Internet Engineering Task Force (IETF)
Request for Comments: 6346
Category: Experimental
ISSN: 2070-1721

R. Bush, Ed.
Internet Initiative Japan

August 2011

The Address plus Port (A+P) Approach to the IPv4 Address Shortage

Abstract

We are facing the exhaustion of the IANA IPv4 free IP address pool. Unfortunately, IPv6 is not yet deployed widely enough to fully replace IPv4, and it is unrealistic to expect that this is going to change before the depletion of IPv4 addresses. Letting hosts seamlessly communicate in an IPv4 world without assigning a unique globally routable IPv4 address to each of them is a challenging problem.

This document proposes an IPv4 address sharing scheme, treating some of the port number bits as part of an extended IPv4 address (Address plus Port, or A+P). Instead of assigning a single IPv4 address to a single customer device, we propose to extend the address field by using bits from the port number range in the TCP/UDP header as additional endpoint identifiers, thus leaving a reduced range of ports available to applications. This means assigning the same IPv4 address to multiple clients (e.g., Customer Premises Equipment (CPE), mobile phones), each with its assigned port range. In the face of IPv4 address exhaustion, the need for addresses is stronger than the need to be able to address thousands of applications on a single host. If address translation is needed, the end-user should be in control of the translation process -- not some smart boxes in the core.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes a technique to deal with the imminent IPv4 address space exhaustion. Many large Internet Service Providers (ISPs) face the problem that their networks' customer edges are so large that it will soon not be possible to provide each customer with a unique public IPv4 address. Therefore, although undesirable, address sharing, in the same molds as NAT, is inevitable.

To allow end-to-end connectivity between IPv4-speaking applications, we propose to extend the semantics of the IPv4 address with bits from the UDP/TCP header. Assuming we could limit the applications' port addressing to any number of bits lower than 16, we can increase the effective size of an IPv4 address by remaining additional bits of up to 16. In this scenario, 1 to 65536 customers could be multiplexed on the same IPv4 address, while allowing them a fixed or dynamic range of 1 to 65536 ports. Customers could, for example, receive an initial fixed port range, defined by the operator, and dynamically request additional blocks, depending on their contract. We call this "extended addressing" or "A+P" (Address plus Port) addressing. The main advantage of A+P is that it preserves the Internet "end-to-end" paradigm by not requiring translation (at least for some ports) of an IP address.

1.1. Problems with Carrier Grade NATs

Various forms of NATs will be installed at different levels and places in the IPv4 Internet to achieve address compression. This document argues for mechanisms where this happens as close to the edge of the network as possible, thereby minimizing damage to the End-to-End Principle and allowing end-customers to retain control over the address and port translation. Therefore, it is essential to create mechanisms to "bypass" NATs in the core, when applicable, and keep the control at the end-user.

With Carrier Grade NATs (CGNs) in the core of the network, the user is trapped behind unchangeable application policies, and the deployment of new applications is hindered by the need to implement the corresponding Application Level Gateways (ALGs) on the CGNs. This is the opposite of the "end-to-end" model of the Internet.

With the smarts at the edges, one can easily deploy new applications between consenting endpoints by merely tweaking the NATs at the corresponding CPE, or even upgrading them to a new version that supports a specific ALG.

Today's NATs are typically mitigated by offering the customers limited control over them, e.g., port forwarding, Universal Plug and Play or the NAT Port Mapping Protocol (UPnP/NAT-PMP). However, this is not expected to work with CGNs. CGN proposals -- other than DS-Lite [RFC6333] with A+P or the Port Control Protocol (PCP) [PCP-BASE] -- admit that it is not expected that applications that require specific port assignment or port mapping from the NAT box will keep working.

Another issue with CGNs is the trade-off between session state and network placement. The farther from the edge the CGN is placed, the more session state needs to be kept due to larger subscriber aggregation and the more disruption that occurs in the case of a failure. In order to reduce the state, CGNs would end up somewhere closer to the edge. Thus, the CGN trades scalability for the amount of state that needs to be kept, which makes optimally placing a CGN a hard engineering problem.

In some deployment scenarios, a CGN can be seen as the single point of failure, and therefore the availability of delivered services can be impacted by a single CGN device. Means to ensure state synchronization and failover would be required to allow for service continuity whenever a failure occurs.

Intra-domain paths may not be optimal for communications between two nodes connected to the same domain deploying CGNs; they may lead to path stretches.

1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Terminology

This document makes use of the following terms:

Public Realm: This realm contains only public routable IPv4 addresses. Packets in this realm are forwarded based on the destination IPv4 address.

A+P Realm: This realm contains both public routable IPv4 and A+P addresses.

A+P Packet: A regular IPv4 packet is forwarded based on the destination IPv4 address and the TCP/UDP port numbers.

Private Realm: This realm contains IPv4 addresses that are not globally routed. They may be taken from the [RFC1918] range. However, this document does not make such an assumption. We regard as private address space any IPv4 address that needs to be translated in order to gain global connectivity, irrespective of whether or not it falls in [RFC1918] space.

Port-Range Router (PRR): A device that forwards A+P packets.

Customer Premises Equipment (CPE): cable or DSL modem.

Provider Edge (PE) Router: Customer aggregation router.

Provider Border Router (BR): Provider's edge to other providers.

Network Core Routers (Core): Provider routers that are not at the edges.

3. Design Constraints and Functions

The problem of address space shortage is first felt by providers with a very large end-user customer base, such as broadband providers and mobile service providers. Though the cases and requirements are slightly different, they share many commonalities. In the following text, we develop a set of overall design constraints for solutions addressing the IPv4 address shortage problem.

3.1. Design Constraints

We regard several constraints as important for our design:

1) End-to-end is under customer control: Customers shall have the ability to deploy new application protocols at will. IPv4 address shortage should not be a license to break the Internet's end-to-end paradigm.

2) Backward compatibility: Approaches should be transparent to unaware users. Devices or existing applications should be able to work without modification. Emergence of new applications should not be limited.

3) Highly scalable and minimal state core: Minimal state should be kept inside the ISP's network. If the operator is rolling out A+P incrementally, it is understood there may be state in the core in the non-A+P part of such a roll-out.

4) Efficiency versus complexity: Operators should have the flexibility to trade off port multiplexing efficiency and scalability and end-to-end transparency.

5) "Double-NAT" should be avoided: Multiple gateway devices might be present in a path, and once one has done some translation, those packets should not be retranslated.

6) Legal traceability: ISPs must be able to provide the identity of a customer from the knowledge of the IPv4 public address and the port. This should have as low an impact as is reasonable on storage by the ISP. We assume that NATs on customer premises do not pose much of a problem, while provider NATs need to keep additional logs.

7) IPv6 deployment should be encouraged. NAT444 strongly biases the users to the deployment of RFC 1918 addressing.

Constraint 5 is important: while many techniques have been deployed to allow applications to work through a NAT, traversing cascaded NATs is crucial if NATs are being deployed in the core of a provider network.

3.2. A+P Functions

The A+P architecture can be split into three distinct functions: encaps/decaps, NAT, and signaling.

Encaps/decaps function: is used to forward port-restricted A+P packets over intermediate legacy devices. The encapsulation function takes an IPv4 packet, looks up the IP and TCP/UDP headers, and puts the packet into the appropriate tunnel. The state needed to perform this action is comparable to a forwarding table. The decapsulation device SHOULD check if the source address and port of packets coming out of the tunnel are legitimate (e.g., see [BCP38]). Based on the result of such a check, the packet MAY be forwarded untranslated, MAY be discarded, or MAY be NATed. In this document, we refer to a device that provides this encaps/decaps functionality as a Port-Range Router (PRR).

Network Address Translation (NAT) function: is used to connect legacy end-hosts. Unless upgraded, end-hosts or end-systems are not aware of A+P restrictions and therefore assume a full IP address. The NAT function performs any address or port translation, including Application Level Gateways (ALGs) whenever required. The state that has to be kept to implement this function is the mapping for which external addresses and ports have been mapped to which internal addresses and ports, just as in CPEs embedding NAT today. A subtle,

but very important, difference should be noted here: the customer has control over the NATing process or might choose to "bypass" the NAT. If this is done, we call the NAT a Large-Scale NAT (LSN). However, if the NAT does NOT allow the customer to control the translation process, we call it a CGN.

Signaling function: is used to allow A+P-aware devices to get to know which ports are assigned to be passed through untranslated and what will happen to packets outside the assigned port range (e.g., could be NATed or discarded). Signaling may also be used to learn the encapsulation method and any endpoint information needed. In addition, the signaling function may be used to dynamically assign the requested port range.

3.3. Overview of the A+P Solution

As mentioned above, the core architectural elements of the A+P solution are three separated and independent functions: the NAT function, the encaps/decaps function, and the signaling function. The NAT function is similar to a NAT as we know it today: it performs a translation between two different address realms. When the external realm is public IPv4 address space, we assume that the translation is many-to-one, in order to multiplex many customers on a single public IPv4 address. The only difference with a traditional NAT (Figure 1) is that the translator might only be able to use a restricted range of ports when mapping multiple internal addresses onto an external one, e.g., the external address realm might be port-restricted.

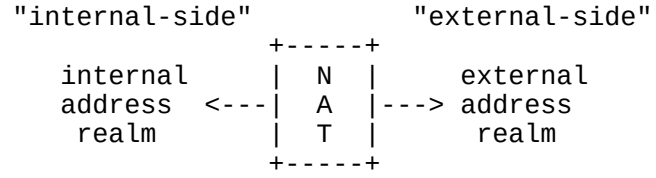


Figure 1: Traditional NAT

The encaps/decaps function, on the other hand, is the ability to establish a tunnel with another endpoint providing the same function. This implies some form of signaling to establish a tunnel. Such signaling can be viewed as integrated with DHCP or as a separate service. Section 3.3.1 discusses the constraints of this signaling function. The tunnel can be an IPv6 or IPv4 encapsulation, a layer-2 tunnel, or some other form of softwire. Note that the presence of a tunnel allows unmodified, naive, or even legacy devices between the two endpoints.

Two or more devices that provide the encaps/decaps function are linked by tunnels to form an A+P subsystem. The function of each gateway is to encapsulate and decapsulate, respectively. Figure 2

depicts the simplest possible A+P subsystem, that is, two devices providing the encaps/decaps function.

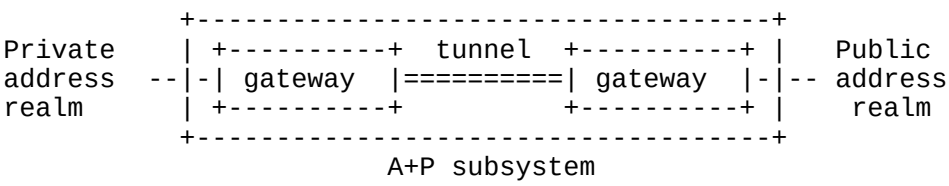


Figure 2: A Simple A+P Subsystem

Within an A+P subsystem, the public address realm is extended by using bits from the port number when forwarding packets. Each device is assigned one address from the external realm and a range of port numbers. Hence, devices that are part of an A+P subsystem can communicate with the public realm without the need for address translation (i.e., preserving end-to-end packet integrity): an A+P packet originated from within the A+P subsystem can be simply forwarded over tunnels up to the endpoint, where it gets decapsulated and routed in the external realm.

3.3.1. Signaling

The following information needs to be available on all the gateways in the A+P subsystem. It is expected that there will be a signaling protocols such as [PR-ADDR-ASSIGN], [SHARED-ADDR-OPT], [PORT-RANGE-OPT], or [PCP-BASE].

The information that needs to be shared is the following:

- a set of public IPv4 addresses,
- for each IPv4 address, a starting point for the allocated port range,
- the number of delegated ports,
- the optional key that enables partial or full preservation of entropy in port randomization -- see [PR-ADDR-ASSIGN],
- the lifetime for each IPv4 address and set of allocated ports,
- the tunneling technology to be used (e.g., "IPv6-encapsulation"),

- addresses of the tunnel endpoints (e.g., IPv6 address of tunnel endpoints),
- whether or not NAT function is provided by the gateway,
- a device identification number and some authentication mechanisms, and
- a version number and some reserved bits for future use.

Note that the functions of encapsulation and decapsulation have been separated from the NAT function. However, to accommodate legacy hosts, NATing is likely to be provided at some point in the path; therefore, the availability or absence of NATing MUST be communicated in signaling, as A+P is agnostic about NAT placement.

The port ranges can be allocated in two different ways:

- If applications or end-hosts behind the CPE are not UPnPv2/NAT-PMP-aware, then the CPE SHOULD request ports via mechanisms, e.g., as described in [PR-ADDR-ASSIGN] and [PORT-RANGE-OPT]. Note that different port ranges can have different lifetimes, and the CPE is not entitled to use them after they expire -- unless it refreshes those ranges. It is up to the ISP to put mechanisms in place (to prevent denial-of-service attacks) that determine what percentage of already allocated port ranges should be exhausted before a CPE may request additional ranges, how often the CPE can request additional ranges, and so on.
- If applications behind the CPE are UPnPv2/NAT-PMP-aware, additional ports MAY be requested through that mechanism. In this case, the CPE should forward those requests to the LSN, and the LSN should reply reporting if the requested ports are available or not (and if they are not available, some alternatives should be offered). Here again, to prevent potential denial-of-service attacks, mechanisms should be in place to prevent UPnPv2/NAT-PMP packet storms and fast port allocation. A detailed description of this mechanism, called PCP, is in [PCP-BASE].

Whatever signaling mechanism is used inside the tunnels -- DHCP, IP Control Protocol (IPCP), or PCP based, synchronization between the signaling server and PRR must be established in both directions. For example, if we use DHCP as the signaling mechanism, the PRR must communicate to the DHCP server at least its IP range. The DHCP server then starts to allocate IP addresses and port ranges to CPEs and communicates back to the PRR which IP and port range have been allocated to which CPE, so the PRR knows to which tunnel to redirect incoming traffic. In addition, DHCP MUST also communicate lifetimes

of port ranges assigned to CPE via the PRR. DHCP server may be co-located with the PRR function to ease address management and also to avoid the need to introduce a communication protocol between the PRR and DHCP.

If UPnPv2/NAT-PMP is used as the dynamic port allocation mechanism, the PRR must also communicate to the DHCP (or IPCP) server to avoid those ports. The PRR must somehow (e.g., using DHCP or IPCP options) communicate back to CPE that the allocation of ports was successful, so CPE adds those ports to existing port ranges.

Note that operation can be even simplified if a fixed length of port ranges is assigned to all customers and no differentiation is implemented based on port-range length. In such case, the binding table maintained by the PRR can be dynamically built upon the receipt of a first packet from a port-restricted device.

3.3.2. Address Realm

Each gateway within the A+P subsystem manages a certain portion of A+P address space; that is, a portion of IPv4 space that is extended by borrowing bits from the port number. This address space may be a single, port-restricted IPv4 address. The gateway MAY use its managed A+P address space for several purposes:

- Allocation of a sub-portion of the A+P address space to other authenticated A+P gateways in the A+P subsystem (referred to as delegation). We call the allocated sub-portion delegated address space.

- Exchange of (untranslated) packets with the external address realm. For this to work, such packets MUST use a source address and port belonging to the non-delegated address space.

If the gateway is also capable of performing the NAT function, it MAY translate packets arriving on an internal interface that are outside of its managed A+P address space into non-delegated address space.

Hence, a provider may have ’islands’ of A+P as they slowly deploy over time. The provider does not have to replace CPE until they want to provide the A+P function to an island of users or even to one particular user in a sea of non-A+P users.

An A+P gateway ("A"), accepts incoming connections from other A+P gateways ("B"). Upon connection establishment (provided appropriate authentication), B would "ask" A for delegation of an A+P address. In turn, A will inform B about its public IPv4 address and will

delegate a portion of its port range to B. In addition, A will also negotiate the encaps/decaps function with B (e.g., let B know the address of the decaps device at the endpoint of the tunnel).

This could be implemented, for example, via a NAT-PMP- or DHCP-like solution. In general, the following rule applies: a sub-portion of the managed A+P address space is delegated as long as devices below ask for it; otherwise, private IPv4 is provided to support legacy hosts.

The following examples use an IPv4 address from the blocks reserved for documentation as defined in [RFC5737].

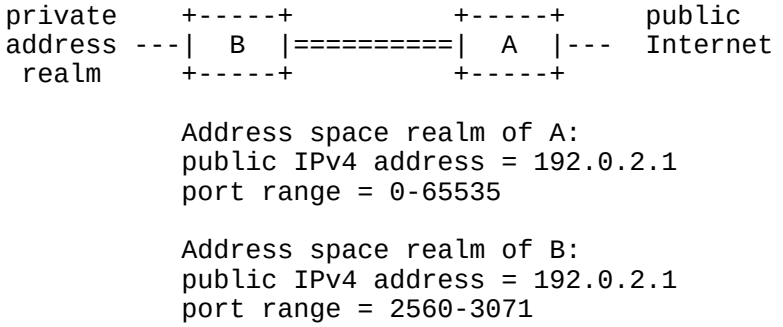


Figure 3: Configuration Example

Figure 3 illustrates a sample configuration. Note that A might actually consist of three different devices: one that handles signaling requests from B; one that performs encapsulation and decapsulation; and, if provided, one device that performs the NATing function (e.g., an LSN). Packet forwarding is assumed to be as follows: in the "outbound" case, a packet arrives from the private address realm to B. As stated above, B has two options: it can either apply or not apply the NAT function. The decision depends upon the specific configuration and/or the capabilities of A and B. Note that NAT functionality is required to support legacy hosts; however, this can be done at either of the two devices A or B. The term "NAT" refers to translating the packet into the managed A+P address (B has address 192.0.2.1 and ports 2560-3071 in the example above). We then have two options:

1) B NATs the packet. The translated packet is then tunneled to A. A recognizes that the packet has already been translated because the source address and port match the delegated space. A decapsulates the packet and releases it in the public Internet.

2) B does not NAT the packet. The untranslated packet is then tunneled to A. A recognizes that the packet has not been translated, so A forwards the packet to a co-located NATing device, which translates the packet and routes it in the public Internet. This device, e.g., an LSN, has to store the mapping between the source port used to NAT and the tunnel where the packet came from, in order to correctly route the reply. Note that A cannot use a port number from the range that has been delegated to B. As a consequence, A has to assign a part of its non-delegated address space to the NATing function.

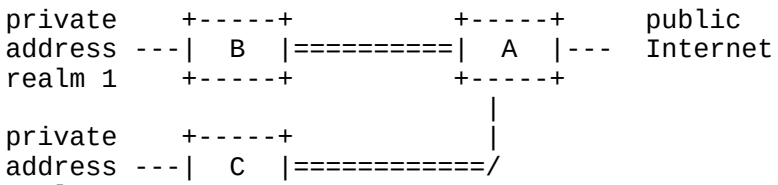
"Inbound" packets are handled in the following way: a packet from the public realm arrives at A. A analyzes the destination port number to understand whether or not the packet needs to be NATed.

1) If the destination port number belongs to the range that A delegated to B, then A tunnels the packet to B. B NATs the packet using its stored mapping and forwards the translated packet to the private domain.

2) If the destination port number is from the address space of the LSN, then A passes the packet on to the co-located LSN, which uses its stored mapping to NAT the packet into the private address realm of B. The appropriate tunnel is stored as well in the mapping of the initial NAT. The LSN then encapsulates the packet to B, which decapsulates it and normally routes it within its private realm.

3) Finally, if the destination port number falls in neither a delegated range nor the address range of the LSN, A discards the packet. If the packet is passed to the LSN, but no mapping can be found, the LSN discards the packet.

Observe that A must be able to receive all IPv4 packets destined to the public IPv4 address (192.0.2.1 in the example), so that it can make routing decisions according to the port number. On the other hand, B receives IPv4 packets destined to the public IPv4 address only via the established tunnel with A. In other words, B uses the public IPv4 address just for translation purposes, but it is not used to make routing decisions. This allows us to keep the routing logic at B as simple as described above, while enabling seamless communication between A+P devices sharing the same public IPv4 address.



Address space realm of A:
public IPv4 address = 192.0.2.1
port range = 0-65535

Address space realm of B:
public IPv4 address = 192.0.2.1
port range = 2560-3071

Address space realm of C:
public IPv4 address = 192.0.2.1
port range = 0-2559

Figure 4: Hierarchical A+P

Consider the example shown in Figure 4. Here, both B and C use the encaps/decaps function to establish a tunnel with A, and they are assigned the same public IPv4 address with different, non-overlapping port ranges. Assume that a host in B's private realm sends a packet destined to address 192.0.2.1 and port 2000, and that B has been instructed to NAT all packets destined to 192.0.2.1. Under these assumptions, B receives the packet and NATs it using its own public IPv4 address (192.0.2.1) and a port selected from its configured port range (e.g., 3000). B then tunnels the translated packet to A. When A receives the packet via the tunnel, it looks at the destination address and port, recognizes C's delegated range, and then tunnels the packet to C. Observe that, apart from stripping the tunnel header, A handles the packet as if it came from the public Internet. When C receives the packet, it NATs the destination address into one address chosen from its private address realm, while keeping the source address (192.0.2.1) and port (3000) untranslated. Return traffic is handled the same way. Such a mechanism allows hosts behind A+P devices to communicate seamlessly even when they share the same public IPv4 address.

Please refer to Section 4 for a discussion of an alternative A+P mechanism that does not incur path-stretch penalties for intra-domain communication.

3.3.3. Reasons for Allowing Multiple A+P Gateways

Since each device in an A+P subsystem provides the encaps/decaps function, new devices can establish tunnels and become in turn part of an A+P subsystem. As noted above, being part of an A+P subsystem implies the capability of talking to the external address realm without any translation. In particular, as described in the previous section, a device X in an A+P subsystem can be reached from the external domain by simply using the public IPv4 address and a port that has been delegated to X. Figure 5 shows an example where three devices are connected in a chain. In other words, A+P signaling can be used to extend end-to-end connectivity to the devices that are in an A+P subsystem. This allows A+P-aware applications (or OSes) running on end-hosts to enter an A+P subsystem and exploit untranslated connectivity.

There are two modes for end-hosts to gain fine-grained control of end-to-end connectivity. The first is where actual end-hosts perform the NAT function and the encaps/decaps function that is required to join the A+P subsystem. This option works in a similar way to the NAT-in-the-host trick employed by virtualization software such as VMware, where the guest operating system is connected via a NAT to the host operating system. The second mode is when applications autonomously ask for an A+P address and use it to join the A+P subsystem. This capability is necessary for some applications that require end-to-end connectivity (e.g., applications that need to be contacted from outside).

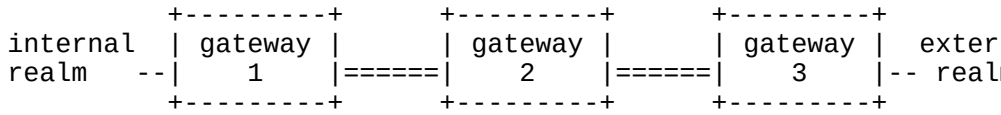


Figure 5: An A+P Subsystem with Multiple Devices

Whatever the reasons might be, the Internet was built on a paradigm that end-to-end connectivity is important. A+P makes this still possible in a time where address shortage forces ISPs to use NATs at various levels. In that sense, A+P can be regarded as a way to bypass NATs.

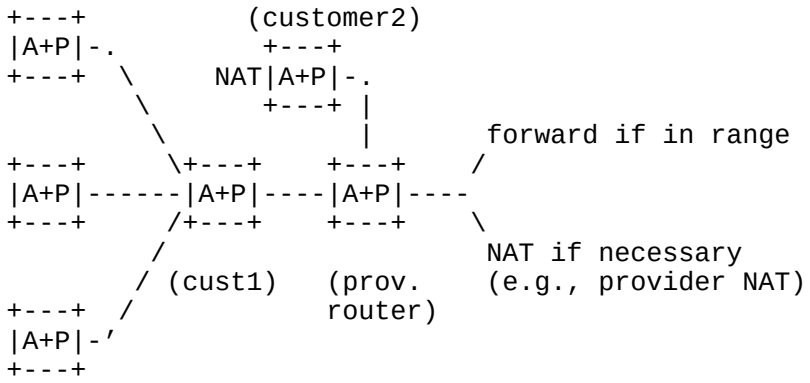


Figure 6: A Complex A+P Subsystem

Figure 6 depicts a complex scenario, where the A+P subsystem is composed of multiple devices organized in a hierarchy. Each A+P gateway decapsulates the packet and then re-encapsulates it again to the next tunnel.

A packet can be NATed either when it enters the A+P subsystem, at intermediate devices, or when it exits the A+P subsystem. This could be, for example, a gateway installed within the provider's network, together with an LSN. Then, each customer operates its own CPE. However, behind the CPE, applications might also be A+P-aware and run their own A+P-gateways; this enables them to have end-to-end connectivity.

One limitation applies when "delayed translation" is used (e.g., translation at the LSN instead of the CPE). If devices using "delayed translation" want to talk to each other, they SHOULD use A+P addresses or out-of-band addressing.

3.3.4. Overall A+P Architecture

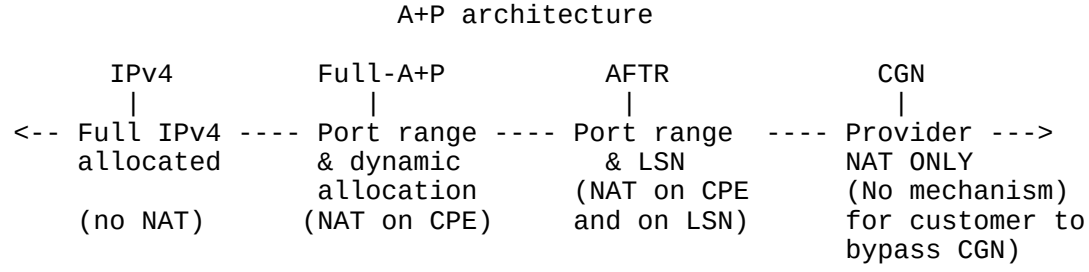


Figure 7: A+P Overall Architecture

The A+P architecture defines various deployment options within an ISP. Figure 7 shows the spectrum of deployment options. On the far left is the common deployment method for broadband subscribers today, an IPv4 address unrestricted with full port range. Full-A+P refers to a port-range allocation from the ISP. The customer must operate an A+P-aware CPE device, and no NATing functionality is provided by the ISP. The Address Family Transition Router (AFTR), such as DS-Lite [RFC6333], is a hybrid. There is NAT present in the core (in this document, referred to as LSN), but the user has the option to "bypass" that NAT in one form or an other, for example, via A+P, NAT-PMP, etc. Finally, a service provider that only deploys CGN will place a NAT in the provider's core and does not allow the customer to "bypass" the translation process or modify ALGs on the NAT. The customer is provider-locked. Notice that all options (besides full IPv4) require some form of tunneling mechanism (e.g., 4in6) and a signaling mechanism (see Section 3.3.1).

3.4. A+P Experiments

There are implementations of A+P as well as documented experiments. France Telecom did experiments that are described in [A+P-EXPERIMENTS]. As seen in that experiment, most tested applications are unaffected. There are problems with torrent protocol and applications, as the listening port is out of A+P port range and some UPnP may be required to make it work with A+P.

Problems with BitTorrent have already been experienced in the wild by users trapped behind a non-UPnP-capable CPE. The current workaround for the end-user is to statically map ports, which can be done in the A+P scenario as well.

BitTorrent tests and experiments in shared IP and port-range environments are well described in [BITTORRENT-ADDR-SHARING]. Conclusions in that document tell us that two limitations were experienced. The first occurred when two clients sharing the same IP address tried to simultaneously retrieve the SAME file located in a SINGLE remote peer. The second limitation occurred when a client tried to download a file located on several seeders, when those seeders shared the same IP address. Mutual file sharing between hosts having the same IP address has been checked. Indeed, machines having the same IP address can share files with no alteration compared to current IP architectures.

Working implementations of A+P can be found in:

- Internet Systems Consortium AFTR (http://www.isc.org/software/aftr),

- o FT Orange opensource A+P (http://opensourceaplusp.weebly.com/) developed by Xiaoyu Zhao, Xiaohong Deng, Tao Zheng, and

- o 4rd (IPv4 Residual Deployment) from ipinfusion.com, which is stateless A+P.

4. Stateless A+P Mapping Function

4.1. Stateless A+P Mapping (SMAP) Gateway Function Description

SMAP stands for Stateless A+P Mapping. This function is responsible for, in a stateless scheme, encapsulating IPv4 packets in IPv6 ones as well as decapsulating IPv4 packets from IPv6 ones. An SMAP function may be hosted in a PRR, end-user device, etc.

As mentioned in Section 4.1 of [RFC6052], the suffix part may enclose the port.

The Stateless A+P Mapping (SMAP) gateway consists in two basic functions as described in Figure 8.

1. SMAP encapsulates an IPv4 packet, destined to a shared IPv4 address, in an IPv6 one. The IPv6 source address is constructed using an IPv4-embedded IPv6 address [RFC6052] from the IPv4 source address and port number plus the IPv6 prefix that has been provisioned to the node performing the SMAP function. The destination IPv6 address is constructed using the shared IPv4 destination address and port number plus the IPv6 prefix that has been provisioned to the SMAP function and that is dedicated to IPv4 destination addresses.

2. SMAP extracts IPv4 incoming packets from IPv6 incoming ones that have IPv6 source addresses belonging to the prefix of the node performing the SMAP function. Extracted IPv4 packets are then forwarded to the point identified by the IPv4 destination address and port number.

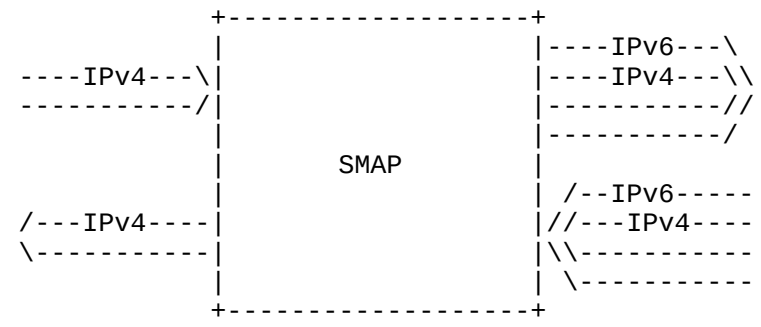


Figure 8: Stateless A+P Mapping Gateway Function

An SMAP-enabled node will perform the stateless 6/4 mapping function for all public shared IPv4 addresses for which it was designated as a stateless 6/4 mapping gateway.

To perform the stateless 6/4 mapping function, an SMAP gateway must:

- be provided with an IPv6 prefix (i.e., Pref6). The SMAP gateway uses this prefix to construct IPv6 source addresses for all IPv4 shared addresses for which it was designated as an SMAP gateway. The IPv6 prefix may be provisioned statically or dynamically (e.g., DHCP).

- be able to know the IPv6 prefix of the node serving as another SMAP gateway for IPv4 destination addresses. This prefix may be known in various ways:

  * Default or Well-Known Prefix (i.e., 64:ff9b::/96) that was provisioned statically or dynamically;

  * Retained at the reception of incoming IPv4-in-IPv6 encapsulated packets;

  * Discovered at the start of communication, thanks to mechanisms such as DNS resolution, for example.

When the SMAP-enabled node receives IPv4 packets with IPv4 source addresses for which it was not designated as an smap gateway, it will not perform stateless 6/4 mapping function for those packets. Those packets will be handled in a classical way (i.e., forwarded, dropped, or locally processed).

When the SMAP-enabled node receives IPv6 packets with IPv6 addresses that do not match with its IPv6 prefix, it will not perform the stateless 6/4 mapping function for those packets. Those packets will be handled in a classical way (i.e., forwarded, dropped, or locally processed).

4.2. Implementation Mode

In this configuration, the node A performs the stateless mapping function on the received IPv4 traffic (encapsulated in IPv6 packets) before forwarding to the node B. The node B performs the stateless mapping function on the received IPv6 traffic (extracting IPv4 packets) before forwarding the IPv4 traffic to the destination identified by the IPv4 destination address and port number. In the opposite direction, and as previously, the node B performs the stateless mapping function on the received IPv4 traffic (encapsulating in IPv6 packets) before forwarding to the node A. The node A performs the stateless mapping function on the received IPv6 traffic (extracting IPv4 packets) before forwarding the IPv4 traffic to the point identified by the IPv4 destination address and port number. In this case, only IPv6 traffic is managed in the network segment between the nodes A and B.

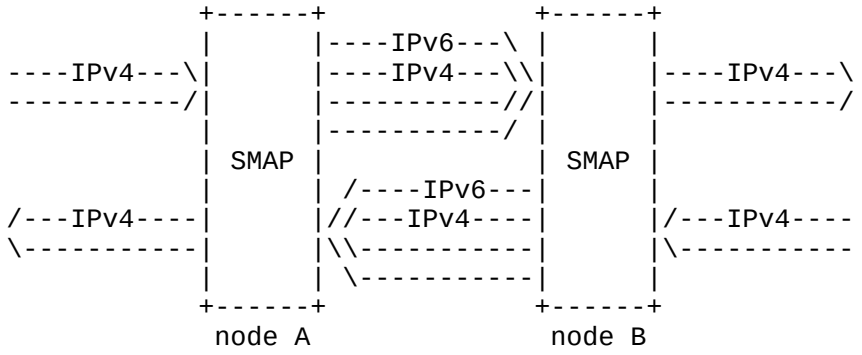


Figure 9

Several deployment scenarios of the SMAP function may be envisaged in the context of port-range-based solutions:

- An SMAP function is embedded in a port-restricted device. Other SMAP-enabled nodes are deployed in the boundaries between IPv6-enabled realms and IPv4 ones. This scenario may be deployed particularly for intra-domain communications so as to interconnect heterogeneous realms (i.e., IPv6/IPv4) within the same Autonomous System (AS).

- An SMAP function is embedded in a port-restricted device. Other SMAP-enabled nodes are deployed in the interconnection segment (with adjacent IPv4-only ones) of a given AS. This deployment scenario is more suitable for service providers targeting the deployment of IPv6 since it eases the migration to full IPv6. Core nodes are not required to continue to activate both IPv4 and IPv6 transfer capabilities.

Other considerations regarding the interconnection of SMAP-enabled domains should be elaborated. The following provides a non-exhaustive list of interconnection schemes.

- The interconnection of two domains implementing the SMAP function may be deployed via IPv4 Internet (Figure 10): this means that IPv4 packets encapsulated in IPv6 packets are transferred using IPv6 until reaching the first SMAP-enabled node. Then, these packets are decapsulated and are forwarded using IPv4 transfer capabilities. A remote SMAP-enabled node will receive those packets and proceed to an IPv4-in-IPv6 encapsulation. These packets are then routed normally until reaching the port-restricted devices that decapsulate the packets.

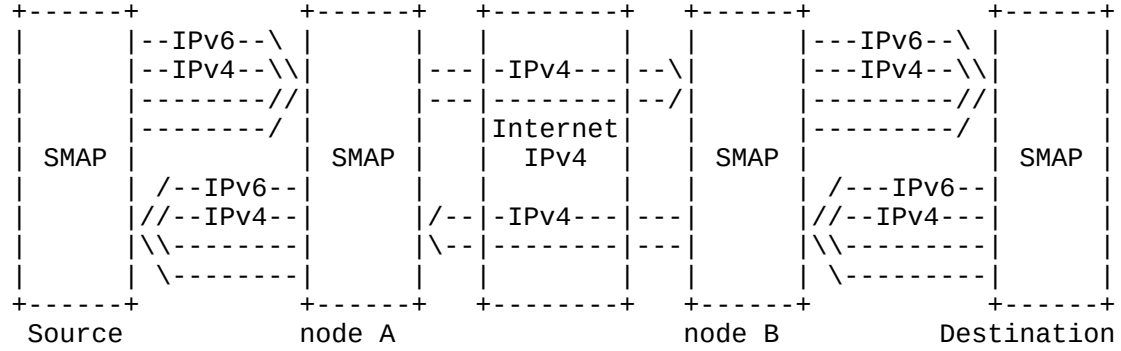


Figure 10: Interconnection Scenario 1

- A second scheme is to use IPv6 to interconnect two realms that implement the SMAP function (Figure 11). An IPv6 prefix (i.e., Pref6) assigned by IANA is used for this service. If appropriate routing configurations have been enforced, then the IPv6-encapsulated packets will be routed until the final destination. In order to implement this model, IPv4-inferred IPv6 prefixes are required to be injected in the IPv6 inter-domain routing tables.

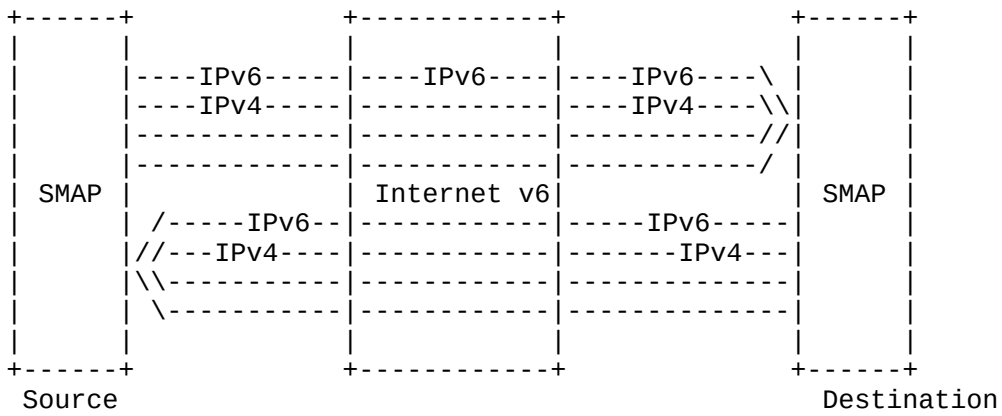


Figure 11: Interconnection Scenario 2

4.3. Towards IPv6-Only Networks

The deployment of the SMAP function allows for smooth migration of networks to an IPv6-only scheme while maintaining the delivery of IPv4 connectivity services to customers. The delivery of IPv4 connectivity services over an IPv6-only network does not require any stateful function to be deployed on the core network. Owing to this A+P mode, both the IPv4 service continuity and the migration to an IPv6-only deployment model are facilitated.

4.4. PRR: On Stateless and Binding Table Modes

The SMAP section (Section 4) discusses two modes: the binding and the stateless modes. Dynamic port allocation is not a feature of the stateless mode, but it is supported in the binding mode. In the binding mode, distinct external IPv4 addresses may be used, but this is not recommended.

o Stateless Mode

Complete stateless mapping implies that the IPv4 address and the significant bits coding the port range are reflected inside the IPv6 prefix assigned to the port-restricted device. This can be achieved either by embedding the full IPv4 address and the significant bits in the IPv6 prefix or by applying an algorithmic approach. Two alternatives are offered when such a stateless mapping is to be enabled:

- use the IPv6 prefix already used for native IPv6 traffic, or

- provide two prefixes to the port-restricted device: one for the native IPv6 traffic and one for the IPv4 traffic.

Note that:

- Providing two IPv6 prefixes has the advantages of allowing a /64 prefix for the port-restricted device along with another prefix (e.g., a /56 or /64) for native IPv6 traffic. This alternative allows the service provider to relate the native IPv6 traffic addressing plan to the IPv4 addressing plan. The drawback is having to allocate two prefixes to each port-restricted device and to route them. In addition, an address selection issue may be encountered.

- Providing one prefix for both needs (e.g., a /56 or a /64) allows the service provider to handle two types of IPv6 prefix for the port-restricted device and in routing tables. But the drawback is that it strongly links the IPv4 addressing plan to the allocated IPv6 prefixes.

As mentioned in Section 4.1 of [RFC6052], the suffix part may enclose the port.

o Binding Table Mode

Another alternative is to assign a "normal" IPv6 prefix to the port-restricted device and to use a binding table, which can be hosted by a service node to correlate the (shared IPv4 address, port range) with an IPv6 address part of the assigned IPv6 prefix. For scalability reasons, this table should be instantiated within PRR-enabled nodes that are close to the port-restricted devices. The number of required entries if hosted at the interconnection segment would be equal to the amount of subscribed users (one per port-restricted device).

4.5. General Recommendations on SMAP

If a Stateless A+P Mapping (SMAP) type of implementation is deployed over intermediate IPv6-only-capable devices, it is recommended that default routes are configured, and the IPv4 routing table is not "leaked" into the IPv6 routing table in terms of having reachability for the packets going towards the Internet.

One of the stateless A+P variants is 4rd [4rd].

5. Deployment Scenarios

5.1. A+P Deployment Models

5.1.1. A+P for Broadband Providers

Some large broadband providers will not have enough public IPv4 address space to provide every customer with a single IP address. The natural solution is sharing a single IP address among many customers. Multiplexing customers is usually accomplished by allocating different port numbers to different customers somewhere within the network of the provider.

It is expected that, when the provider wishes to enable A+P for a customer or a range of customers, the CPE can be upgraded or replaced to support A+P encaps/decaps functionality. Ideally, the CPE also provides NATing functionality. Further, it is expected that at least another component in the ISP network provides the corresponding A+P functionality, and hence is able to establish an A+P subsystem with the CPE. This device is referred to as an A+P router or Port-Range Router (PRR), and could be located close to PE routers. The core of the network MUST support the tunneling protocol (which SHOULD be IPv6, as per Constraint 7) but MAY be another tunneling technology when necessary. In addition, we do not wish to restrict any initiative of customers who might want to run an A+P-capable network on or behind their CPE. To satisfy both Constraints 1 and 2, unmodified legacy hosts should keep working seamlessly, while upgraded/new end-systems should be given the opportunity to exploit enhanced features.

5.1.2. A+P for Mobile Providers

In the case of mobile service providers, the situation is slightly different. The A+P border is assumed to be the gateway (e.g., Gateway GPRS Support Node (GGSN) / Packet Data Network (PDN) gateway (GW) of 3GPP, or Access Service Network (ASN) GW of Worldwide Interoperability for Microwave Access (WiMAX)). The need to extend the address is not within the provider network, but on the edge between the mobile phone devices and the gateway. While desirable, IPv6 connectivity may or may not be provided.

For mobile providers, we use the following terms and assumptions:

1. provider network (PN)

2. gateway (GW)

3. mobile phone device (phone)

4. devices behind the phone, e.g., laptop computer connecting via phone to Internet

We expect that the gateway has a pool of IPv4 addresses and is always in the data-path of the packets. Transport between the gateway and phone devices is assumed to be an end-to-end layer-2 tunnel. We assume that the phone as well as gateway can be upgraded to support A+P. However, some applications running on the phone or devices behind the phone (such as laptop computers connecting via the phone) are not expected to be upgraded. Again, while we do not expect that devices behind the phone will be A+P-aware or upgraded, we also do not want to hinder their evolution. In this sense, the mobile phone would be comparable to the CPE in the broadband provider case; it would be the gateway to the PRR/LSN box in the network of the broadband provider.

5.1.3. A+P from the Provider Network Perspective

ISPs suffering from IPv4 address space exhaustion are interested in achieving a high address space compression ratio. In this respect, an A+P subsystem allows much more flexibility than traditional NATs: the NAT can be placed at the customer and/or in the provider network. In addition, hosts or applications can request ports and thus have untranslated end-to-end connectivity.

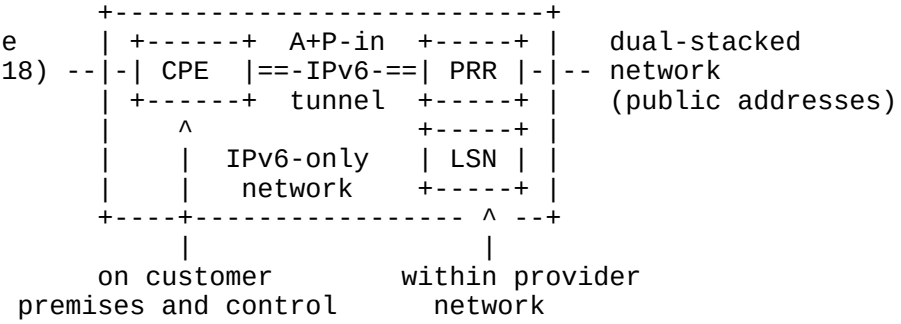


Figure 12: A Simple A+P Subsystem Example

Consider the deployment scenario in Figure 12, where an A+P subsystem is formed by the CPE and a PRR within the ISP core network and preferably is close to the customer edge. Inside the subsystem, packets are forwarded based on address and port. The provider MAY deploy an LSN co-located with the PRR to handle packets that have not been translated by the CPE. In such a configuration, the ISP allows the customer to freely decide whether the translation is done at the

CPE or at the LSN. In order to establish the A+P subsystem, the CPE will be configured automatically (e.g., via a signaling protocol that conforms to the requirements stated above).

Note that the CPE in the example above is provisioned with only an IPv6 address on the external interface.

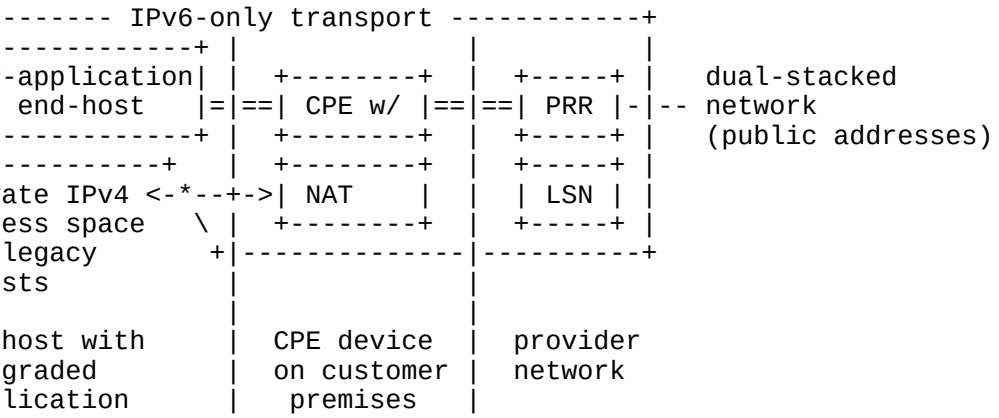


Figure 13: An Extended A+P Subsystem with End-Host Running A+P-Aware Applications

Figure 13 shows an example of how an upgraded application running on a legacy end-host can connect to another host in the public realm. The legacy host is provisioned with a private IPv4 address allocated by the CPE. Any packet sent from the legacy host will be NATed either at the CPE (if configured to do so) or at the LSN (if available).

An A+P-aware application running on the end-host MAY use the signaling described in Section 3.3.1 to connect to the A+P subsystem. In this case, the application will be delegated some space in the A+P address realm, and will be able to contact the public realm (i.e., the public Internet) without the need for translation.

Note that part of A+P signaling is that the NATs are optional. However, if neither the CPE nor the PRR provides NATing functionality, then it will not be possible to connect legacy end-hosts.

To enable packet forwarding with A+P, the ISP MUST install at its A+P border a PRR that encaps/decaps packets. However, to achieve a higher address space compression ratio and/or to support CPEs without NATing functionality, the ISP MAY decide to provide an LSN as well. If no LSN is installed in some part of the ISP’s topology, all CPEs

in that part of the topology MUST support NAT functionality. For reasons of scalability, it is assumed that the PRR is located within the access portion of the network. The CPE would be configured automatically (e.g., via an extended DHCP or NAT-PMP, which has the signaling requirements stated above) with the address of the PRR and of the LSN (if one is being provided). Figure 12 illustrates a possible deployment scenario.

5.2. Dynamic Allocation of Port Ranges

Allocating a fixed number of ports to all CPEs may lead to exhaustion of ports for high-usage customers. This is a perfect recipe for upsetting more demanding customers. On the other hand, allocating to all customers ports sufficiently to match the needs of peak users will not be very efficient. A mechanism for dynamic allocation of port ranges allows the ISP to achieve two goals: a more efficient compression ratio of the number of customers on one IPv4 address and, on the other hand, no limit of the more demanding customers' communication.

Additional allocation of ports or port ranges may be made after an initial static allocation of ports.

The mechanism would prefer allocations of port ranges from the same IP address as the initial allocation. If it is not possible to allocate an additional port range from the same IP, then the mechanism can allocate a port range from another IP within the same subnet. With every additional port range allocation, the PRR updates its routing table. The mechanism for allocating additional port ranges may be part of normal signaling that is used to authenticate the CPE to the ISP.

The ISP controls the dynamic allocation of port ranges by the PRR by setting the initial allocation size and maximum number of allocations per CPE, or the maximum allocations per subscription, depending on subscription level. There is a general observation that the more demanding customer uses around 1024 ports when heavily communicating. So, for example, a first suggestion might be 128 ports initially and then dynamic allocations of ranges of 128 ports up to 511 more allocations maximum. A configured maximum number of allocations could be used to prevent one customer acting in a destructive manner should they become infected. The maximum number of allocations might also be more finely grained, with parameters of how many allocations a user may request per some time frame. If this is used, evasive applications may need to be limited in their bad behavior; for example, one additional allocation per minute would considerably slow a port request storm.

There is likely no minimum request size. This is because A+P-aware applications running on end-hosts MAY request a single port (or a few ports) for the CPE to be contacted on (e.g., Voice over IP (VoIP) clients register a public IP and a single delegated port from the CPE, and accept incoming calls on that port). The implementation on the CPE or PRR will dictate how to handle such requests for smaller blocks: for example, half of available blocks might be used for "block-allocations", 1/6 for single port requests, and the rest for NATing.

Another possible mechanism to allocate additional ports is UPnP/NAT-PMP (as defined in Section 3.3.1), if applications behind CPE support it. In the case of the LSN implementation (DS-Lite), as described in Section 3.3.4 about the A+P overall architecture, signaling packets are simply forwarded by the CPE to the LSN and back to the host running the application that requested the ports, and the PRR allocates the requested port to the appropriate CPE. The same behavior may be chosen with AFTR, if requested ports are outside of the static initial port allocation. If a full A+P implementation is selected, then UPnPv2/NAT-PMP packets are accepted by the CPE, processed, and the requested port number is communicated through the normal signaling mechanism between CPE and PRR tunnel endpoints (PCP).

5.3. Example of A+P-Forwarded Packets

This section provides a detailed example of A+P setup, configuration, and packet flow from an end-host connected to an A+P service provider to any host in the IPv4 Internet, and how the return packets flow back. The following example discusses an A+P-unaware end-host, where the NATing is done at the CPE. Figure 14 illustrates how the CPE receives an IPv4 packet from the end-user device. We first describe the case where the CPE has been configured to provide the NAT functionality (e.g., by the customer through interaction with a website or by automatic signaling). In the following, we call a packet that is translated at the CPE an "A+P-forwarded packet", an analogy with the port-forwarding function employed in today’s CPEs. Upon receiving a packet from the internal interface, the CPE translates, encapsulates, and forwards it to the PRR. The NAT on the CPE is assumed to have a default route to the public realm through its tunnel interface.

When the PRR receives the A+P-forwarded packet, it decapsulates the inner IPv4 packet and checks the source address. If the source address does match the range assigned to A+P-enabled CPEs, then the PRR simply forwards the decapsulated packet onward. This is always the case for A+P-forwarded packets. Otherwise, the PRR assumes that the packet is not A+P-forwarded and passes it to the LSN function,

which in turn translates and forwards the packet based on the destination address. Figure 14 shows the packet flow for an outgoing A+P-forwarded packet.

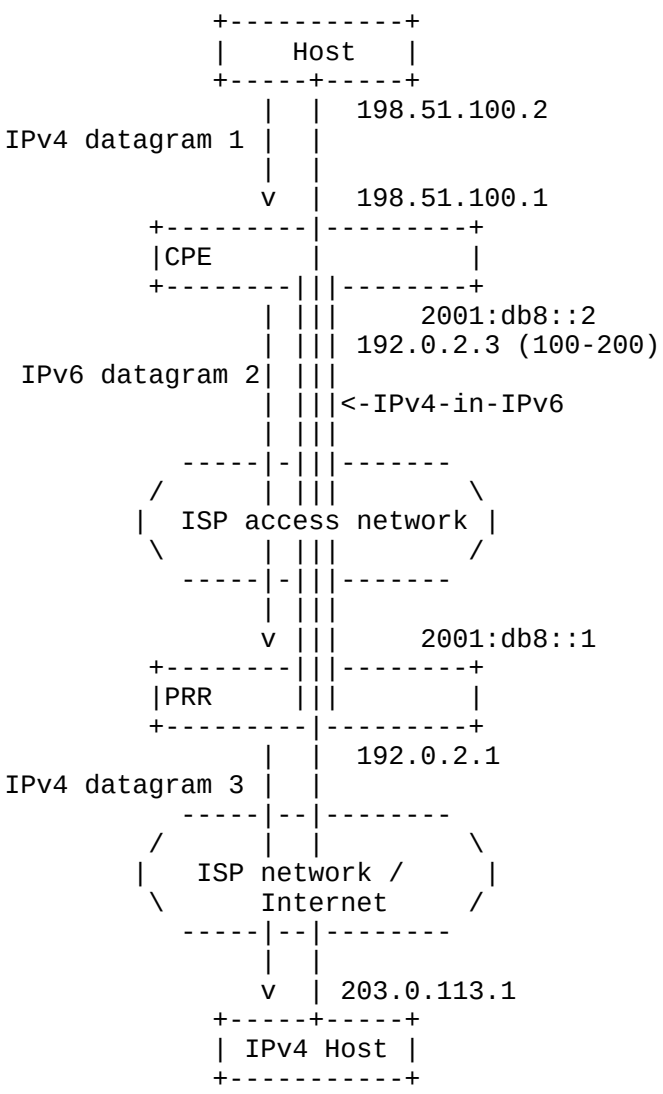


Figure 14: Forwarding of Outgoing A+P-Forwarded Packets

Datagram	Header field	Contents
IPv4 datagram 1	IPv4 Dst	203.0.113.1
	IPv4 Src	198.51.100.2
	TCP Dst	80
	TCP Src	8000
---	---	---
IPv6 datagram 2	IPv6 Dst	2001:db8::1
	IPv6 Src	2001:db8::2
	IPv4 Dst	203.0.113.1
	IPv4 Src	192.0.2.3
	TCP Dst	80
	TCP Src	100
---	---	---
IPv4 datagram 3	IPv4 Dst	203.0.113.1
	IPv4 Src	192.0.2.3
	TCP Dst	80
	TCP Src	100

Datagram Header Contents

An incoming packet undergoes the reverse process. When the PRR receives an IPv4 packet on an external interface, it first checks whether or not the destination address falls within the A+P CPE delegated range. If the address space was delegated, then the PRR encapsulates the incoming packet and forwards it through the appropriate tunnel for that IP/port range. If the address space was not delegated, the packet would be handed to the LSN to check if a mapping is available.

Figure 15 shows how an incoming packet is forwarded, under the assumption that the port number matches the port range that was delegated to the CPE.

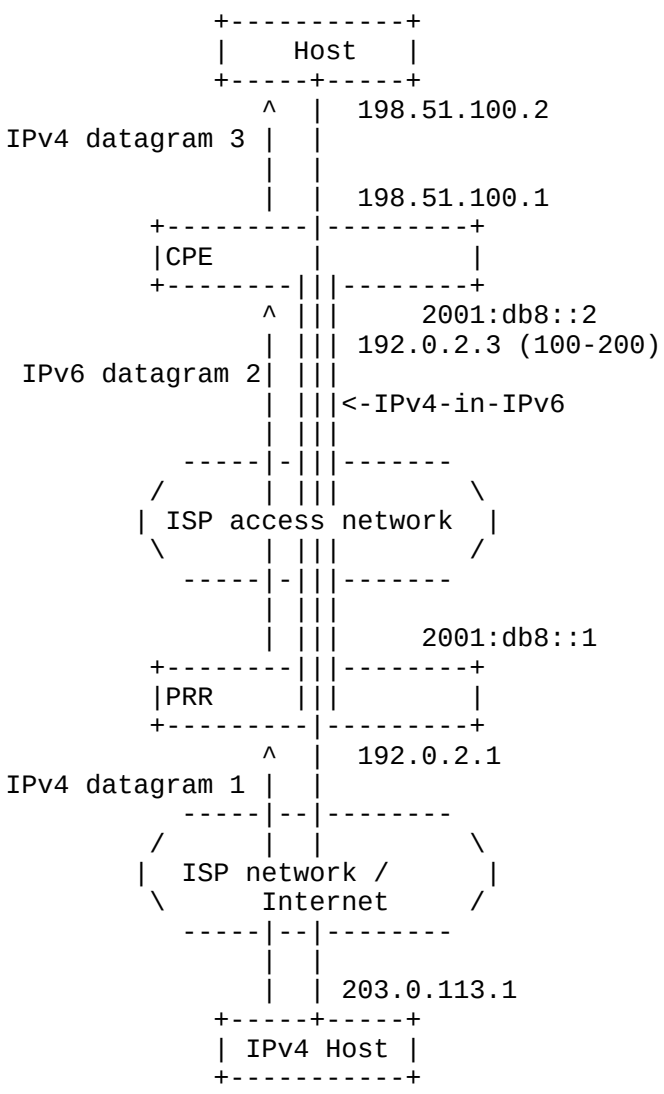


Figure 15: Forwarding of Incoming A+P-Forwarded Packets

Datagram	Header field	Contents
IPv4 datagram 1	IPv4 Dst	198.51.100.3
	IPv4 Src	203.0.113.1
	TCP Dst	100
	TCP Src	80
---	---	---
IPv6 datagram 2	IPv6 Dst	2001:db8::2
	IPv6 Src	2001:db8::1
	IPv4 Dst	198.51.100.3
	IP Src	203.0.113.1
	TCP Dst	100
	TCP Src	80
---	---	---
IPv4 datagram 3	IPv4 Dst	198.51.100.2
	IPv4 Src	203.0.113.1
	TCP Dst	8000
	TCP Src	80

Datagram Header Contents

Note that datagram 1 travels untranslated up to the CPE; thus, the customer has the same control over the translation as he has today -- a home gateway with customizable port-forwarding.

5.3.1. Forwarding of Standard Packets

Packets for which the CPE does not have a corresponding port-forwarding rule are tunneled to the PRR that provides the LSN function. We underline that the LSN MUST NOT use the delegated space for NATing. See [RFC6333] for network diagrams that illustrate the packet flow in this case.

5.3.2. Handling ICMP

ICMP is problematic for all NATs because it lacks port numbers. A+P routing exacerbates the problem.

Most ICMP messages fall into one of two categories: error reports or ECHO/ECHO replies (commonly known as "pings"). For error reports, the offending packet header is embedded within the ICMP packet; NAT devices can then rewrite that portion and route the packet to the actual destination host. This functionality will remain the same with A+P; however, the PRR will need to examine the embedded header to extract the port number, while the A+P gateway will do the necessary rewriting.

ECHO and ECHO replies are more problematic. For ECHO, the A+P gateway device must rewrite the "Identifier" and perhaps "Sequence Number" fields in the ICMP request, treating them as if they were port numbers. This way, the PRR can build the correct A+P address for the returning ECHO replies, so they can be correctly routed back to the appropriate host in the same way as TCP/UDP packets. Pings originated from the public realm (Internet) towards an A+P device are not supported.

5.3.3. Fragmentation

In order to deliver a fragmented IP packet to its final destination (among those having the same IP address), the PRR should activate a dedicated procedure similar to the one used by [RFC6146], Section 3.5, in the sense that it should reassemble the fragments in order to look at the destination port number.

Note that it is recommended to use a Path MTU Discovery (PMTUD) mechanism (e.g., [RFC1191]).

Security issues related to fragmentation are out of scope of this document. For more details, refer to [RFC1858].

5.3.4. Limitations of the A+P Approach

One limitation that A+P shares with any other IP-address-sharing mechanism is the availability of well-known ports. In fact, services run by customers that share the same IP address will be distinguished by the port number. As a consequence, it will be impossible for two customers who share the same IP address to run services on the same port (e.g., port 80). Unfortunately, working around this limitation usually implies application-specific hacks (e.g., HTTP and HTTPS redirection), discussion of which is out of the scope of this document. Of course, a provider might charge more for giving a customer the well-known port range, 0..1024, thus allowing the customer to provide externally available services. Many applications require the availability of well-known ports. However, those applications are not expected to work in an A+P environment unless they can adapt to work with different ports. Such applications do not work behind today’s NATs either.

Another problem that is common to all NATs is coexistence with IPsec. In fact, a NAT that also translates port numbers prevents the Authentication Header (AH) and Encapsulating Security Payload (ESP) from functioning properly, both in tunnel and in transport mode. In this respect, we stress that, since an A+P subsystem exhibits the same external behavior as a NAT, well-known workarounds (such as [RFC3715]) can be employed.

A+P, as all other port-sharing solutions, suffers from the issues documented in [RFC6269], but that’s something we’ll have to live with.

For the host-based A+P, issues related to application conflicts when trying to bind to an out-of-range port are to be further assessed. Note that extensions to the host-based model have been proposed in the past (e.g., the Port-Enhanced Address Resolution Protocol (ARP) extension documented in http://software.merit.edu/pe-arp/).

5.3.5. Port Allocation Strategy Agnostic

Issues raised by [PR-IP-ISSUES] have been analyzed in [STATELESS-4v6]. As seen in that document, most of the issues apply to host-based port-sharing solutions. A+P is not intended to be a host-based port-sharing solution.

The conclusion of [STATELESS-4v6] is that the set of issues specifically attributed to A+P either do not apply to CPE-based flavors or can be mitigated. The A+P solution represents a reasonable trade-off compared to alternatives in areas such as binding logging (for data storage purposes) and ease of deployment and operations, all of which are actually facilitated by such a solution.

6. Security Considerations

With CGNs/LSNs, tracing hackers, spammers, and other criminals will be difficult, requiring logging, recording, and storing of all connection-based mapping information. The need for storage implies a trade-off. On one hand, the LSNs can manage addresses and ports as dynamically as possible in order to maximize aggregation. On the other hand, the more quickly the mapping between private and public space changes, the more information needs to be recorded. This would cause concern not only for law enforcement services, but also for privacy advocates.

A+P offers a better set of trade-offs. All that needs to be logged is the allocation of a range of port numbers to a customer. By design, this will be done rarely, improving scalability. If the NAT functionality is moved further up the tree, the logging requirement will be as well, increasing the load on one node, but giving it more resources to allocate to a busy customer, perhaps decreasing the frequency of allocation requests.

The other extreme is A+P NAT on the customer premises. Such a node would be no different than today's NAT boxes, which do no such logging. We thus conclude that A+P is no worse than today's situation, while being considerably better than CGNs.

7. Acknowledgments

The authors wish to especially thank Remi Despres and Pierre Levis for their help on the development of the A+P approach. We also thank David Ward for review, constructive criticism, and interminable questions, and Dave Thaler for useful criticism on "stackable" A+P gateways. We would also like to thank the following persons for their feedback on earlier versions of this work: Rob Austein, Gert Doering, Dino Farinacci, Russ Housley, Ruediger Volk, Tina Tsou, and Pasi Sarolahti.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

[4rd] Despres, R., Matsushima, S., Murakami, T., and O. Troan, "IPv4 Residual Deployment across IPv6-Service networks (4rd) ISP-NAT's made optional", Work in Progress, March 2011.

[A+P-EXPERIMENTS] Deng, X., Boucadair, M., and F. Telecom, "Implementing A+P in the provider's IPv6-only network", Work in Progress, March 2011.

[BCP38] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[BITTORRENT-ADDR-SHARING] Boucadair, M., Grimault, J., Levis, P., and A. Villefranque, "Behavior of BitTorrent service in an IP Shared Address Environment", Work in Progress, March 2011.

[PCP-BASE] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", Work in Progress, July 2011.

[PORT-RANGE-OPT]
Boucadair, M., Levis, P., Bajko, G., Savolainen, T., and T. ZOU), "Huawei Port Range Configuration Options for PPP IPCP", Work in Progress, June 2011.

[PR-ADDR-ASSIGN]
Bajko, G., Savolainen, T., Boucadair, M., and P. Levis, "Port Restricted IP Address Assignment", Work in Progress, September 2010.

[PR-IP-ISSUES]
Thaler, D., "Issues With Port-Restricted IP Addresses", Work in Progress, February 2010.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1858] Ziemba, G., Reed, D., and P. Traina, "Security Considerations for IP Fragment Filtering", RFC 1858, October 1995.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC3715] Aboba, B. and W. Dixon, "IPsec-Network Address Translation (NAT) Compatibility Requirements", RFC 3715, March 2004.

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", RFC 5737, January 2010.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6269] Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, June 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

[SHARED-ADDR-OPT]
Boucadair, M., Levis, P., Grimault, J., Savolainen, T., and G. Bajko, "Dynamic Host Configuration Protocol (DHCPv6) Options for Shared IP Addresses Solutions", Work in Progress, December 2009.

[STATELESS-4v6]
Dec, W., Asati, R., Bao, C., and H. Deng, "Stateless 4Via6 Address Sharing", Work in Progress, July 2011.

9. Contributing Authors

This document has nine primary authors.

Gabor Bajko
Nokia
EMail: gabor.bajko@nokia.com

Mohamed Boucadair
France Telecom
3, Av Francois Chateaux
Rennes 35000
France
EMail: mohamed.boucadair@orange-ftgroup.co

Steven M. Bellovin
Columbia University
1214 Amsterdam Avenue
MC 0401
New York, NY 10027
US
Phone: +1 212 939 7149
EMail: bellovin@acm.org

Randy Bush
Internet Initiative Japan
5147 Crystal Springs
Bainbridge Island, Washington 98110
US
Phone: +1 206 780 0431 x1
EMail: randy@psg.com

Luca Cittadini
Universita’ Roma Tre
via della Vasca Navale, 79
Rome, 00146
Italy
Phone: +39 06 5733 3215
EMail: luca.cittadini@gmail.com

Olaf Maennel
Loughborough University
Department of Computer Science - N.2.03
Loughborough
United Kingdom
Phone: +44 115 714 0042
EMail: o@maennel.net

Reinaldo Penno
Juniper Networks
1194 North Mathilda Avenue
Sunnyvale, California 94089
US
EMail: rpenno@juniper.net

Teemu Savolainen
Nokia
Hermiankatu 12 D
TAMPERE, FI-33720
Finland
EMail: teemu.savolainen@nokia.com

Jan Zorz
Go6 Institute Slovenia
Frankovo naselje 165
Skofja Loka, 4220
Slovenia
EMail: jan@go6.si

Editor’s Address

Randy Bush (editor)
Internet Initiative Japan
5147 Crystal Springs
Bainbridge Island, Washington 98110
US

Phone: +1 206 780 0431 x1
EMail: randy@psg.com