



National Cyber Alert System

Technical Cyber Security Alert TA04-111A

[Archive](#)

Vulnerabilities in TCP

Original release date: April 20, 2004

Last revised: --

Source: US-CERT

Systems Affected

- Systems that rely on persistent TCP connections, for example routers supporting BGP

Overview

Most implementations of the Border Gateway Protocol (BGP) rely on the Transmission Control Protocol (TCP) to maintain persistent unauthenticated network sessions. There is a vulnerability in TCP which allows remote attackers to terminate network sessions. Sustained exploitation of this vulnerability could lead to a denial of service condition; in the case of BGP systems, portions of the Internet community may be affected. Routing operations would recover quickly after such attacks ended.

I. Description

In 2001, the CERT Coordination Center released [CA-2001-09](#), describing statistical weaknesses in various TCP/IP Initial Sequence generators. In that document, it was noted by Tim Newsham:

```
[I]f a sequence number within the receive window is known, an
attacker can inject data into the session stream or terminate
the connection. If the ISN value is known and the number of
bytes sent already sent is known, an attacker can send a
simple packet to inject data or kill the session. If these
values are not known exactly, but an attacker can guess a
suitable range of values, he can send out a number of packets
with different sequence numbers in the range until one is
accepted. The attacker need not send a packet for every
sequence number, but can send packets with sequence numbers a
window-size apart. If the appropriate range of sequence
numbers is covered, one of these packets will be accepted. The
total number of packets that needs to be sent is then given by
the range to be covered divided by the fraction of the window
size that is used as an increment.
```

Paul Watson has performed the statistical analysis of this attack when the ISN is not known and has pointed out that such an attack could be viable when specifically taking into account the TCP Window size. He has also created a proof-of-concept tool demonstrating the practicality of the attack. The National Infrastructure Security Co-Ordination Centre (NISCC) has published an advisory summarizing Paul Watson's analysis in "[NISCC Vulnerability Advisory 236929](#)."

Since TCP is an insecure protocol, it is possible to inject transport-layer packets into sessions between hosts given the right preconditions. The [TCP/IP Initial Sequence Number vulnerability \(VU#498440\)](#) referenced in CA-2001-09 is one example of how an attacker could inject TCP packets into a session. If an attacker were to send a Reset (RST) packet for example, they would cause the TCP session between two endpoints to terminate without any further communication.

The Border Gateway Protocol (BGP) is used to exchange routing information for the Internet and is primarily used by Internet Service Providers (ISPs). For detailed information about BGP and some tips for securing it, please see [Cisco System's documentation](#) or [Team Cymru](#). A vulnerable situation arises due to the fact that BGP relies on long-lived persistent TCP sessions with larger window sizes to function. When a BGP session is disrupted, the BGP application restarts and attempts to re-establish a connection to its peers. This may result in a brief loss of service until the fresh routing tables are created.

In a TCP session, the endpoints can negotiate a TCP Window size. When this is taken into account, instead of attempting to send a spoofed packet with all potential sequence numbers, the attacker would only need to calculate a valid sequence number that falls within the next expected ISN plus or minus half the window size. Therefore, the larger the TCP Window size, the the larger the range of sequence numbers that will be accepted in the TCP stream. According to Paul Watson's report, with a typical xDSL data connection (80 Kbps, upstream) capable of sending of 250 packets per second (pps) to a session with a TCP Window size of 65,535 bytes, it would be possible to inject a TCP packet approximately every 5 minutes. It would take approximately 15 seconds with a T-1 (1.544 Mbps) connection. These numbers are significant when large numbers of compromised machines (often called "botnets" or "zombies") can be used to generate large amounts of packets that can be directed at a particular host.

To protect against such injections, [RFC 2385](#) provides a method of using MD5 signatures on the TCP Headers. If this form of verification is supported and enabled between two peers, then an attacker would have to obtain the key used to transmit the packet in order to successfully inject a packet into the TCP session. Another alternative would be to tunnel BGP over IPsec. Again, this would provide a form of authentication between the BGP peers and the data that they transmit. The lack of authentication when using TCP for BGP makes this type of attack more viable.

US-CERT is tracking this issue as [VU#415294](#). This reference number corresponds to [CVE](#) candidate [CAN-2004-0230](#).

NISCC is tracking this issue as [Advisory 236929](#).

II. Impact

Sustained exploitation of the TCP injection vulnerability with regard to the BGP vulnerability could lead to a denial-of-service condition that could affect a large segment of the Internet community. Normal operations would most likely resume shortly after the attack stopped.

Since the [TCP/IP Initial Sequence Number vulnerability \(VU#498440\)](#) has been proven more viable of an attack, any services or sites that rely on persistent TCP sessions could also be affected by this vulnerability. Impacts could range from data corruption or session hijacking to a denial-of-service condition.

III. Solution

Apply a patch from your vendor

Please see your vendor's statement regarding the availability of patches, updates and mitigation strategies.

Workaround

Deploy and Use Cryptographically Secure Protocols

TCP initial sequence numbers were not designed to provide proof against TCP connection attacks. The lack of cryptographically-strong security options for the TCP header itself is a deficiency that technologies like [IPSec](#) try to address. It must be noted that in the final analysis that if an attacker has the ability to see unencrypted TCP traffic generated from a site, that site is vulnerable to various TCP attacks - not just those mentioned here. A stronger measure that would aid in protecting against such TCP attacks is end-to-end cryptographic solutions like those outlined in various [IPSec](#) documents.

The key idea with an end-to-end cryptographic solution is that there is some secure verification that a given packet belongs in a particular stream. However, the communications layer at which this cryptography is implemented will determine its effectiveness in repelling ISN based attacks. Solutions that operate above the Transport Layer (OSI Layer 4), such as SSL/TLS and SSH1/SSH2, only prevent arbitrary packets from being inserted into a session. They are unable to prevent a connection reset (denial of service) since the connection handling will be done by a lower level protocol (i.e., TCP). On the other hand, Network Layer (OSI Layer 3) cryptographic solutions such as IPSec prevent both arbitrary packets entering a transport-layer stream and connection resets because connection management is directly integrated into the secure Network Layer security model.

The solutions presented above have the desirable attribute of not requiring any changes to the TCP protocol or implementations to be made. Some sites may want to investigate hardening the TCP transport layer itself. [RFC2385](#) ("Protection of BGP Sessions via the TCP MD5 Signature Option") and other technologies provide options for adding cryptographic protection within the TCP header at the cost of some potential denial of service, interoperability, and performance issues.

Ingress filtering

Ingress filtering manages the flow of traffic as it enters a network under your administrative control. You can configure your BGP routers to only accept packets on a specific network connection. Servers are typically the only machines that need to accept inbound connections from the public Internet. In the network usage policy of many sites, there are few reasons for external hosts to initiate inbound connections to machines that provide no public services. Thus, ingress filtering should be performed at the border to prohibit externally initiated inbound connections to non-authorized services. In this fashion, the effectiveness of many intruder scanning techniques can be dramatically reduced.

Network Isolation

Complex networks can benefit by separating data channels and control channels, such as BGP, into different logical or physical networks. Technologies such as VLANs, VPNs, leased links, and NAT may all be able to contribute to separating the transmission of control information from the transmission of the data stream.

Egress filtering

Egress filtering manages the flow of traffic as it leaves a network under your administrative control. There is typically limited need for machines providing public services to initiate outbound connections to the Internet.

In the case of BGP, only your BGP routers should be establishing connections to your peers. Other BGP traffic generated on your network could be a sign of an attempted attack.

Appendix A. Vendor Information

For vendor information, please see NISCC Vulnerability Advisory 236929 "Vulnerability Issues in TCP" (<http://www.uniras.gov.uk/vuls/2004/236929/index.htm>) or Vulnerability Note VU#415294 (<http://www.kb.cert.org/vuls/id/415294#systems>). As vendors report new information to US-CERT, we will update the vulnerability note. If a particular vendor is not listed in either the NISCC advisory, or the vulnerability, we recommend that you contact them for their comments.

US-CERT thanks Paul Watson, Cisco Systems and NISCC for notifying us about this problem and for helping us to construct this advisory.

Feedback can be directed to the [US-CERT Technical Staff](#).

Copyright 2004 Carnegie Mellon University. [Terms of use](#)

Revision History

April 20, 2004: Initial release

Last updated April 20, 2004