

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

This paper was not written with the TCP/IP expert in mind. However, even those that consider themselves 'experts' may learn something. If this paper does not go into as much detail as you would like, remember, its not for experts.

### Preface

Panix.com is a large Internet Service Provider that provides access to several hundred thousand new york inhabitants. Panix came under heavy fire on Spet. 6, 1996.

The attack on Panix was one of the worst DoS (Denial of Service) attacks seen. A DoS does not require that user gain access to a network, instead, it renders a network completely useless, if its devastating enough. Tha Panix attack was a SYN Flood.

### What is a SYN Flood?

A SYN attack is an attack thats based on a 3-way TCP/IP handshake. So, before we answer what a SYN Flood is, lets get some background on the 3-way handshake.

### The Quick and Dirty on 3-Way Handshakes.

A TCP system relies on a virtual circuit connection that is established between the requesting machine and its target (server). This connection circuit is accomplished through a three part process known as the 3-way handshake:

**Step 1:** The requesting client machine sends a connection request, specifying a port to connect to on the remote machine.

**Step 2:** The server machine answers with both an acknowledgement and a queue for the connection.

**Step 3:** The client returns an acknowledgement and the circuit is opened.

After the connection circuit is established, data can simultaneously travel in both directions. This is known as a full-duplex transmission path. Full-duplex transmission allows data to travel to both machines at the same time.

### Ok, so now.. Whats a SYN Flood?

Well, lets take a quick look at something that happens during that 3-way handshake:

**Step 1:** Client sends a Synchronize (SYN) request to the server. Accompanying this SYN request is an initial sequence number (ISN). The ISN plays an important role in the ordering of all messages subsequently exchanged on the connection (ISN's also play a role in Spoofing attacks).

**Step 2:** The server receives the clients SYN request, it replies with a SYN, an ISN and an acknowledgment (also known as an ACK).

**Step 3:** Client acknowledges the servers ISN and ACK.

With that in mind, lets continue. In a SYN Flood attack, the requesting (client) machine sends a series of connection requests but fails to acknowledge (ACK) the servers response. Because the server never receives the clients ACK, it waits. If this process is repeated many times, it renders the target server's ports useless becuase the target is waiting for a response (ACK) from the client. These requests are dealt with in sequence, eventually, the target will abandon waiting for each ACK.

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

Nevertheless, if it receives tens or even hundreds of these requests, the port will remain engaged until it has processed, or discarded, each client request.

[- The term SYN\_FLOOD is taken from the tools that are used to perform these attacks, known as SYN Flooders. During a typical attack, a series of these packets are forwarded to a target (perhaps from a WinGated system) purporting to be from another address that is non-existent. The target machine therefore cannot resolve to the host. In turn, the server is being flooded with requests that cannot be fulfilled. -]

### -----From the CERT Advisory

I. Description: When a system (called the client) attempts to establish a TCP connection to a system providing a service (the server), the client and server exchange a set sequence of messages. This connection technique applies to all TCP connections--telnet, Web, email, etc. The client system begins by sending a SYN message to the server. The server then acknowledges the SYN message by sending SYN-ACK message to the client. The client then finishes establishing the connection by responding with an ACK message. The connection between the client and the server is then open, and the service-specific data can be exchanged between the client and the server. Client and server can now send service-specific data. The potential for abuse arises at the point where the server system has sent an acknowledgment (SYN-ACK) back to client but has not yet received the ACK message. This is what we mean by half-open connection. The server has built in its system memory a data structure describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially-open connections. Creating half-open connections is easily accomplished with IP spoofing. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system. The half-open connections data structure on the victim server system will eventually fill; then the system will be unable to accept any new incoming connections until the table is emptied out. Normally there is a timeout associated with a pending connection, so the half-open connections will eventually expire and the victim server system will recover. However, the attacking system can simply continue sending IP-spoofed packets requesting new connections faster than the victim system can expire the pending connections. In most cases, the victim of such an attack will have difficulty in accepting any new incoming network connection. In these cases, the attack does not affect existing incoming connections nor the ability to originate outgoing network connections. However, in some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative. The location of the attacking system is obscured because the source addresses in the SYN packets are often implausible. When the packet arrives at the victim server system, there is no way to determine its true source. Since the network forwards packets based on destination address, the only way to validate the source of a packet is to use input source filtering.

II. Impact: Systems providing TCP-based services to the Internet community may be unable to provide those services while under attack and for some time after the attack ceases. The service itself is not harmed by the attack; usually only the ability to provide the service is impaired. In some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative.

III. Solution: There is, as yet, no generally accepted solution to this problem with the current IP protocol technology. However, proper router configuration can reduce the likelihood that your site will be the source of one of these attacks. Many networking experts are working together to devise improvements to existing IP implementations to "harden" kernels to this type of attack. When these

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

improvements become available, we suggest that you install them on all your systems as soon as possible. This advisory will be updated to reflect changes made by the vendor community.

**IV. Detecting an Attack:** Users of the attacked server system may notice nothing unusual since the IP-spoofed connection requests may not load the system noticeably. The system is still able to establish outgoing connections. The problem will most likely be noticed by client systems attempting to access one of the services on the victim system. To verify that this attack is occurring, check the state of the server system's network traffic.

For example, on SunOS this may be done by the command:

```
netstat -a -f inet
```

Note that use of the above command depends on the OS version, for example for a FreeBSD system use `netstat -s |grep "listenqueue overflows"` Too many connections in the state "SYN\_RECEIVED" could indicate that the system is being attacked.

### **But it said there is no cure... is there or isn't there?**

There are a couple of methods to help buffer or stop these attacks. 2 Methods Ive studied are SYN Storm Mods by Avi Freedman and SYN Cookies by Alex Yuriev and Avi Freedman. Read on, read on.

### **Avi Freedmans SYN Storm Modifications (also called Adaptive Time-Outs).**

One way to deal with SYN Floods is to slightly modify the algorithm to allow a very large number (perhaps thousands) of embryonic TCP sockets to be queued up 'on' your server socket. By modifying this algorithm, you can basicly somewhat buffer or prevent SYN damage or attacks. Although this modification has already been somewhat adopted and used in Unix enviroments (SunOS, FreeBSD, OpenBSD and NetBSD), to my knowledge, it has yet to be adopted by Microsoft.

### **SYN Cookies /developed by Alex Yuriev and Avi Freedman.**

Using a secret number plus data extracted from certain fields of a TCP header, SYN Floods can almost always be prevented. The extraction from certain header fields would be indexed and referenced to data from other parts of the header. This information is input into a one-way cryptographic hash function. Put the result of the hash into the header as part (or all, not clearly stated) of the acknowledgement. If a packet has an ACK flag raised, then the ACK number of that packet has the result of said crypto calculation. The information is checked by feeding back into the one-way crypto hash. If the result matches, then is a SYN-ACK packet for the SYN-ACK packet that was sent by the server, and communication continues, if the result doesnt match, the packet is immediately discarded, and no handshake is complete. In this method, the server does not have to "wait", and the attack is stopped.

### **So what about NT...Microsofts Answer:**

Here is Microsofts provided resolution to the problem:

```
*****
* 1. Tcpi.sys times out half-open connections faster *
*****
```

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

A new version of Tcpip.sys has been produced that allows control of the number of times a response to a TCP connection request (SYN-ACK) will be retransmitted. Control is handled through a new registry parameter:

```
HKEY_LOCAL_MACHINE
\SYSTEM
\CurrentControlSet
\Services
\Tcpip
\Parameters
\TcpMaxConnectResponseRetransmissions
Value Type: REG_DWORD
Valid Range: 0-0xFFFFFFFF
Default: 3
```

The default value for this parameter is now 3. The following table shows Windows NT 4.0 TCP/IP behavior for various values of this parameter:

Value	Retransmission Times	Elapsed Time	Comments
3	3, 6, and 12 seconds	45 seconds	Cleanup 24 secs after last retx
2	3, and 6 seconds	21 seconds	Cleanup 12 secs after last retx
1	3 seconds	9 seconds	Cleanup 6 secs after last retx

This parameter changes the default time that it takes to clean up a half-open TCP connection from 189 seconds to 45 seconds, and provides more granular control to the administrator. A site that is under heavy attack might set the value as low as "1". A value of "0" is also valid; however if this parameter is set to 0, SYN-ACKs will not be retransmitted at all, and will time out in 3 seconds. With the value this low, legitimate connection attempts from distant clients may fail.

```
*****
* 2. NetBT has a Higher, Configurable Backlog *
*****
```

NetBT (NetBIOS over TCP/IP) uses TCP port 139 and is used by Microsoft Network Services such as file and print sharing. Version 3.51 and 4.0 NetBT has a "backlog" of connection blocks available that is two plus an incremental number depending on the NetBT clients (such as the redirector, server, and any NetBIOS applications running). On a typical server, this number will be 7-11. A new version of NetBT has been produced that automatically allocates more connection blocks as needed, in a configurable manner.

On a connection event, it now checks to see if the number of free blocks is below 2, and if so, adds an "increment" number of blocks, where "increment" is configurable in the registry as shown here:

```
HKEY_LOCAL_MACHINE
\SYSTEM
\CurrentControlSet
\Services
\NetBt
```

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

**\Parameters**  
**\BacklogIncrement**  
**Value Type: REG\_DWORD**  
**Valid Range: 1-0x14 (1-20 decimal)**  
**Default: 3**

Each connection block consumes 78 bytes of memory. The total number of connection blocks that can be allocated by NetBT is also registry configurable:

**HKEY\_LOCAL\_MACHINE**  
**\SYSTEM**  
**\CurrentControlSet**  
**\Services**  
**\NetBt**  
**\Parameters**  
**\MaxConnBackLog**  
**Value Type: REG\_DWORD**  
**Valid Range: 1-0x9c40 (1-40,000 decimal)**  
**Default: 1000**

MaxConnBackLog defaults to 1000, but can be set as high as 40,000. Connection blocks are "scavenged," or recycled, when the SYN-ACK retransmission timer expires and TCP fails the connection attempt.

\*\*\*\*\*  
\* 3. Afd.sys has been modified to withstand large numbers of \*  
\* "half-open" connections efficiently \*  
\*\*\*\*\*

Windows Sockets applications such as ftp servers and web servers have their connection attempts handled by Afd.sys. Afd.sys has been modified to support large numbers of connections in the "half-open" state without denying access to legitimate clients. This is accomplished by allowing the administrator to configure a dynamic backlog.

The new version of Afd.sys supports four new registry parameters that can be used to control the dynamic backlog behavior.

EnableDynamicBacklog is a global switch to enable or disable dynamic backlog. It defaults to 0 (off), and this setting provides no change from the existing versions. Setting it to 1 enables the new dynamic backlog feature.

**HKEY\_LOCAL\_MACHINE**  
**\SYSTEM**  
**\CurrentControlSet**  
**\Services**  
**\AFD**  
**\Parameters**  
**\EnableDynamicBacklog**  
**Value Type: REG\_DWORD**

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

**Valid Range: 0,1**

**Default: 0**

**Suggested value for a system under heavy attack: 1**

MinimumDynamicBacklog controls the minimum number of free connections allowed on a listening endpoint. If the number of free connections drops below this value, then a thread is queued to create additional free connections. This value should not be made too large, as the dynamic backlog code engages whenever the number of free connections falls below this value. Too large a value may lead to a performance reduction.

**HKEY\_LOCAL\_MACHINE**

**\SYSTEM**

**\CurrentControlSet**

**\Services**

**\AFD**

**\Parameters**

**\MinimumDynamicBacklog**

**Value Type: REG\_DWORD**

**Valid Range: 0-0xFFFFFFFF**

**Default: 0**

**Suggested value for a system under heavy attack: 20**

MaximumDynamicBacklog controls the maximum number of "quasi-free" connections allowed on a listening endpoint. "Quasi-free" connections include the number of free connections plus those connections in a half-connected (SYN\_RECEIVED) state. No attempt is made to create additional free connections if doing so would exceed this value.

**HKEY\_LOCAL\_MACHINE**

**\SYSTEM**

**\CurrentControlSet**

**\Services**

**\AFD**

**\Parameters**

**\MaximumDynamicBacklog**

**Value Type: REG\_DWORD**

**Valid Range: 0-0xFFFFFFFF**

**Default: 0**

Suggested value for a system under heavy attack: Memory dependent. This value should not exceed 5000 per 32M of RAM installed in the server, in order to prevent exhaustion of non-paged pool when under attack.

DynamicBacklogGrowthDelta controls the number of free connections to create when additional connections are necessary. Be careful with this value, as a large value could lead to explosive free connection allocations.

**HKEY\_LOCAL\_MACHINE**

**\SYSTEM**

**\CurrentControlSet**

# SYN FLOODS & SYN COOKIES

## The Cause and Cure

By NeonSurge

**\Services**

**\AFD**

**\Parameters**

**\DynamicBacklogGrowthDelta**

**Value Type: REG\_DWORD**

**Valid Range: 0-0xFFFFFFFF**

**Default: 0**

**Suggested value for a system under heavy attack: 10 (0xa)**

To take advantage of the changes to Afd.sys, Windows Sockets applications must specifically request a backlog greater than the value configured for MinimumDynamicBacklog when they issue their listen() call. Microsoft applications such as Internet Information Server (which has a default backlog of 25) are configurable. Application-specific details are available from the Microsoft Knowledgebase at <http://www.microsoft.com/kb>.

The modified drivers for Windows NT and instructions for installing them are available from Microsoft support channels or directly from the Microsoft website.

[The information above is Copyright Microsoft Corporation 1996]

### **This Authors Opinion**

My suggestion would be for Microsoft to adopt SYN Cookies. To this date, they seem to be the strongest and most logical defense. That's it for now, more on these SYN cures as I stumble, read, or perhaps make one myself.

NeonSurge

Rhino9: The WindowsNT Security Research Team

[rhino9.abys.com](http://rhino9.abys.com)