of well-known ports. These changes would need to made to applications which do not support this standard capability correctly. The DNS protocol and implementation already support SRV records and do not need to be modified.

E. Deployment Scenarios and Experiments

We have developed a prototype implementation of PE-ARP based on the descriptions of each of the components in the previous section. Our implementation is based on Linux kernel version 2.6.29.3 and ISC DHCP version 4.1.1. ARP table changes as well as ARP protocol modifications were implemented in the Linux kernel and the port range management functionality was implemented by modifying the DHCP server. We also modified some of the library functions and example applications to use SRV queries.

Figure 3 shows our test setup for the modified edge router. We do not require any changes to the packet forwarding or routing functions of the router. The only changes that are necessary are to the MAC address look-up capability. The gateway in this configuration has two physical interfaces each with an IP address on one of the routed networks. The PE-ARP aware hosts share one or more IP addresses on the local network (198.108.63.0/24).

There are three packet forwarding scenarios to consider; outgoing traffic from the end hosts to the Internet, inbound traffic from the Internet to the hosts, and traffic among the hosts in the local subnet. In the case of outbound traffic, each end host only allows applications to use ports in its configured port range and then forwards the packets to the gateway. The gateway simply forwards these to the Internet without any additional PE-ARP related processing. For inbound traffic the gateway performs a PE-ARP lookup to determine which destination MAC address it should forward each packet to. Any communication among hosts in the local network must also use PE-ARP lookups to identify the correct target.

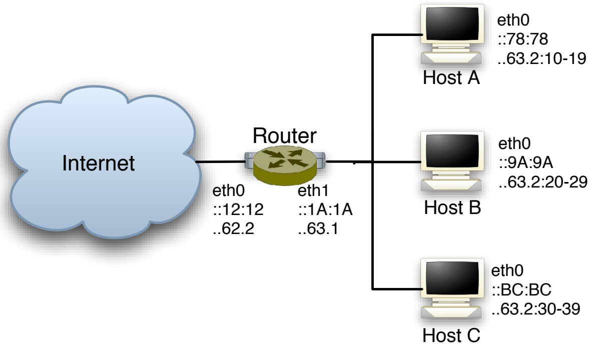


Fig. 3. PE-ARP Router Scenario: Host A, B, and C all share the same IP (198.108.63.2). Each host is limited to a port range. Interfaces are shown with shortened MAC address and IP address/port range (if applicable)

For those situations where it is impractical to modify the local router, we have also implemented a PE-ARP bridge. It operates in the same way as a standard Linux bridge except that it implements PE-ARP on the local network side.

Though not shown in the figure our test network also has a modified DHCP server and a DNS sever that serves SRV records.

III. Discussion

While a full transition to IPv6 is the ultimate solution to the IPv4 address depletion problem, techniques such as PE-ARP can be helpful as an interim solution. In this section we attempt to address some of the more challenging issues that emerge in our approach.

PE-ARP depends on TCP/UDP port numbers to multiplex connections from multiple end hosts. This creates a problem for protocols such as ICMP that do not have port numbers. This can be handled in the PE-ARP gateway in a way similar to the way NAT devices handle it. The gateway can maintain the state of an ICMP query/response by Query ID to sent the reply to the intended recipient. An interesting alternate approach is the use of pseudo-ports, which could be associated with portless protocols to allow them to operate correctly.

Services operating on well-known ports are also challenging to accommodate in PE-ARP. While almost any service can be moved off of its well-known port, making the service port known to the client can be a problem. Wider use of the DNS SRV record can help. It might also be possible to extend PE-ARP to allow arbitrary, non-contiguous port ranges so that any port can be assigned to any server.

While it is technically possible to cascade multiple NAT devices, in practice this defeats many of the workarounds that allow various protocols to operate through a NAT, such as UPNP or IPSec NAT traversal. PE-ARP does not have this problem. PE-ARP gateways can be cascaded as many times as needed until the available port space is exhausted.

Port ranges must be assigned and allocated to hosts and to the PE-ARP gateway. An extra DHCP option can be used to hand out the port ranges, but they still must be allocated in some systematic way, and synchronized between the gateway and the hosts. Using DHCP as the basis for port management eliminates the complexity of an additional stand-alone mechanism. There is currently an Internet draft under consideration at the IETF that proposes a DHCP based port management technique [10] and we intend to implement something similar.

IV. Related Work

Carrier Grade NAT has been proposed as a way to conserve IP addresses in a large network. Like NAT, it changes the address fields in each IP packet, breaking end-to-end consistency. It requires that the translation device maintain state for each connection, which causes scaling problems. There is a tension between placing the translation device at too central a point, which requires large state tables, or too close to the edges, where it can't provide as much benefit in address conservation. PE-ARP does not translate addresses and does not require per-connection state tables.

The most directly related research to PE-ARP has been in the broader Port Scavenging area. Below we describe some of the key proposals related with the port scavenging approach and how PE-ARP differs from them.

Several port scavenging approaches have emerged in recent months. The A+P approach [11] is similar to PE-ARP in that it reclaims unused source port space as a part of the end-host identifier. This was developed largely to address issues and complications of the Carrier Grade NAT technique [12]. However, the A+P scheme continues to rely on the use of a A+P NAT middle device to implement NAT-like capability. End to End NAT [13] uses a NAT but makes the NAT configuration visible to the end host. The host can then reverse the translation so that an application running on the host sees the same network addresses as its peer at the other end of the connection. Address translation still takes place but the endpoints have a consistent view of the connection. Port Range Routing [10] depends on routing infrastructure rather than ARP to deliver packets to the intended destination. Several methods are defined, including encapsulation with a secondary IP address for each end host and use of a source routing option.

Our approach is fundamentally different from other Port Scavenging techniques in that we rely on changes to the ARP protocol to implement IP address sharing in an end-to-end consistent manner.

V. Conclusions and Future Work

PE-ARP presents a unique ARP based approach to Port Scavenging. It allows us to share a single IP address among multiple end-hosts in an end-to-end consistent manner. The IP address and port number are combined to form a unique identifier that is then used to map to a specific MAC address.

We have installed PE-ARP on several test systems on our network. We hope to gain more experience with PE-ARP in a variety of situations including both desktop and server use. We also plan to characterize the scalability of the system and have begun measurements to determine how large a pool of ports is required by a typical host. We intend to extend our implementation by implementing a dynamic port range management mechanism. Our prototype runs on Linux, however, there is nothing OS-specific about it. We would like to implement prototypes on popular consumer operating systems to investigate portability and scaling on these platforms.

We also hope to investigate the use of PE-ARP as part of an IPv6 migration strategy. It should be possible to embed a description of the port ranges into an IPv6 address, which would give us the ability to directly map between PE-ARP host identifiers and IPv6 addresses. We are currently working on an IETF Internet Draft and intend to actively participate in the emerging BoF/mailing list in this new area. As IPv4 address exhaustion approaches, network operators will be required to make the most of increasingly scarce IPv4 address resources. Techniques such as PE-ARP are likely to be a valuable tool for network management in this new era.

References

[1] Geoff Huston. Ipv4 address report. *http://www.potaroo.net/tools/ipv4/index.html*, July 2009.

[2] K. Egevang and P. Francis. The IP Network Address Translator (NAT). RFC 1631 (Informational), May 1994. Obsoleted by RFC 3022.

[3] L. Phifer. The trouble with NAT. *The Internet Protocol Journal*, Dec 2000.

[4] ISO 29341-1:2008. *Information technology – UPnP Device Architecture – Part 1: UPnP Device Architecture Version 1.0*. ISO, Geneva, Switzerland, 2008.

[5] M. Holdrege and P. Srisuresh. Protocol Complications with the IP Network Address Translator. RFC 3027 (Informational), January 2001.

[6] T. Kivinen, B. Swander, and A. Huttunen. Negotiation of NAT-Traversal in the IKE. RFC 3947 (Standards Track), January 2005.

[7] IETF Mailing List. shara Disucssion Archive. *http://www.ietf.org/mail-archive/web/shara/current/maillist.html*, Oct 2009.

[8] D. Plummer. Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 826 (Standard), November 1982. Updated by RFCs 5227, 5494.

[9] A. Gulbrandsen, P. Vixie, and L. Esibov. A DNS RR for specifying the location of services (DNS SRV). RFC 2782 (Proposed Standard), February 2000.

[10] M. Boucadair, P. Levis, G. Bajko, and T. Savolainen. IPv4 Connectivity Access in the Context of IPv4 Address Exhaustion: Port Range based IP Architecture. draft-boucadair-port-range-02.txt (Work In Progress), July 2009.

[11] O. Maennel, R. Bush, L. Cittadi, and S.M. Bellovin. A Better Approach than Carrier-Grade-NAT. *Technical Report CUCS-041-80*, Sep 2008.

[12] A. Durand. Managing 100+ Million IP Addreses. *NANOG37: http://nanog.org/mtg-0606/durand.html*, June 2006.

[13] M. Ohta. End to End NAT. draft-ohta-e2e-nat-00.txt (Work In Progress), July 2009.