Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 27, 2012
K. Naito
A. Matsumoto
NTT
March 26, 2012

NAT resource optimizing extension
draft-naito-nat-resource-optimizing-extension-01

Abstract

When network address translation (NAT) is used in an address resource restricted environment, or when a lot of users are located under a NAT device, IP addresses and port resources may be eaten up, and this affects user experiences very negatively. This situation can be greatly mitigated by tweaking mapping behavior and session timer handling in NAT functions. This document proposes two extensions for optimizing NAT IP address and port resources in address resource restricted environments. One extension enables simultaneous use of a NAT external port for different transport sessions, and the other makes use of a TCP timestamp for TIME_WAIT assassination.

Status of this Memo

Copyright Notice

1. Introduction

After IPv4 addresses run out, IPv4 address resources will be further restricted site-by-site. If global IPv4 address are shared between several clients, assignable port resources at each client will be limited.

NAT is a tool that is widely used to deal with this IPv4 address shortage problem. However, the demand for resources to provide Internet access to users and devices will continue to increase.IPv6 is a fundamental solution to this problem, but the deployment of IPv6 will take time.

In some cases, e.g. browsing a dynamic web page for a map service, a lot of sessions are used by the browser, and a number of ports are eaten up in a short time. What is worse is that when a NAT is between a PC and a server, TIME_WAIT state of each TCP connection is kept for certain period, typically for four minutes, which consumes port resources. Therefore, new connections cannot be established.

This problem is caused or worsened by the following behaviors.

1: In a lot of NAT implementations, a port that is available in NAT is allocated for a transport session.That is, a NAT does not use a port for multiple sessions simultaneously.

2: TIME_WAIT state assigned for a TCP connection remains active for 2MSL after the last ACK to the last FIN is transferred.

We propose two mechanisms to change the above behaviors that make it possible to save addresses and ports resources.

1.1. TCP TIME_WAIT

The TCP TIME_WAIT state is described in RFC793 [RFC0793]. The TCP TIME_WAIT state needs to be kept for 2MSL before a connection is CLOSED, for the reasons below.

1: In the event that packets from a session are delayed in the in-between network, and delivered to the end relatively later, we should prevent the packets from being transferred and interpreted as a packet that belongs to a new session.
2: If the remote TCP has not received the acknowledgment of its connection termination request, it will re-send the FIN packet several times.

These points are important for the TCP to work without problems.

1.2. TIME_WAIT Assassination

A TCP server MAY accept a TCP SYN for the 5-tuple session that is just finished and marked as TIME_WAIT state, as far as the TCP sequence number is increased. This is known as TIME-WAIT assassination. It should also be noted that some assassination hazards are described in RFC1337 [RFC1337].

1.3. Protect Against Wrapped Sequence numbers (PAWS)

The TCP sequence number wraps frequently especially in a high bandwidth session. PAWS is used to prevent old duplicate packets that occurred in a previous session from being transferred to the new session whose valid TCP sequence numbers happen to overlap with the old duplicate packets. This is implemented by introducing TCP timestamp option, and checking the timestamp option value of each packet. PAWS is described in RFC1323 [RFC1323].

2. NAT resource optimizing extension proposal

2.1. Port overloading mechanism

If destination addresses and ports are different at the outgoing sessions started by local clients, NAT MAY assign the same external port as the source ports at the sessions. Port overlapping mechanism

manages mappings between external packets and internal packets by looking at and storing the 5-tuple (protocol, source address, source port, destination address, destination port) of them. Thus, enables concurrent use of single port for multiple transport sessions, which enables NAT to work correctly in IP address resource limited network.

Discussions:

RFC4787 [RFC4787] and RFC5382 [RFC5382] requires "endpoint-independent mapping" at NAT, and port overlapping NAT cannot meet the requirement. This mechanism can degrade the transparency of NAT in that its mapping mechanism is endpoint-dependent and makes NAT traversal harder. However, if a NAT adopts endpoint-independent mapping together with endpoint-dependent filtering, then the actual behavior of the NAT will be the same as port overlapping NAT. It should also be noted that a lot of existing NAT devices adopted this port overlapping mechanism.

2.2. Apply RFC6191 to NAT

RFC 6191 [RFC6191] defines a mechanism for reducing the TIME_WAIT state using TCP timestamps. This document proposes to apply this RFC6191 mechanism at NAT. By tracing timestamp values in NAT that manages states of traversing TCP sessions, a TIME_WAIT remaining wait-time can be reduced to zero, when a TCP-SYN packet carrying a larger timestamp value arrives. In this case, PAWS (Protect Against Wrapped Sequence Numbers) works to discard old duplicate packets at NAT. A packet can be discarded as an old duplicate if it is received with a timestamp value less than a timestamp recently received on the connection. When there are several clients with nonsuccessive timestamp values are connected to a NAT device (i.e. not monotonically increasing among clients), it prevents some clients from getting a port to start a connection for a long time because other clients with larger timestamp values are preferred.Two workarounds for this issue are described below.

2.2.1. Rewrite timestamp values at NAT

Rewrite timestamp values of outgoings packets at NAT to be monotonically increasing.

2.2.2. Split an assignable number of port space to each client

Set some rules among clients connecting to NAT, e.g., split assignable ports between clients. This MAY be done by distributing rules to clients via NAT equipment.

3. Security Considerations

Security issues are not discussed in this memo.

4. Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC1323] Jacobson, V., Braden, B., and D. Borman, "TCP Extensions for High Performance", RFC 1323, May 1992.

[RFC1337] Braden, B., "TIME-WAIT Assassination Hazards in TCP", RFC 1337, May 1992.

[RFC4787] Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.

[RFC5382] Guha, S., Biswas, K., Ford, B., Sivakumar, S., and P. Srisuresh, "NAT Behavioral Requirements for TCP", BCP 142, RFC 5382, October 2008.

[RFC6191] Gont, F., "Reducing the TIME-WAIT State Using TCP Timestamps", BCP 159, RFC 6191, April 2011.

Authors' Addresses

Kengo Naito
NTT SI Lab
3-9-11 Midori-Cho
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 4949
Email: naito.kengo@lab.ntt.co.jp

Arifumi Matsumoto
NTT SI Lab
3-9-11 Midori-Cho
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 3334
Email: arifumi@nttv6.net