TCP Maintenance and Minor
Extensions (tcpm)
Internet-Draft
Intended status: BCP
Expires: December 24, 2010

F. Gont
UK CPNI
A. Oppermann
FreeBSD
June 22, 2010

On the generation of TCP timestamps
draft-gont-timestamps-generation-00.txt

Abstract

This document discusses the generation of TCP timestamps. In particular, it discusses a number of algorithms for producing monotonically-increasing timestamps such that timestamps can be used to reduce the TIME-WAIT state, and an algorithm for generating timestamps that allows for extended SYN-cookies.

Table of Contents

1. Introduction

The Timestamps option, specified in RFC 1323 [RFC1323], allows a TCP to include a timestamp value in its segments, that can be used used to perform two functions: Round-Trip Time Measurement (RTTM), and Protection Against Wrapped Sequences (PAWS).

For the purpose of PAWS, the timestamps sent on a connection are required to be monotonically increasing. RFC 1323 does not include any further requirements for the TCP timestamps (such as the initial timestamp value or the relationship between timestamps that correspond to different connections).

[I-D.gont-tcpm-tcp-timestamps] discusses an algorithm that employs TCP timestamps to reduce the TIME-WAIT state. The aforementioned algorithm benefits from timestamps that are monotonically-increasing across connections.

Some TCP implementations have employed TCP timestamps to implement extended SYN-Cookies [RFC4987]. These implementations encode part of the information received in an incomming SYN segment in the TCP timestamps sent in the SYN/ACK.

It should be noted that a TCP implementation could benefit from the benefits of both timestamps generation approaches. Monotonically-increasing timestamps could be generated for TCPs that perform the active open, while timestamps for TCPs that perform the passive open could be generated according to [Opperman].

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. Timestamps generation for TCPs that perform the active open

While there is no requirement that timestamps are monotonically increasing across TCP connections, the generation of timestamps such that they are monotonically increasing across connections between the same two endpoints allows the use of timestamps for improving the handling of SYN segments that are received while the corresponding four-tuple is in the TIME-WAIT state. That is, the timestamp option could be used to perform heuristics to determine whether to allow the creation of a new incarnation of a connection that is in the TIME-WAIT state.

It is RECOMMENDED that timestamps are generated with a similar algorithm to that introduced by RFC 1948 [RFC1948] for the generation

of Initial Sequence Numbers (ISNs). That is,

timestamp = T() + F(localhost, localport, remotehost, remoteport, secret_key)

where the result of T() is a global system clock that complies with the requirements of Section 4.2.2 of RFC 1323 [RFC1323], and F() is a function that should not be computable from the outside without knowledge of the secret key (secret_key). Therefore, we suggest F() to be a cryptographic hash function of the connection-id and some secret data (which could be chosen randomly).

F() provides an offset that will be the same for all incarnations of a connection between the same two endpoints, while T() provides the monotonically increasing values that are needed for PAWS.

3. Timestamps generation for TCPs that perform the passive open

3.1. Extended SYN cookies

The purpose of SYN cookies is to avoid keeping track of all SYN's we receive and to be able to handle SYN floods from bogus source addresses (where we will never receive any reply). SYN floods try to exhaust all our memory and available slots in the SYN cache table to cause a denial of service to legitimate users of the local host.

The idea of SYN cookies is to encode and include all necessary information about the connection setup state within the SYN-ACK we

send back and thus to get along without keeping any local state until the ACK to the SYN-ACK arrives (if ever). Everything we need to know should be available from the information we encoded in the SYN-ACK.

This implementation extends the orginal idea and first implementation of FreeBSD by using not only the initial sequence number field to store information but also the timestamp field if present. This way we can keep track of the entire state we need to know to recreate the session in its original form. Almost all TCP speakers implement RFC1323 timestamps these days. For those that do not we still have to live with the known shortcomings of the ISN-only SYN cookies.

Initial Sequence Number (ISN) we send:

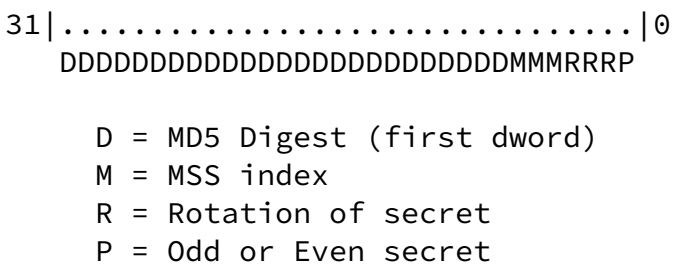

Figure 1

The MD5 Digest is computed with over following parameters:

- randomly rotated secret
- struct in_conninfo containing the remote/local ip/port (IPv4&IPv6)
- the received initial sequence number from remote host
- the rotation offset and odd/even bit

Timestamp we send:

```
   31|..............................|0
      DDDDDDDDDDDDDDDDDDDDDDSSSSRRRRA5

D = MD5 Digest (third dword) (only as filler)
S = Requested send window scale
R = Requested receive window scale
A = SACK allowed
5 = TCP-MD5 enabled (not implemented yet)

XORed with MD5 Digest (forth dword)
```

Figure 2

The timestamp isn't cryptographically secure and doesn't need to be. The double use of the MD5 digest dwords ties it to a specific remote/ local host/port, remote initial sequence number and our local time limited secret. A received timestamp is reverted (XORed) and then the contained MD5 dword is compared to the computed one to ensure the timestamp belongs to the SYN-ACK we sent. The other parameters may have been tampered with but this isn't different from supplying bogus values in the SYN in the first place.

3.2. Potential problems with SYN-cookies

Some of the problems of ISN-only SYN cookies remain, nevertheless. Consider the problem of a recreated (and retransmitted) cookie. If

the original SYN was accepted, the connection is established. The second SYN is inflight, and if it arrives with an ISN that falls within the receive window, the connection is killed.

A heuristic to determine when to accept syn cookies is not necessary. An ACK flood would cause the syncookie verification to be attempted, but a SYN flood causes syncookies to be generated. Both are of equal cost, so there's no point in trying to optimize the ACK flood case.

Also, if you don't process certain ACKs for some reason, then all someone would have to do is launch a SYN and ACK flood at the same time, which would stop cookie verification and defeat the entire purpose of syncookies.

4. Security Considerations

This document describes a number of algorithms for generating TCP timestamps.

[CPNI-TCP] provides a thorough discussion of the security implications of TCP timestamps.

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

Fernando Gont would like to thank the United Kingdom's Centre for the Protection of National Infrastructure (UK CPNI) for their continued support.

7. References

7.1. Normative References

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

[RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

[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.

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

7.2. Informative References

[CPNI-TCP]
CPNI, "Security Assessment of the Transmission Control Protocol (TCP)", http://www.cpni.gov.uk/Docs/tn-03-09-security-assessment-TCP.pdf, 2009.

[FreeBSD-impl]
FreeBSD, "FreeBSD", http://www.freebsd.org/cgi/cvsweb.cgi/src/sys/netinet/tcp_syncache.c?rev=1.99&content-type=text/x-cvsweb-markup, 2008.

[I-D.gont-tcpm-tcp-timestamps]
Gont, F., "Reducing the TIME-WAIT state using TCP timestamps", draft-gont-tcpm-tcp-timestamps-04 (work in progress), March 2010.

[Linux] The Linux Project, "http://www.kernel.org".

[Linux-impl]
Westphal, F., "Add support for TCP-options via timestamps", http://lwn.net/Articles/277219/, 2008.

[Opperman]
Oppermann, A., "FYI: Extended TCP syncookies in FreeBSD-current", Post to the tcpm mailing-list. Available at: http://www.ietf.org/mail-archive/web/tcpm/current/msg02251.html, 2006.

[RFC1948] Bellovin, S., "Defending Against Sequence Number Attacks", RFC 1948, May 1996.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, August 2007.

Authors' Addresses

Fernando Gont
UK Centre for the Protection of National Infrastructure

Email: fernando@gont.com.ar
URI: http://www.cpni.gov.uk

Andre Oppermann
FreeBSD

Email: andre@freebsd.org