TCP Maintenance and Minor
Extensions (tcpm)
Internet-Draft
Obsoletes: 1948 (if approved)
Updates: 793 (if approved)
Intended status: Standards Track
Expires: July 7, 2011

F. Gont
UTN/FRH
S. Bellovin
Columbia University

January 3, 2011

Defending Against Sequence Number Attacks
draft-gont-tcpm-rfc1948bis-00.txt

Abstract

This document specifies an algorithm for the generation of TCP Initial Sequence Numbers (ISNs), such that the chances of an off-path attacker of guessing the sequence numbers in use by a target connection are reduced. This document is a revision of RFC 1948, and takes the ISN generation algorithm originally proposed in that document to Standards Track.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on July 7, 2011.

Table of Contents

1. Introduction

During the last 25 years, the Internet has experienced a number of off-path attacks against TCP connections. These attacks have ranged from trust relationships exploitation to Denial of Service attacks [CPNI-TCP]. Discusion of some of these attacks dates back to at least 1985, when Morris [Morris1985] described a form of attack based on guessing what sequence numbers TCP [RFC0793] will use for new connections.

In 1996, RFC 1948 [RFC1948] proposed an algorithm for the selection of TCP Initial Sequence Numbers (ISNs), such that the chances of an off-path attacker of guessing valid sequence numbers are reduced. With the aforementioned algorithm, such attacks would remain possible if and only if the Bad Guy already had the ability to launch even more devastating attacks.

This document is a revision of RFC 1948, and takes the ISN generation algorithm originally proposed in that document to Standards Track.

Section 2 provides a brief discussion of the requirements for a good ISN generation algorithm. Section 3 specifies a good ISN randomization algorithm. Finally, Appendix A provides a discussion of the trust-relationship exploitation attacks that originally motivated the publication of RFC 1948 [RFC1948].

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. Generation of Initial Sequence Numbers

RFC 793 [RFC0793] suggests that the choice of the Initial Sequence Number of a connection is not arbitrary, but aims to reduce the chances of a stale segment from being accepted by a new incarnation of a previous connection. RFC 793 [RFC0793] suggests the use of a global 32-bit ISN generator that is incremented by 1 roughly every 4 microseconds.

It is interesting to note that, as a matter of fact, protection against stale segments from a previous incarnation of the connection is enforced by preventing the creation of a new incarnation of a previous connection before 2*MSL have passed since a segment corresponding to the old incarnation was last seen. This is accomplished by the TIME-WAIT state, and TCP's "quiet time" concept (see Appendix B of [RFC1323]).

Based on the assumption that ISNs are monotonically-increasing across connections, many stacks (e.g., 4.2BSD-derived) use the ISN of an incomming SYN segment to perform "heuristics" that enable the creation of a new incarnation of a connection while the previous incarnation is still in the TIME-WAIT state (see pp. 945 of [Wright1994]). This avoids an interoperability problem that may arise when a systems establishes connections to a specific TCP end-point at a high rate [Silbersack2005].

Unfortunately, the ISN generator described in [RFC0793] makes it trivial for an off-path attacker to predict the ISN that a TCP will use for new connections, thus allowing a variety of attacks against TCP connections [CPNI-TCP]. One of the possible attacks that took advantage of weak sequence numbers was first described in [Morris1985], and its exploitation was widely publicized about 10 years later [Shimomura1995]. [CERT2001] and [USCERT2001] are advisories about the security implications of weak ISN generators. [Zalewski2001] and [Zalewski2002] contain a detailed analysis of ISN generators, and a survey of the algorithms in use by popular TCP implementations.

Simple randomization of the TCP Initial Sequence Numbers would mitigate those attacks that require an attacker to guess valid sequence numbers. However, it would also break the 4.4BSD "heuristics" to accept a new incoming connection when there is a previous incarnation of that connection in the TIME-WAIT state [Silbersack2005].

We can prevent sequence number guessing attacks by giving each connection -- that is, each 4-tuple of (localip, localport, remoteip, remoteport) -- a separate sequence number space. Within each space, the initial sequence number is incremented according to [RFC0793]; however, there is no obvious relationship between the numbering in different spaces.

The obvious way to do this is to maintain state for dead connections, and the easiest way to do that is to change the TCP state transition diagram so that both ends of all connections go to TIME-WAIT state. That would work, but it's inelegant and consumes storage space. Instead, we propose an improvement to the TCP ISN generation algorithm.

3. Proposed Initial Sequence Number (ISN) generation algorithm

TCP SHOULD generate its Initial Sequence Numbers with the expression:

ISN = M + F(localip, localport, remoteip, remoteport)

where M is the 4 microsecond timer, and F is a pseudorandom function (PRF) of the connection-id. It is vital that F not be computable from the outside, or an attacker could still guess at sequence numbers from the initial sequence number used for some other connection. The PRF could be implemented as a cryptographic hash of the concatenation of the connection-id and some secret data; SHA-256 [FIPS-SHS] would be a good choice for the hash function. The secret data can either be a true random number [RFC4086], or it can be the combination of some per-host secret and the boot time of the machine. The boot time is included to ensure that the secret is changed on occasion.

Note that the secret cannot easily be changed on a live machine. Doing so would change the initial sequence numbers used for reincarnated connections; to maintain safety, either dead connection state must be kept or a quiet time observed for two maximum segment lifetimes after such a change.

4. Security Considerations

Good sequence numbers are not a replacement for cryptographic authentication, such as that provided by IPsec [RFC4301]. At best, they're a palliative measure.

If random numbers are used as the sole source of the secret, they MUST be chosen in accordance with the recommendations given in [RFC4086].

A security consideration that should be made about the algorithm proposed in this document is that it might allow an attacker to count the number of systems behind a Network Address Translator (NAT) [RFC3022]. Depending on the ISN generators implemented by each of the systems behind the NAT, an attacker might be able to count the number of systems behind a NAT by establishing a number of TCP connections (using the public address of the NAT) and indentifying the number of different sequence number "spaces". [I-D.gont-behave-nat-security] discusses how this and other information leakages at NATs could be mitigated.

An eavesdropper who can observe the initial messages for a connection can determine its sequence number state, and may still be able to launch sequence number guessing attacks by impersonating that connection. However, such an eavesdropper can also hijack existing connections [Joncheray1995], so the incremental threat isn't that high. Still, since the offset between a fake connection and a given

real connection will be more or less constant for the lifetime of the secret, it is important to ensure that attackers can never capture such packets. Typical attacks that could disclose them include both eavesdropping and the variety of routing attacks discussed in [Bellovin1989].

[CPNI-TCP] contains a discussion of all the currently-known attacks that require an attacker to know or be able to guess the TCP sequence numbers in use by the target connection.

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

Matt Blaze and Jim Ellis contributed some crucial ideas to RFC 1948, on which this document is based. Frank Kastenholz contributed constructive comments to that memo.

The authors of this document woul like to thank (in chronological order) Alfred Hoenes for providing valuable comments on earlier versions of this document.

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.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

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

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

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness

Requirements for Security", BCP 106, RFC 4086, June 2005.

7.2. Informative References

[Bellovin1989]
Morris, R., "Security Problems in the TCP/IP Protocol Suite", Computer Communications Review, vol. 19, no. 2, pp. 32-48, 1989.

[CERT2001]
CERT, "CERT Advisory CA-2001-09: Statistical Weaknesses in TCP/IP Initial Sequence Numbers", http://www.cert.org/advisories/CA-2001-09.html, 2001.

[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.

[FIPS-SHS]
FIPS, "Secure Hash Standard (SHS)", Federal Information Processing Standards Publication 180-3, 2008, available at: http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf.

[I-D.gont-behave-nat-security]
Gont, F. and P. Srisuresh, "Security implications of Network Address Translators (NATs)", draft-gont-behave-nat-security-03 (work in progress), October 2009.

[Joncheray1995]
Joncheray, L., "A Simple Active Attack Against TCP", Proc. Fifth Usenix UNIX Security Symposium, 1995.

[Morris1985]
Morris, R., "A Weakness in the 4.2BSD UNIX TCP/IP Software", CSTR 117, AT&T Bell Laboratories, Murray Hill, NJ, 1985.

[RFC0854] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, May 1983.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

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

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[RFC4251] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4954] Siemborski, R. and A. Melnikov, "SMTP Service Extension for Authentication", RFC 4954, July 2007.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.

[RFC5936] Lewis, E. and A. Hoenes, "DNS Zone Transfer Protocol (AXFR)", RFC 5936, June 2010.

[Shimomura1995] Shimomura, T., "Technical details of the attack described by Markoff in NYT", http://www.gont.com.ar/docs/post-shimomura-usenet.txt, Message posted in USENET's comp.security.misc newsgroup, Message-ID: <3g5gkl$5j1@ariel.sdsc.edu&gt, 1995.

[Silbersack2005] Silbersack, M., "Improving TCP/IP security through randomization without sacrificing interoperability.", EuroBSDCon 2005 Conference .

[USCERT2001] US-CERT, "US-CERT Vulnerability Note VU#498440: Multiple TCP/IP implementations may use statistically predictable initial sequence numbers", http://www.kb.cert.org/vuls/id/498440, 2001.

[Wright1994] Wright, G. and W. Stevens, "TCP/IP Illustrated, Volume 2: The Implementation", Addison-Wesley, 1994.

[Zalewski2001] Zalewski, M., "Strange Attractors and TCP/IP Sequence Number Analysis",

http://lcamtuf.coredump.cx/oldtcp/tcpseq.html, 2001.

[Zalewski2002]
Zalewski, M., "Strange Attractors and TCP/IP Sequence Number Analysis - One Year Later", http://lcamtuf.coredump.cx/newtcp/, 2002.

Appendix A. Address-based trust relationship exploitation attacks

This section discusses the trust-relationship exploitation attack that originally motivated the publication of RFC 1948 [RFC1948]. It should be noted that while RFC 1948 focused its discussion of address-based trust relationship exploitation attacks on Telnet [RFC0854] and the various UNIX "r" commands, both Telnet and the various "r" commands have since been largely replaced by secure counter-parts (such as SSH [RFC4251]) for the purpose of remote login and remote command execution. Nevertheless, address-based trust relationships are still employed nowadays in some scenarios. For example, some SMTP [RFC5321] deployments still authenticate their users by means of their IP addresses, even when more appropriate authentication mechanisms are available [RFC4954]. Another example is the authentication of DNS secondary servers [RFC1034] by means of their IP addresses for allowing DNS zone transfers [RFC5936], or any other access control mechanism based on IP addresses.

In 1985, Morris [Morris1985] described a form of attack based on guessing what sequence numbers TCP [RFC0793] will use for new connections. Briefly, the attacker gags a host trusted by the target, impersonates the IP address of the trusted host when talking to the target, and completes the 3-way handshake based on its guess at the next initial sequence number to be used. An ordinary connection to the target is used to gather sequence number state information. This entire sequence, coupled with address-based authentication, allows the attacker to execute commands on the target host.

Clearly, the proper solution for these attacks is cryptographic authentication [RFC4301] [RFC4120] [RFC4251].

The following subsections provide technical details for the trust relationship exploitation attack described by Morris [Morris1985].

A.1. Blind TCP connection-spoofing

In order to understand the particular case of sequence number guessing, one must look at the 3-way handshake used in the TCP open sequence [RFC0793]. Suppose client machine A wants to talk to rsh

server B. It sends the following message:

A->B: SYN, ISNa

That is, it sends a packet with the SYN ("synchronize sequence number") bit set and an initial sequence number ISNa.

B replies with

B->A: SYN, ISNb, ACK(ISNa)

In addition to sending its own initial sequence number, it acknowledges A's. Note that the actual numeric value ISNa must appear in the message.

A concludes the handshake by sending

A->B: ACK(ISNb)

RFC 793 [RFC0793] specifies that the 32-bit counter be incremented by 1 in the low-order position about every 4 microseconds. Instead, Berkeley-derived kernels traditionally incremented it by a constant every second, and by another constant for each new connection. Thus, if you opened a connection to a machine, you knew to a very high degree of confidence what sequence number it would use for its next connection. And therein lied the vulnerability.

The attacker X first opens a real connection to its target B -- say, to the mail port or the TCP echo port. This gives ISNb. It then impersonates A and sends

Ax->B: SYN, ISNx

where "Ax" denotes a packet sent by X pretending to be A.

B's response to X's original SYN (so to speak)

B->A: SYN, ISNb', ACK(ISNx)

goes to the legitimate A, about which more anon. X never sees that message but can still send

Ax->B: ACK(ISNb')

using the predicted value for ISNb'. If the guess is right -- and usually it will be, if the sequence numbers are weak -- B's rsh server thinks it has a legitimate connection with A, when in fact X is sending the packets. X can't see the output from this session,

but it can execute commands as more or less any user -- and in that case, the game is over and X has won.

There is a minor difficulty here. If A sees B's message, it will realize that B is acknowledging something it never sent, and will send a RST packet in response to tear down the connection. There are a variety of ways to prevent this; the easiest is to wait until the real A is down (possibly as a result of enemy action, of course). In actual practice, X can gag A by exploiting a very common implementation bug; this is described in the next subsection.

A.2. An old BSD bug

As mentioned in the previous sub-section, attackers performing a trust relationship exloitation attack may want to "gag" the trusted machine first. While a number of strategies are possible, most of the attacks detected in the wild relied on an implementation bug.

When SYN packets are received for a connection, the receiving system creates a new TCB in SYN-RCVD state. To avoid overconsumption of resources, 4.2BSD-derived systems permit only a limited number of TCBs in this state per connection. Once this limit is reached, future SYN packets for new connections are discarded; it is assumed that the client will retransmit them as needed.

When a packet is received, the first thing that must be done is a search for the TCB for that connection. If no TCB is found, the kernel searches for a "wild card" TCB used by servers to accept connections from all clients. Unfortunately, in many kernels this code was invoked for any incoming packets, not just for initial SYN packets. If the SYN-RCVD queue was full for the wildcard TCB, any new packets specifying just that host and port number were discarded, even if they weren't SYN packets.

To gag a host, then, the attacker sent a few dozen SYN packets to the rlogin port from different port numbers on some non-existent machine. This filled up the SYN-RCVD queue, while the SYN+ACK packets went off to the bit bucket. The attack on the target machine then appeared to come from the rlogin port on the trusted machine. The replies -- the SYN+ACKs from the target -- were perceived as packets belonging to a full queue, and were dropped silently. This could have been avoided if the full queue code checked for the ACK bit, which could not legally be on for legitimate open requests (if it was on, an RST should be sent in response).

Appendix B. Changes from previous versions of the document

B.1. Changes from RFC 1948

- New document aims at Standards Track (rather than Informaitonal).
- The discussion of address-based trust relationship attacks was updated and moved to an Appendix.
- The recommended hash algorithm has been changed to SHA-256 [FIPS-SHS], in response to the security concerns for MD5 [RFC1321].
- Formal requirements ([RFC2119]) are specified.

Authors' Addresses

Fernando Gont
Universidad Tecnologica Nacional / Facultad Regional Haedo
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fernando@gont.com.ar
URI: http://www.gont.com.ar

Steven M. Bellovin
Columbia University
1214 Amsterdam Avenue
MC 0401
New York, NY 10027
US

Phone: +1 212 939 7149
Email: bellovin@acm.org