Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 23, 2013

W. Kumari
Google
I. Gashinsky
Yahoo!
J. Jaeggli
Zynga
K. Chittimaneni
Google

September 19, 2012

Neighbor Discovery Enhancement for DOS mititgation
draft-gashinsky-6man-v6nd-enhance-01

Abstract

In IPv4, subnets are generally small, made just large enough to cover the actual number of machines on the subnet. In contrast, the default IPv6 subnet size is a /64, a number so large it covers trillions of addresses, the overwhelming number of which will be unassigned. Consequently, simplistic implementations of Neighbor Discovery can be vulnerable to denial of service attacks whereby they attempt to perform address resolution for large numbers of unassigned addresses. Such denial of attacks can be launched intentionally (by an attacker), or result from legitimate operational tools that scan networks for inventory and other purposes. As a result of these vulnerabilities, new devices may not be able to "join" a network, it may be impossible to establish new IPv6 flows, and existing IPv6 transported flows may be interrupted.

This document describes a modification to the [RFC4861] neighbor discovery protocol aimed at improving the resilience of the neighbor discovery process. We call this process Gratuitous neighbor discovery and it derives inspiration in part from analogous IPv4 gratuitous ARP implementation.

Status of this Memo

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 23, 2013.

Copyright Notice

Table of Contents

1. Introduction

This document describes modifications to the IPv6 Neighbor Discovery protocol [RFC4861] in order to reduce exposure to vulnerabilities when a network is scanned, either by an intruder, as part of a deliberate DOS attempt, or through the use of scanning tools that perform network inventory, security audits, etc. (e.g., "nmap"). In some cases, DOS-like conditions can also be induced by legitimate traffic in heavy traffic networks such as campuses or datacenters.

1.1. Applicability

This document is primarily intended for implementors of [RFC4861].

2. The Problem

In IPv4, subnets are generally small, made just large enough to cover the actual number of machines on the subnet. For example, an IPv4 /20 contains only 4096 address. In contrast, the default IPv6 subnet size is a /64, a number so large it covers literally billions of billions of addresses, the overwhelming number of which will be unassigned. Consequently, simplistic implementations of Neighbor Discovery can be vulnerable to denial of service attacks whereby they perform address resolution for large numbers of unassigned addresses. Such denial of attacks can be launched intentionally (by an attacker), or result from legitimate operational tools that scan networks for inventory and other purposes. As a result of these vulnerabilities, new devices may not be able to "join" a network, it may be impossible to establish new IPv6 flows, and existing IPv6 transport flows may be interrupted.

Network scans attempt to find and probe devices on a network. Typically, scans are performed on a range of target addresses, or all the addresses on a particular subnet. When such probes are directed via a router, and the target addresses are on a directly attached network, the router will to attempt to perform address resolution on a large number of destinations (i.e., some fraction of the 2^64 addresses on the subnet). The process of testing for the (non)existence of neighbors can induce a denial of service condition, where the number of Neighbor Discovery requests overwhelms the implementation's capacity to process them, exhausts available memory, replaces existing in-use mappings with incomplete entries that will never be completed, etc. The result can be network disruption, where existing traffic may be impacted, and devices that join the net find that address resolutions fails.

In some network environments, legitimate Neighbor Discovery traffic

from a large number of connected hosts could induce a DoS condition even without the use of any scanning tools. For e.g., Consider a campus network with a pair of core routers that aggregate traffic from a few thousand wifi clients. In this scenario, high volume of regular ND traffic from clients can easily overwhelm the routers such that they are no longer able to process regular traffic anymore.

In order to alleviate risk associated with this DOS threat, some router implementations have taken steps to rate-limit the processing rate of Neighbor Solicitations (NS). While these mitigations do help, they do not fully address the issue and may introduce their own set of potential liabilities to the neighbor discovery process.

This document is a companion to two additional documents. The first document was Operational Neighbor Discovery Problems [RFC6583] which addressed the problem in detail and described operational and implementation mitigation within the framework of the Existing protocol. The second related document Neighbor Unreachability Detection is too impatient [1] proposes to alter the Neighbor unreachability Detection by relaxing rules in an attempt to keep devices in the cache.

In this document we propose alterations that allow the update or installation of neighbor entries without the instigation of a full [RFC4861] neighbor solicitation.

3. Terminology

Address Resolution Address resolution is the process through which a node determines the link-layer address of a neighbor given only its IP address. In IPv6, address resolution is performed as part of Neighbor Discovery [RFC4861], p60

Forwarding Plane That part of a router responsible for forwarding packets. In higher-end routers, the forwarding plane is typically implemented in specialized hardware optimized for performance. Forwarding steps include determining the correct outgoing interface for a packet, decrementing its Time To Live (TTL), verifying and updating the checksum, placing the correct link-layer header on the packet, and forwarding it.

Control Plane That part of the router implementation that maintains the data structures that determine where packets should be forwarded. The control plane is typically implemented as a "slower" software process running on a general purpose processor and is responsible for such functions as the routing protocols, performing management and resolving the correct link-layer address

for adjacent neighbors. The control plane "controls" the forwarding plane by programming it with the information needed for packet forwarding.

Neighbor Cache As described in [RFC4861], the data structure that holds the cache of (amongst other things) IP address to link-layer address mappings for connected nodes. The forwarding plane accesses the Neighbor Cache on every forwarded packet. Thus it is usually implemented in an ASIC .

Neighbor Discovery Process The Neighbor Discovery Process (NDP) is that part of the control plane that implements the Neighbor Discovery protocol. NDP is responsible for performing address resolution and maintaining the Neighbor Cache. When forwarding packets, the forwarding plane accesses entries within the Neighbor Cache. Whenever the forwarding plane processes a packet for which the corresponding Neighbor Cache Entry is missing or incomplete, it notifies NDP to take appropriate action (typically via a shared queue). NDP picks up requests from the shared queue and performs any necessary actions. In many implementations it is also responsible for responding to router solicitation messages, Neighbor Unreachability Detection (NUD), etc.

4. Background

Modern router architectures separate the forwarding of packets (forwarding plane) from the decisions needed to decide where the packets should go (control plane). In order to deal with the high number of packets per second the forwarding plane is generally implemented in hardware and is highly optimized for the task of forwarding packets. In contrast, the NDP control plane is mostly implemented in software processes running on a general purpose processor.

When a router needs to forward an IP packet, the forwarding plane logic performs the longest match lookup to determine where to send the packet and what outgoing interface to use. To deliver the packet to an adjacent node, It encapsulates the packet in a link-layer frame (which contains a header with the link-layer destination address). The forwarding plane logic checks the Neighbor Cache to see if it already has a suitable link-layer destination, and if not, places the request for the required information into a queue, and signals the control plane (i.e., NDP) that it needs the link-layer address resolved.

In order to protect NDP specifically and the control plane generally from being overwhelmed with these requests, appropriate steps must be

taken. For example, the size and rate of the queue might be limited. NDP running in the control plane of the router dequeues requests and performs the address resolution function (by performing a neighbor solicitation and listening for a neighbor advertisement). This process is usually also responsible for other activities needed to maintain link-layer information, such as Neighbor Unreachability Detection (NUD).

An attacker sending the appropriate packets to addresses on a given subnet can cause the router to queue attempts to resolve so many addresses that it crowds out attempts to resolve "legitimate" addresses (and in many cases becomes unable to perform maintenance of existing entries in the neighbor cache, and unable to answer Neighbor Solicitiation). This condition can result the inability to resolve new neighbors and loss of reachability to neighbors with existing ND-Cache entries. During testing it was concluded that 4 simultaneous nmap sessions from a low-end computer was sufficient to make a router's neighbor discovery process unhappy and therefore forwarding unusable.

This behavior has been observed across multiple platforms and implementations.

5. Neighbor Discovery Overview

When a packet arrives at (or is generated by) a router for a destination on an attached link, the router needs to determine the correct link-layer address to send the packet to. The router checks the Neighbor Cache for an existing Neighbor Cache Entry for the neighbor, and if none exists, invokes the address resolution portions of the IPv6 Neighbor Discovery [RFC4861] protocol to determine the link-layer address.

RFC4861 Section 5.2 (Conceptual Sending Algorithm) outlines how this process works. A very high level summary is that the device creates a new Neighbor Cache Entry for the neighbor, sets the state to INCOMPLETE, queues the packet and initiates the actual address resolution process. The device then sends out one or more Neighbor Solicitations, and when it receives a corresponding Neighbor Advertisement, completes the Neighbor Cache Entry and sends the queued packet.

6. NDP Protocol Gratuitous NA

RFC 4861, section 7.2.5 and 7.2.6 [RFC4861] requires that unsolicited neighbor advertisements result in the receiver setting it's neighbor

cache entry to STALE, kicking off the resolution of the neighbor using neighbor solicitation. If the link layer address in an unsolicited neighbor advertisement matches that of the existing ND cache entry, routers SHOULD retain the existing entry updating it's status with regards to LRU retention policy.

Hosts MAY be configured to send unsolicited Neighbor advertisement at a rate set at the discretion of the operators. The rate SHOULD be appropriate to the sizing of ND cache parameters and the host count on the subnet. An unsolicited NA rate parameter MUST NOT be enabled by default. The unsolicited rate interval as interpreted by hosts must jitter the value for the interval between transmissions. Hosts receiving a neighbor solicitation requests from a router following each of three subsequent gratuitous NA intervals MUST revert to RFC 4861 behavior.

Implementation of new behavior for unsolicited neighbor advertisement would make it possible under appropriate circumstances to greatly reduce the dependence on the neighbor solicitation process for retaining existing ND cache entries.

This may impact the detection of one-way reachability.

7. IANA Considerations

No IANA resources or consideration are requested in this draft.

8. Security Considerations

This technique has potential impact on neighbor detection and in particular the discovery of unidirectional forwarding problems.

9. Acknowledgements

The authors would like to thank Ron Bonica, Troy Bonin, John Jason Brzozowski, Randy Bush, Vint Cerf, Jason Fesler Erik Kline, Jared Mauch, Chris Morrow and Suran De Silva. Special thanks to Thomas Narten for detailed review and (even more so) for providing text!

Apologies for anyone we may have missed; it was not intentional.

10. References

10.1. Normative References

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

[RFC4398] Josefsson, S., "Storing Certificates in the Domain Name System (DNS)", RFC 4398, March 2006.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC6164] Kohno, M., Nitzan, B., Bush, R., Matsuzaki, Y., Colitti, L., and T. Narten, "Using 127-Bit IPv6 Prefixes on Inter-Router Links", RFC 6164, April 2011.

10.2. Informative References

[I-D.ietf-6man-impatient-nud] Nordmark, E. and I. Gashinsky, "Neighbor Unreachability Detection is too impatient", draft-ietf-6man-impatient-nud-02 (work in progress), July 2012.

[RFC4255] Schlyter, J. and W. Griffin, "Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints", RFC 4255, January 2006.

[RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, March 2012.

URIs

[1] <http://tools.ietf.org/html/draft-ietf-6man-impatient-nud-02>

Authors' Addresses

Warren Kumari
Google

Email: warren@kumari.net

Igor
Yahoo!
45 W 18th St
New York, NY
USA

Email: igor@yahoo-inc.com

Joel
Zynga
111 Evelyn
Sunnyvale, CA
USA

Email: jjaeggli@zynga.com

Kiran
Google
1600 Amphitheater Pkwy
Mountain View, CA
USA

Email: kk@google.com