Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 17, 2009

D. Thaler
Microsoft

January 13, 2009

Exit Strategies for IPv6 UNSAF Mechanisms
draft-thaler-ipv6-saf-00.txt

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on July 17, 2009.

Copyright Notice

Abstract

RFC 3424 requires an exit strategy for UNilateral Self-Address Fixing (UNSAF) mechanisms which are required for applications to work with

most translation schemes. There are various recent proposals that would result in translation becoming permanent. This document discusses a strategy for avoiding UNSAF mechansisms becoming permanent in this case.

Table of Contents

1. Introduction

Many applications and protocols use one or more addresses of the local machine, e.g. to send in an application protocol exchange or to advertise a public address at which it will accept connections.

RFC 2993 [RFC2993] discusses architectural implications of Network Address Translation (NAT). One of the implications of translation is that in general the address that must be used by other nodes to reach a destination is not the address assigned to an interface on the destination, where the destinations applications and protocols would naturally find it. As a result, NAT generally requires a mechanism whereby an endpoint can determine the address by which it is known to other endpoints, and then fix its own messages to use that address instead of the one(s) it would normally use. This category of mechanisms is known as UNilateral Self-Address Fixing (UNSAF).

RFC 3424 [RFC3424] discusses architectural implications of UNSAF mechanisms, and concludes that they are only appropriate as short term fixes and recommends that any UNSAF proposal require, among other things, an exit strategy. Since NAT mechanisms generally require UNSAF mechanisms, an exit strategy for an UNSAF proposal often requires an exit strategy for the NAT mechanism motivating it.

2. IPv6 Translation

The notion of IPv4-IPv6 translation (e.g., NAT-PT [RFC2766]) first introduced the NAT problems into IPv6 and motivated UNSAF mechanisms in IPv6. Although NAT-PT was deprecated ([RFC4966]), the notion of IPv4-IPv6 translation has become even more important. There is a fairly clear exit strategy (although the timeframe of it is not at all clear), which is that IPv4-IPv6 translation use decreases as IPv4-only nodes decrease over time. As a result, the exit strategy of any resulting UNSAF mechanisms is that their use declines as IPv4-IPv6 translation declines.

Recently however there has been discussion of the possibility of IPv6-IPv6 translation (e.g., NAT66 [I-D.mrw-behave-nat66] to address renumbering pains, Six/One [I-D.vogt-rrg-six-one] to address routing scalability, etc.). Such proposals, if adopted, are not proposed as short term mechanisms but rather as more permanent changes to the architecture. As such, if UNSAF mechanisms are required, the exit strategy cannot be simply based on declining IPv6-IPv6 translation.

3. IPv6 Translation Without UNSAF

In this section, we focus primarily on IPv6-IPv6 translation, although there may be cases where the same concepts might be applicable to IPv4-IPv6 translation or IPv4-IPv4 translation.

While translation in general requires UNSAF mechanisms, some uses of translation do not. Recall that UNSAF mechanisms are needed whenever the address reachable by outside parties is not an address of the local machine. Hence any use of translation whereby the address reachable by outside parties is still an address that appears to be assigned to some interface on the machine, does not require UNSAF mechanisms. For example, the Host Identity Protocol (HIP) [RFC5201] uses translation in this respect. The address seen by applications is in fact not the address used on the wire, but is translated by the HIP layer on both the sender and the receiver.

There are two key requirements for the translation mechanism:

1. The translation is reversible without loss of information, and
2. The address is presented by the host to upper layers in the same way as a normal IP address

When these requirements are met, reversible translation can be compared to (and contrasted with) a tunnel with header compression. To reverse translation, both translators must have the information necessary to perform the translation, which requires some configuration or per-host signaling mechanism (e.g., DHCP, as opposed to per-flow as HIP does) for learning an address to configure on an interface, which obviates the need for applications to use an UNSAF mechanism above the transport layer. We will refer to this concept as Self-Address Finding (SAF) to distinguish it from UNSAF mechanisms. Note that "finding" is intentionally used here instead of "fixing" as in UNSAF; since the address found is actually used by IP and higher layers, there is nothing to "fix" up higher.

Tunneling mechanisms, however, have incentive issues (as pointed out in [RFC5218]) in that they require both ends to be changed before either end benefits. Translation mechanisms such as NAT, on the other hand, have the advantage of being unilaterally deployable, at the expense of breaking some applications.

Reversible IPv6-IPv6 translation can be initially deployed unilaterally (at the expense of breaking some applications) at a translation middlebox without touching end hosts, avoiding the incentive issues with tunneling. End-to-end connectivity can then be restored once the host is able to learn the external address and configure it on a virtual interface; hence, there is further incentive built-in which restores the end-to-end model. This

provides an exit strategy that does not require an UNSAF mechanism or result in the issues discussed in [RFC3424].

3.1. Evaluation of Architectural Issues

Regarding issues with NAT mechanisms raised in [RFC2993]:

- Per-flow state in the middlebox (scaling, multihoming, single point-of-failure, etc): Reversible translation can be done without any per-flow state in the middlebox. NAT66 and Six/One are examples of this.
- Inhibit IPsec: If translation and reversing can be done below IPsec, IPsec works normally. (Or if translation and reversing is done within IPsec as HIP does, IPsec also works.)
- Address sharing (NAPT) inhibits other transport protocols: Reversible translation can be done without address sharing, allowing arbitrary transport protocols to work.

Regarding issues with UNSAF mechanisms raised in [RFC3424]:

- No unique outside: When nested translators exist, there are multiple outside areas and hence multiple addresses by which one is reachable by different peers. Reversible translation does not change this. This means that a node must be able to discover the address assigned by each translator in front of it.
- Circumventing firewalls: Firewalls are orthogonal to reversible translation. SAF mechanisms should not circumvent firewalls. Since translators can be stateless, there is no need for periodic messages that often keep holes open in firewalls.
- Timeout issues of address assignment in middlebox: Since translators can be stateless, there is no state to time out.
- Fate sharing when a server separate from the middlebox is used: Like UNSAF mechanisms, SAF mechanisms could either use a server separate from the middle box or communicate directly with the middlebox itself. Communicating with a server on the Internet, without any support from the translator, generally only allows discovering the address assigned by the outermost translator (i.e., the address seen by the server outside), not each translator. Furthermore, communicating with a remote server results in depending on reachability all the way to that server, whereas the desired communication may be much closer and otherwise be possible even when the server is unreachable. Hence the use of an external server is not recommended for SAF mechanisms.

3.2. Requirements for SAF Mechanisms

From the above discussion, we obtain the following requirements for SAF mechanisms.

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].

1. Discovery: A SAF mechanism MUST allow a node to find the addresses assigned by all translators it is behind.
2. Fate-Sharing: A SAF mechanism SHOULD allow a node discover the addresses assigned by translators even when the network behind them is currently unreachable.
3. Staleness: A SAF mechanism MUST allow a node to know when to stop using the address (e.g., if the assigned address changes due to an ISP change). That is, a SAF proposal MUST specify what a node uses as the ValidLifetime and the PreferredLifetime of an address found.
4. Multihoming: A SAF mechanism MUST support a node being connected to a network with multiple equivalent translators, meaning that the same translation would be done regardless of the path taken. In other words, it MUST NOT assume that it gets a unique address from every translator. This is not a requirement that there be such translators (e.g., egress routers on opposite sides of a continent are not necessarily expected to translate to the same prefix, only that if two translators are configured to translate to the same prefix, then the SAF mechanism should support this).
5. Privacy: A SAF mechanism SHOULD support temporary addresses [RFC3041] in addition to public addresses.
6. Security: A SAF mechanism SHOULD support Cryptographically Generated Addresses (CGAs) [RFC3972].

4. Security Considerations

NATs and UNSAF mechanisms generally interfere with security mechanisms because they change the addresses and/or content of messages exchanged. This document discusses requirements for SAF mechanisms that avoid these issues.

5. IANA Considerations

[RFC Editor: please remove this section prior to publication.]

This document has no IANA Actions.

6. References

6.1. Normative References

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

[RFC3041] Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

6.2. Informative References

[I-D.mrw-behave-nat66] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Address Translation (NAT66)", draft-mrw-behave-nat66-01 (work in progress), November 2008.

[I-D.vogt-rrg-six-one] Vogt, C., "Six/One: A Solution for Routing and Addressing in IPv6", draft-vogt-rrg-six-one-01 (work in progress), November 2007.

[RFC2766] Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February 2000.

[RFC2993] Hain, T., "Architectural Implications of NAT", RFC 2993, November 2000.

[RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

[RFC4966] Aoun, C. and E. Davies, "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status", RFC 4966, July 2007.

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

[RFC5218] Thaler, D. and B. Aboba, "What Makes For a Successful Protocol?", RFC 5218, July 2008.

Author's Address

Dave Thaler
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

Phone: +1 425 703 8835
Email: dthaler@microsoft.com