SYN Cookies

Ed L. Cashin

(B)RATS
November 2001

Topics

The Solution and the Problem

Solution: SYN Cookies
Problem: tcpcb Limit


Solution: SYN Cookies

Let's start with the denouement: you don't need a queue of tcpcb's.

- use cryptographic hashing to make a sequence number
- don't allocate tcpcb


Problem: tcpcb Limit

In the first part of the TCP handshake, a TCP control block of about 140 bytes is traditionally allocated to store information about the new connection.

Here is the TCP handshake that we saw in the synkill paper.

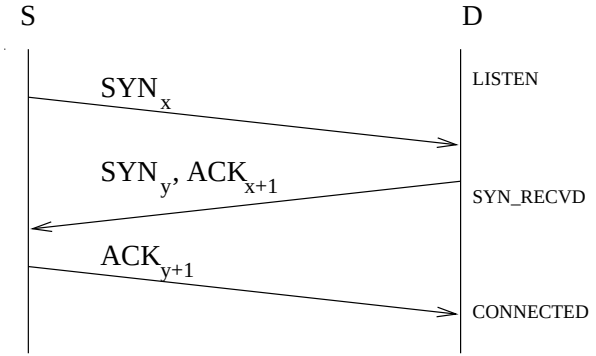

Figure 1.1 TCP handshake

- If O.S. must allocate a tcpcb for each incoming SYN packet and . . .
- if there's a limit to the number that may be allocated

. . . then a SYN flood may prevent new TCP connections.

Implications

The implications of SYN Cookies as a solution to the vulnerability of having a finite half-open connection queue.

Scales Well

Different from increasing the limit.

Since the queue is eliminated entirely, this specific solution to this specific problem should scale well, even under currently-popular DDOS attacks through fat pipes.


Firewall

A firewall with SYN cookies turned on could play one of the roles described in the synkill paper.

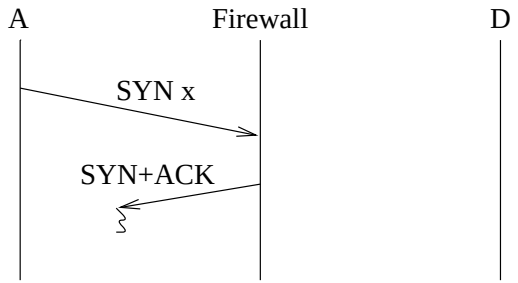

Figure 2.1 invulnerable firewall

Hosts

"If it's possible to make a firewall immune to DOS via SYN flooding, why not make the hosts invulnerable?"


Caveats

Even without the tcpcb limit, other resources are limited.

Examples: Network bandwidth; CPU.


The Method

People
Disadvantages
The Algorithm: ISN
The Algorithm: Handshake Step 2
The Algorithm: Handshake Step 3


People

- ⋆ Dan Bernstein
- ⋆ Eric Schenk
- ⋆ Andi Kleen


Disadvantages

- no fancy TCP options during SYN flood.
- limited choice of MSS values.
- lost ACK may hang clients.
- brute-force sequence number guessing
  inject 1 per 2^{27} packets

Also, unlike the synkill solution, only one host is protected.

The Algorithm: ISN

2 constant secret keys: "sec1" and "sec2".

A constant sorted table of 8 common MSS values, "msstab".

Keep track of a "last overflow time."

Maintain a counter that increases slowly over time and never repeats, such as "number of seconds since 1970, shifted right 6 bits."

When a SYN comes in from $(saddr, sport)$ to $(daddr, dport)$ with ISN x, find the largest i for which $msstab[i] \leq$ the incoming MSS. Compute . . .

$$\begin{aligned} z = {} & \mathrm{MD5}(sec1, saddr, sport, daddr, dport, sec1) \\ & + x \\ & + (counter \ll 24) \\ & + (\mathrm{MD5}(sec2, counter, saddr, sport, daddr, dport, sec2) \,\%\, (1 \ll 24)) \end{aligned}$$

. . . and then . . .

$$y = (i \ll 29) + (z \,\%\, (1 \ll 29)).$$

. . . where y is the ISN.

The Algorithm: Handshake Step 2

If not out of memory for tcpcb's, create a tcpcb as usual, with y as our ISN. Send back a SYNACK packet.

Else the queue is full, so set the "last overflow time" to the current time and send the SYNACK anyway, with all fancy options turned off. Do not allocate tcpcb.

The Algorithm: Handshake Step 3

1. Look for a $(saddr, sport, daddr, dport)$ tcpcb. If it's there, done.
2. If the "last overflow time" is earlier than a few minutes ago, give up.
3. Figure out whether ISN makes sense. This means recomputing y as above, for each of the counters that could have been used in the last few minutes (say, the last four counters), and seeing whether any of the y's match the ISN in the bottom 29 bits. If none of them do, give up.
4. Create a new tcpcb. The top three bits of our ISN give a usable MSS. Turn off all fancy options.

Current Status

Kleen's Advice
Bernstein's Perspective
Valuable TCP Options
Random Drop
Tests
Conclusion


Kleen's Advice

Andi Kleen, who implemented the syncookie feature in the Linux kernel, says this:

```
From: Andi Kleen <ak@suse.de>
Subject: Re: testing syncookie functionality
To: Ed L Cashin <ecashin@terry.uga.edu>
Date: Tue, 6 Nov 2001 01:17:05 +0100

Hi,

First I would suggest not putting much time anymore
into syncookies.  They're basically obsolete because
the cost of not using time stamps and SACK is too high,
and linux has the infrastructure now to keep a big
enough real queue that makes them not really needed
anymore.

Also they don't have enough bits to be secure from
brute force.
```


Bernstein's Perspective

From: "D. J. Bernstein" <djb@cr.yp.to>
Subject: Re: SYN cookies testing and use
To: Ed L Cashin <ecashin@terry.uga.edu>
Date: 9 Nov 2001 19:58:17 -0000

Kleen is an idiot. Here's what Google's Jim Reese said about SYN cookies in a talk a year ago:

> Security. Obviously a big issue, as we get more and more of these SYN flood attacks. ... The script kiddies are out there and they're out there to get us. We've seen a _tremendous_ increase in the amount of attacks on us as we grow more popular. It's inevitable. Every site sees it.
>
> SYN flood attacks are actually extremely well handled now by the Linux kernel with SYN cookies. They work extremely well. If you're not using them, you should be.

Valuable TCP Options

Today some TCP options are more critical than in 1996.

- SACK and D-SACK

  Selective acknowledgement and duplicate SACK.

- timestamping

  For RTT calculation and also protection against wrapped sequence numbers.

- window scaling

These options are especially important for "long fat pipes."

Random Drop

The "intelligent dropping algorithms" Kleen refers to are likely variants on random drop.

Bernstein: random drop adversely affects legitimate clients' new connections.


Tests

- lower the queue size

  similar to queue full of legitimate users

- three hosts
  - attacker
  - victim
  - monitor
- the tools
  - synbo
  - connect.rb
  - icmpecho.rb

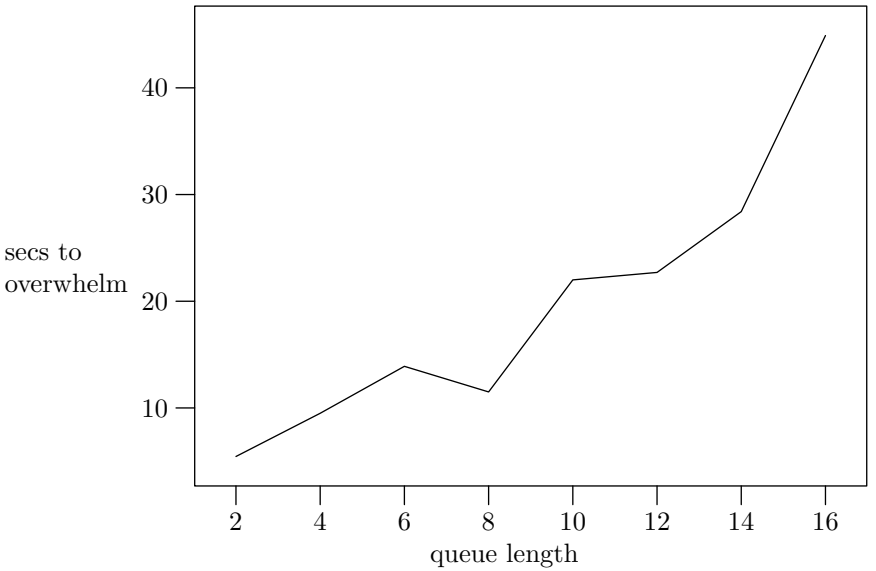

Figure 4.1 without SYN cookies

Conclusion

SYN cookies solve the full-queue problem but ...

- the cost of missing fancy TCP options is greater today
- SYN floods cause other problems, like network congestion


Appendices

Resources
Packet Rates


Resources

[1.] Dan J. Bernstein. *SYN Cookies.* http://cr.yp.to/syncookies.html

[2.] Dan J. Bernstein. personal email correspondence, Nov. 2001.

[3.] Brendan Conoboy and Erik Fichtner. *ipfilter HOWTO.* http://www.obfuscation.org/ipf/ipf-howto.txt

[4.] Andi Kleen. personal email correspondence, Nov. 2001.

[5.] Christoph L. Schuba, et al. *Analysis of a Denial of Service Attack on TCP.* (The "synkill paper".)

[6.] W. Richard Stevens. *TCP/IP Illustrated, V.1, The Protocols.* Addison-Wesley, 1994.

[7.] W. Richard Stevens and Gary R. Wright. *TCP/IP Illustrated, V.2, The Implementation.* Addison-Wesley, 1995.

[8.] W. Richard Stevens. *UNIX Network Programming, V.1, second ed.* Prentice Hall PTR, 1998.

Packet Rates

Net Type	Mbps	SYN's/sec
T1	1.5	4,825
10 Base-T	10.0	31,025
T3	45.0	140,621
100 Base-T	100.0	310,025
OC-3	155.0	484,375
GigE	1,000.0	3,100,250

Table 5.1 packet rates of popular networks

fin