Why are NATs so popular?

September 2003

Geoff Huston

Last month in this column the analysis of the consumption of IPv4 addresses came up with an observation that was highly likely that that we had quite some time to go before we are in the position of exhausting the IPv4 address pool. Even though 4.4 billion addresses is a large number, this is still a surprising observation. The Internet is indeed everywhere these days, in every office, in every home, and with the introduction of various forms of wireless services, its now embarking on a quest to be in everyone's pocket as well. Now that's a lot of pockets and a lot of devices to connect to the Internet. The production runs for PDA's alone appears to be in quantities of tens of millions of units per year, and the production run for the entire collection of IP-capable devices is in the order of hundreds of millions of devices per year. If every single IP-capable device were configured with its own permanent unique IP address its possible that we would have already run out of IP v4 addresses!

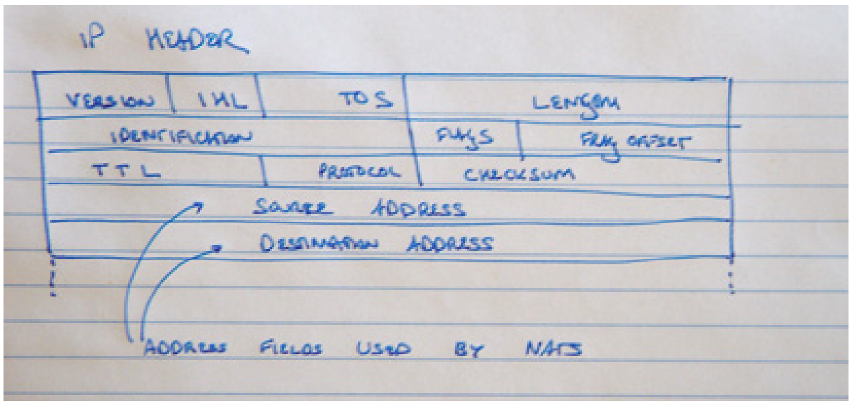
Obviously, this has not happened. So what has happened to allow all these devices to use the Internet, yet at the same time using a little under a third of the total address space in the routing table? The evidence points to very widespread use of NATs or Network Address Translators as the means of compressing the address requirements of all these devices into a fraction of the space.

While there has been no definitive census of the number of devices configured behind NATs, NATS are very widely used in all kinds of contexts, whether its used to interface a private corporate network to the Internet, or used within an ISP environment to provide private addresses to the ISPs' customers. Its fair to predict that not only will some readers pick up this column from behind a NAT, but that the majority of readers will be behind a NAT. It appears that more of the Internet lives a NAT-sheletered existence than the 'exposed' Internet of public addresses.

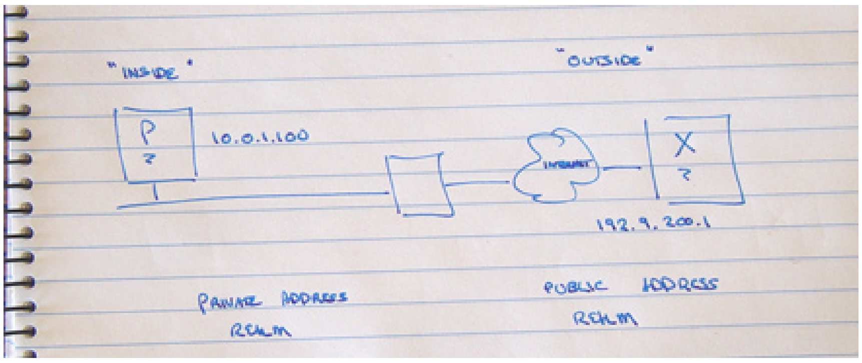
By its very nature of operation its difficult to undertake any form of census of the number of devices that are lurking behind NATs. A recent approach has been proposed by Steve Bellovin, described in a paper: http://www.research.att.com/~smb/papers/fnat.pdf, but to date no estimates of the population of NATted devices using this approach has been published. Estimates range from 30% to 400% of the currently visible Internet population, giving some indication of the difficulty inherent in peering into the interior of NAT space.

Despite their widespread use, we keep on hearing that NATs are a poor technology choice, and that they make some applications perform poorly, while prevent others from working at all. NATs, they claim, are worse that just bad - NATs are evil! So if what they claim is true, then why have NATs become so popular? Why has an entire industry collectively invested so much in a very poor technology? Lets look at NATs in a little closer detail, and see if we can provide some clues as to what's happening here.

NATs are active units placed in the data path. They intercept all packets, and may forward the packet on, with or without alteration, or may elect to discard the packet. NATs have an 'inside' and an 'outside', and undertake different operations on intercepted packets depending on whether the packet is going from the 'inside' to the 'outside' or in the opposite direction. NATs are IP header translators. The header of an IP packet contains the source and destination IP addresses. The destination address is the address of where the packet is destined. The source address is the address of where the packet originated from. When the packet is delivered, the response is generated by swapping the source and destination addresses. In other words the source address is the address the sender wants others to use to reach it.



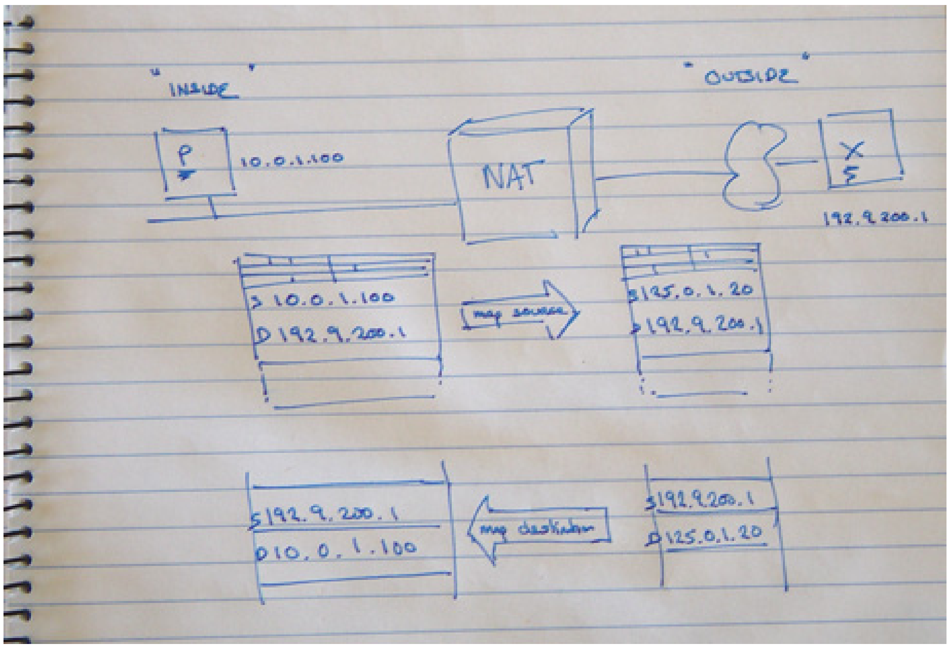
In the direction from the inside to the outside a NAT will rewrite the source address in the packet header to a different value (and alter the header checksum of the packet header at the same time). When a packet is received from the outside, the destination address is rewritten to a new value (and again the header checksum is recalculated). This function sounds very simple, but perhaps somewhat pointless! Why would anyone want to transform packet headers in such a fashion? The answer lies in private addresses.



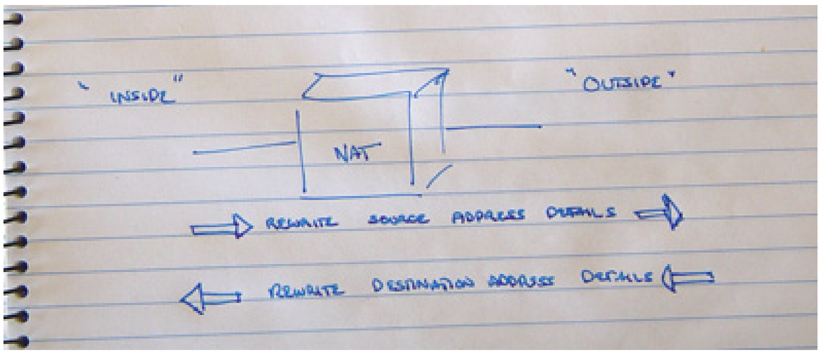
In the figure above, how can device P send an IP packet to X? Well if P does the usual thing then it will first look up the DNS to find the IP address for X, and then create an IP packet using X's address as the destination address and P's local address as the source, and pass the packet to the local network for delivery. If the packet was delivered to X without any further alteration, then X would be unable to respond. The public Internet does not (or should not at any rate!) carry private addresses.
With a NAT between P and X, the NAT will intercept P's outgoing packet and rewrite the source address with some other address. What address should the NAT box use? NATs are configured with a pool of public addresses. When an 'insider' sends an outbound packet, an address is drawn from the pool and mapped as a temporary alias to the inside address. This mapped address is used as the new source address for the outgoing packet.

Once this mapping is made all subsequent packets within this application stream from this internal address to the specified external address will also have their source address mapped to the external address in the same fashion.

When a packet arrives on the external interface, the destination address is checked. If its one of the NAT pool addresses, the NAT box looks up its translation table. If it finds a corresponding table entry the destination address is mapped to the local internal address and the packet is forwarded. If there is no current mapping entry for the destination address, the packet is discarded.



So the mode of operation of a NAT is shown in the figure below:


"INSIDE"
"OUTSIDE"
NAT
REWRITE SOURCE ADDRESS DETAILS
REWRITE DESTINATION ADDRESS DETAILS

A variant of the NAT is the Port-translating NAT (or NAPT) This form of NAT is used in the context of TCP and UDP sessions, where there is a pair of source and destination port addresses as well as the pair of source and destination IP addresses. Outgoing packets have both their source port and source IP addresses altered by the NAT, and incoming packets have their destination port and IP addresses altered by the NAT.

Why bother with port translation as well? Aren't straight address translations enough? Surprisingly NATs can be relatively profligate with addresses. If each TCP session from the same local host is assigned a different and unique external pool address, then the peak address demands on the external address pool could readily match or exceed the number of local hosts, in which case the NAT could be consuming more public addresses than if there were no NAT at all! NAPTs allow concurrent outgoing sessions to be distinguished by the combination of the mapped address and mapped port value. In this way each external pool address may be used for up to 65535 concurrent mapped sessions.

The major limitation of NATs and NAPTs is that an interior local device cannot be reached from the outside unless it has already initiated a session with an external device, and even then the mode of communication is limited. In the case of TCP sessions its limited to the specific external device and to the service on that device. This limitation also applies to UDP with NAPTs. UDP with NATs are often slightly different, and once a UDP association has been forged across a NAT its commonly the case that any other device can direct UDP packets to the local host assuming that they know of the NATs current mapping of external to internal addresses.

The NAT-created translations are temporary. In the case of TCP the NAT and NAPT examine TCP flags, and the translation will be pulled down when a TCP reset (RST) or close (FIN) flag is seen. Its commonly the case that an inactivity timer is also used to terminate the translation following a certain period of no packets at all. Its also possible that an overall session timer is used, and the translation will be terminated some time after it is started. There are also a related set of timers for UDP, with DNS translations separately placed on short inactivity timers, often around 60 seconds, while other forms of UDP traffic are typically placed on 300 second inactivity timers.

So, in essence, NATs allow a local network of systems using private addresses share a smaller pool of public addresses, and at the same time impose a 'speak only when spoken to' admission control regime on external communications.

Where do NATs work? Perhaps surprisingly, NATs work in many situations and for many applications. These include the majority of today's most popular applications. Client / Server applications, such as web page retrieval, mail reader clients, mail sending, and data transfers all work reasonably well for most of the time. As long as the 'insider' can initiate the application, the NAT sets up a translation table on the fly as the session is set up, the translation can be held up for the duration of the session and torn down when the session completes.

No only do NATs mostly work, NATs are very common. So why has the market been so enthusiastic about NATs?

> The huge advantage of this approach is that it can be installed incrementally, without changes to either hosts or routers. (A few unusual applications may require changes). As such, this solution can be implemented and experimented with quickly. If nothing else, this solution can serve to provide temporarily relief while other, more complex and far-reaching solutions are worked out.
> Egevang & Fancis, "Network Address Translator", RFC 1631

Well, a feature list of the selling points of NATs would have to include the following:

- End hosts and local routers don't change. Whether there's a NAT in place between the local network and the Internet or not, local devices can use the same software, and support the same applications. NATs don't require customized versions of operating systems or router images.
- As long as you accept the limitation that sessions must be initiated from the 'inside', NATs mostly work in an entirely transparent fashion.
- And the services and usage scenarios that are not supported by NATs are often perceived as 'unwelcome' or 'unsafe'.
- In this vein NATs are often perceived as part of a site's security architecture, providing protection from attacks launched from the outside towards the inside network.

- It conserves its use of V4 address space. We have heard much about the imminent exhaustion of IPv4 address space, and there is a certain amount of reluctance to consume public address space in contexts of semi-private use where NATs will suffice.
- It allows previously disconnected privately addressed network to connect to the global Internet without any form of renumbering or host changes - and renumbering networks can be a very time consuming, disruptive and expensive operation, or, in other words, renumbering is hard.
- NAT address space is effective provider independent. NATs allows for rapid switching to a different upstream provider, by renumbering the NAT address pool to the new provider's address space. In essence NATs provide the local network manager with the flexibility of using provider independent space without having to meet certain size and use requirements that would normally be required for an allocation of public provider independent address space.
- NATs allows the network administrator to exercise some control over the form of network transactions that can occur between local hosts and the public network.
- NATs requires no local device or application changes. This is perhaps one of the major 'features' of NATs, in that the local network requires no changes in configuration to operate behind a NAT.
- NATs do not require a coordinated deployment. There is no transition, and no 'flag day'. Each local network manager can make an independent decision whether or not to use a NAT. This allows for incremental deployment without mutual dependencies.
- These days the common theme of the public address assignment policy stresses conservative use of address space with minimum wastage. The standard benchmark is to be able to show that a target of 80% of assigned address space is assigned to number connected devices. Achieving such a very high utilization rate is a challenging task in many network scenarios, and NATs represent an alternative approach where the local network can be configured using private addresses without reference to the use of public addresses.
- NATs are very widely available and bundled into a large variety of gateway and firewall units. In many units NATs are not an optional extra - they are configured in as a basic item of product functionality.

So the market has taken NATs and embraced them wholeheartedly. And in a market-oriented business environment, what's wrong with that?
Unfortunately NATS represent a set of design compromises, and no delving into the world of NATs would be complete without exploring some of NATs shortcomings. So after extolling their benefits its now necessary to enumerate some of the broken aspects of the world of NATs.

This solution has the disadvantage of taking away the end-to-end significance of an IP address, and making up for it with increased state in the network.
Egevang & Francis, "Network Address Translator", RFC 1631
"Published in May 1994, written by K. Egevang and P. Francis, RFC-1631 defined NAT as one means to ease the growth rate of IPv4 address use. But the authors were worried about the impact of this technology. Several places in the document they pointed out the need to experiment and see what applications may be adversely affected by NAT's header manipulations, even before there was any significant operational experience. This is further evidenced in a quote from the conclusions: 'NAT has several negative characteristics that make it inappropriate as a long term solution, and may make it inappropriate even as a short term solution.'"

Tony Hain, "Architectural Implications of NATs", RFC 2993

- Firstly, NATs cannot support applications where the initiator lies on the 'outside'. The external device has no idea of the address of the local internal device, and therefore cannot direct any packets to that device in order to initiate a session. This implies that peer-to-peer services, such a voice, cannot work unaltered in a NAT environment.

- The workaround to this form of shortcoming is to force an altered deployment architecture, where service platforms used by external entities are placed 'beside' the NAT, allowing command and control from the interior of the local network, and having a permanent (non-NAT) interface to the external network. Obviously this implies some further centralization of IT services within the NATted entity.
- Even this approach does not work well for applications such as Voice over IP, where the 'server' now needs to operate as some form of proxy agent. The generic approach here for applications to traverse NATs in the 'wrong' direction is for the inside device to forge a UDP connection to the outside agent, and for the inside device to then establish the NAT translated address that has been used, and then re-publish this address as its service point. Sounds fragile? Unfortunately, yes. The other approach is to shift the application to use a set of end point identifiers that are distinct from IP addresses, and use a distributed set of 'agents' and ' helpers' to dynamically translate the application-level identifiers into transport IP addresses as required. This tends to create added complexity in application deployment, and embarks on a path of interdependency which is less than desirable. In summary, workarounds to re-establish a peer-to-peer networking model with NATs tend to be limited, complex and often fragile.
- Robust security in IP environments typically operates on an end-to-end model, where both ends include additional information in the packet that can detect attempts to alter the packet in various ways. In IPSEC the header part of the packet is protected by the Authentication header, where an encrypted signature of certain packet header fields is included in the IPSEC packet. If the packet header is changed in transit in unexpected ways the signature check will fail. Obviously IPSEC attempts to protect the packet's address fields - the very same fields that NATs alter! This leads to the observation that robust security measures and NATs don't mix very well. NATs inhibit implementation of security at the IP level.
- NATs have no inherent failover - NATs are an active in-band mechanism that cannot fail into a safe operating fallback mode. When a NAT goes offline all traffic through the NAT stops. NATs create a single point where fates are shared, in the device maintaining connection state and dynamic mapping information.
- NATs sit on the data path and attempt to process every packet. Obviously bandwidth scaling requires NAT scaling.
- NATs are not backed up by industry standardized behavior. While certain NAT-traversal applications make assumptions about the way NATs behave, it is not the case that all NATs will necessarily behave in precisely the same way. Applications that work in one context may not necessarily operate in others.
- Multiple NATs can get very confusing with 'inside' and 'outside' concepts when NATs are configured in arbitrary ways. NATs are best deployed in a strict deployment model of an "inside" being a stub private network and an "outside" of the public Internet. Forms of multiple interconnects, potential loops and other forms of network transit with intervening NATs lead to very strange failure modes that are at best highly frustrating.
- With NATs there is no clear, coherent and stable concept of network identity. From the outside these NAT-filtered interior devices are only visible as transient entities.
- Policy based mechanisms that are based on network identity (e.g. Policy QoS) cannot work through NATS
- Normal forms of IP mobility are broken when any element behind the NAT attempts to roam beyond its local private domain. Solutions are possible to this, generally involving specific NAT-related alterations to the behavior of the Home Agent and the mobile device.
- Applications that work with identified devices, or that actually identify devices (such as SNMP and DNS) require very careful configuration when operating an a NAT environment.
- NATs complicate the use of multi-homing by a site in order to increase the reliability of their Internet connectivity. In general NATs and multi-homing are a very complex configuration.
- More generically, NATs break the flexible end-to-end model of the Internet.

So can NATs scale into larger, broader more versatile networks? The answer is no, unfortunately not.

NATs are a short term expediency that is currently turning into a longer term set of overriding constraints placed on the further evolution of the Internet. Not only do new applications need to include considerations of NAT traversal, but we appear to be entering into a situation where if an application cannot work across NATs than the application itself fails to gain acceptance. We seem to be locking into a world that is almost the antithesis of the Internet concept. In this NAT-based world servers reside within the network and are operated as part of the service-provider's role, while end devices are seen as 'dumb' clients, who can establish connections to servers but cannot establish connections between each other. The widespread use of NATs appear to be reinforcing a reemergence of the model of 'smart network, dumb clients'.

> Now, six years later and in spite of the prediction, the use of NATs is becoming widespread in the Internet. Some people are proclaiming NAT as both the short and long term solution to some of the Internet's address availability issues and questioning the need to continue the development of IPv6. The claim is sometimes made that NAT 'just works' with no serious effects except on a few legacy applications. At the same time others see a myriad of difficulties caused by the increasing use of NAT.
> The arguments pro & con frequently take on religious tones, with each side passionate about its position.
>
> - Proponents bring enthusiasm and frequently cite the most popular applications of Mail & Web services as shining examples of NAT transparency. They will also point out that NAT is the feature that finally breaks the semantic overload of the IP address as both a locator and the global endpoint identifier (EID).
> - An opposing view of NAT is that of a malicious technology, a weed which is destined to choke out continued Internet development. While recognizing there are perceived address shortages, the opponents of NAT view it as operationally inadequate at best, bordering on a sham as an Internet access solution. Reality lies somewhere in between these extreme viewpoints.
>
> Tony Hain, "Architectural Implications of NAT", RFC 2993

So what's the answer?

If we accept NATs as a fact of the Internet the set of potential applications and related services that fail to work with NATs grows larger over time. The Internet will be locked into a client / server model of interaction and will loose the true potential of a massive system of diverse edge systems capable of interacting in ways that are constantly evolving.

And if we deliberately phase out NATs then we will need an address space that is appreciably larger than IPv4. In this approach IPv6 looks like our only current option.

In considering NATs it seems that we are back to the very basics of networking. The basic requirements of any network are "who", "where" and " how", or "identity", "location" and "forwarding". In the case of IP all these elements were included in the semantics of an IP address. Maybe, just maybe, as we look at the longer term developments of IP technology, one potential refinement may be the separation of end point identity to that of location.

If we ever venture down such a path I trust that such as move towards the use of explicit identities does not generate a complementary deployment of Network Identity Translators, or NITs, to complement a continued widespread use NATs!

Disclaimer

The above views do not represent the views of the Internet Society, nor do they represent the views of the author's employer, the Telstra Corporation. They were possibly the opinions of the author at the time of writing this article, but things always change, including the author's opinions!

About the Author

GEOFF HUSTON holds a B.Sc. and a M.Sc. from the Australian National University. He has been closely involved with the development of the Internet for the past decade, particularly within Australia, where he was responsible for the initial build of the Internet within the Australian academic and research sector. Huston is currently the Chief Scientist in the Internet area for Telstra. He is also a member of the Internet Architecture Board, and is the Secretary of the APNIC Executive Committee. He was an inaugural Trustee of the Internet Society, and served as Secretary of the Board of Trustees from 1993 until 2001, with a term of service as chair of the Board of Trustees in 1999 – 2000. He is author of *The ISP Survival Guide*, ISBN 0-471-31499-4, *Internet Performance Survival Guide: QoS Strategies for Multiservice Networks*, ISBN 0471-378089, and coauthor of *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*, ISBN 0-471-24358-2, a collaboration with Paul Ferguson. All three books are published by John Wiley & Sons.
E-mail: gih@telstra.net