Network Working Group   Tim Shepard
INTERNET DRAFT   July 12, 2004
Expires: January 13, 2005

   **By submitting this Internet-Draft, I certify that any applicable
   patent or other IPR claims of which I am aware have been disclosed,
   or will be disclosed, and any of which I become aware will be**
   disclosed, in accordance with RFC 3668.

                   Reassign Port Number option for TCP
             draft-shepard-tcp-reassign-port-number-00.txt

Status of this Memo

   This document is an Internet-Draft and is in full conformance with
   all provisions of Section 10 of RFC2026.

   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 a "work in progress."

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

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

Copyright Notice

   Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

   Most TCP connections are protected from spoofing attacks from off-
   path attackers by their obscurity. This memo suggests that the few
   TCP connections that aren't so protected today may be protected by
   making them obscure by using random values for both port numbers.
   The obvious difficulty with this approach is that the well-known port

   number is required on the initial SYN to connect to the desired
   service. A TCP option is proposed which can be used during the SYN
   and SYN-ACK exchange to request (and accomplish) reassignment of the
   well known port number to a random value.

Introduction

   Communication on the Internet today typically involves no end-to-end
   cryptography and is therefore secured only as much as the entire path
   of routers and links between the two communicating parties is
   secured. More robust methods of securing end-to-end communication
   involving cryptography have been developed but have only seen limited
   deployment. Most communication on the Internet today remains
   vulnerable to an attacker who is located somewhere along the path of
   communication.

   In the past few months a vulnerability of TCP connections to an off-
   path attacker has raised concern. An attacker who merely knows of or
   can surmise the existence of an active TCP connection can inject
   packets into the connection with forged source addresses (making
   packets arrive at one host with a source address which makes it
   appear to have come from the other host), disrupting communication in
   a variety of ways. These injected packets could come from distant
   parts of the Internet, and tracing the true origin of packets with
   forged source addresses can be difficult.

   Even if the attacker's knowledge of the existence of the connection
   is imperfect, the attacker may cover the unknown space rather quickly
   by injecting many packets covering all of the possibilities.

   To know completly of the existence of a TCP connection, an attacker
   would need to know the IP addresses of the two endpoints and the two
   16-bit TCP port numbers in use by the TCP connection. If this much
   is known to an attacker, then the only hurdle remaining for the
   attacker's forged packets is the sequence number acceptability test
   where the 32-bit sequence number in the TCP header of the arriving
   packet is tested to see if it is in-window.

   The RFC1323 window scale option, makes the fraction of TCP sequence
   space that is acceptable potentially quite large (as much as 1/4 of
   the 32-bit sequence number space).

   The RFC1323 timestamp option would seem to help, but packets without
   the RFC1323 timestamp option can still cause disruption---RSTs
   without a timestamp option generally need to be accepted and
   processed (e.g. in case a dialup user hangs up and a new dialup user
   sends RSTs in response to arriving TCP packets). Also note that no
   known implementations include a timestamp option on RST packets.

In most cases, TCP connections as implemented and used by
applications today are robust against spoofing attacks simply because
the attacker has no way of learning or knowing of the existence of
the TCP connection (identified by the two IP addresses and the two
16-bit port numbers). The vulnerability comes when the off-path
attacker can, for a long-lived TCP connection, obtain or surmise the
IP addresses and port numbers involved, or sufficiently narrow the
possibilities so that the search space is small enough to allow the
attacker to exhaustively enumerate the remaining possibilities. In
particular, this can happen if the IP addresses involved and one of
the port numbers are known, leaving only one 16-bit port number
unknown. In the case of some services listening on a well-known port
(such as BGP, SSH, or XMPP), this situation is typical.

This memo suggest that the way to secure TCPs from spoofed packets
sent by off-path attackers is:

     1. Don't reveal the port numbers, and

     2. Use good random values for both port numbers.

The difficulty with doing this today is that typically a well-known
port number is used to find the server, and this would be difficult
to change. The remainder of this memo describes a TCP option which
can be used to negotiate a reassignment of the well-known port number
to a randomly assigned value as part of the SYN/SYN-ACK handshake
that occurs at the beginning of the connection.

The Reassign Port Number option carried on the SYN

   A client wishing to have the server reassign the well-known port
   number to a random value may include a Reassign Port Number option on
   the segment that carries the SYN.

                    +---------+---------+
                    | Kind=99 |Length=2 |
                    +---------+---------+

   (note: 99 is here as a place holder until an IANA assignment)

   This option requests that the server reassign the port it will use
   for this connection to a random value. The Destination Port field in
   the TCP header itself will carry the normal well-known port number on
   which the server is expected to be listening.

   If the server does not implement this option, it will be ignored, and
   the TCP connection will continue without any reassignment of port

numbers.

The Reassign Port Number option carried on the SYN ACK

   If a server receives a SYN carrying the Reassign Port Number option,
   then it will (as usual) use the Destination Port field in the segment
   carrying the SYN to identify which listening socket it should be
   associated with, but when creating the established socket, it may (if
   it implements this option) reassign a random value to the local port
   number, and return a SYN ACK with the newly assigned port number, and
   include the Reassign Port Number option with the SYN ACK:

                   +---------+---------+---------+---------+
                   | Kind=99 |Length=4 | original port num |
                   +---------+---------+---------+---------+

   (note: 99 is here as a place holder until an IANA assignment)

   This SYN ACK will be received by the host that sent the SYN with the
   Reassign Port Number request, and will have a pair of port numbers in
   the TCP header which will not correspond to the port numbers in any
   connection block the host has. The option, however, will provide the
   client with the original port number of the server and allow the
   connection block to be found.   The client should process this
   option on a SYN ACK when it fails to find a corresponding connection
   block for TCP segment with both the SYN and ACK control bits turned
   on. If the ACK of the client's SYN is correct, then the connection
   block on the client can have the foreign port number overwritten with
   the Source Port number from the TCP header.

   Note that until the server receives an ACK of its SYN, and for some
   time after that, it will need to be prepared to process any received
   retransmissions of the original SYN which would still be directed at
   the listening port number, and the server MUST ensure that the
   processing of any such duplicate SYNs gets the same reassignment,
   even if the duplicate SYNs do not carry the Reassign Port Number
   option.

Steps a client may take to ensure robustness

   The client MAY, upon any retransmission of the SYN, try removing the
   Reassign Port Number option. A better strategy to cope with a broken
   path (e.g. containing a NAT or some other connection tracking
   middlebox that does not understand the remapping) would be for the
   client, upon failure to establish a connection with the Reassign Port
   Number option, would be to retry the connection from a new port
   number without the Reassign Port Number option.

   Implementors should carefully consider whether or not this option
   should be on by default for all TCP connections. Most TCP
   connections are already protected by their natural obscurity and/or
   short lifetime. Setting this option on by default in a general
   purpose operating system would probably cause many more failed
   connections due to the meddling of middleboxes than connections saved
   from spoofing attacks. In other words, there is potential to do more
   harm then good.

   Applications (such as BGP, JABBER, and perhaps SSH) may benefit from
   code which explicitly requests the use of this option and carefully
   manages the fallback to a connection attempt without this option.

Firewall and NAT considerations

   (More thought may be needed here.)

   Simple sorts of firewalls that disallow incoming TCP segments which
   do not have an ACK control bit set (to disallow the initiation of TCP
   connections from machines beyond the firewall) should not cause any
   problem for this scheme.

   Firewalls that track TCP connections, and NATs, will need to be aware
   of this option and track the connection to the new port number. Any
   such firewalls or NATs on the path that have not been upgraded to
   handle this option will likely cause connection attempts using this
   option to fail.

   Upgrading firewalls and NATs to track this option should be
   straightforward. (The simplest upgrade to a firewall that avoids
   blocking connection attempts carrying this option would be to simply
   strip this option from the TCP SYN segment, though that would leave
   such connections more vulnerable to spoofed packets.)

Security Considerations

   This memo describes a method of making it more difficult for an off-
   path attacker to slip a spoofed packet into a TCP connection by
   randomizing the port numbers involved (including what would normally
   be a well-known port number), and recommends that the port numbers
   not be revealed publicly (e.g. through any publicly-readable network
   management databases). This method provides only a limited
   enhancement of security, and does not come close to the robustness
   that can be achieved by techniques employing end-to-end encryption of
   the TCP segments (such as IPSEC).

IANA Considerations

   A single new TCP option value will need to be assigned for the
   Reassign Port Number option described above.

Author's Address:

   Tim Shepard
   122 Beech Street
   Belmont, MA 02478

   +1 617 489 7135
   shep@alum.mit.edu

Full Copyright Statement