IPSec over Ham Radio

Steven Kreuzer

NYC*BSD Users Group

June 2007

Techniques For Tuning FreeBSD To Keep Furries, Angry Goths & Annoying Brazilians at Bay

What's The Deal?

- The Internet is full of jerks
- Sometimes those jerks will consume your available Internet-facing bandwidth, or overpower your CPU, in an attempt to take you offline
- We call this a Denial of Service attack

What's a Denial of Service Attack?

- An attempt to make a computer resource unavailable to its intended users.
  - Web servers stop serving web pages
  - Email servers stop accepting or delivering e-mail
  - DNS servers stop resolving domain names
  - In general, servers stop serving
- The end result is those users keep calling you until they can get their email

How is This Accomplished?

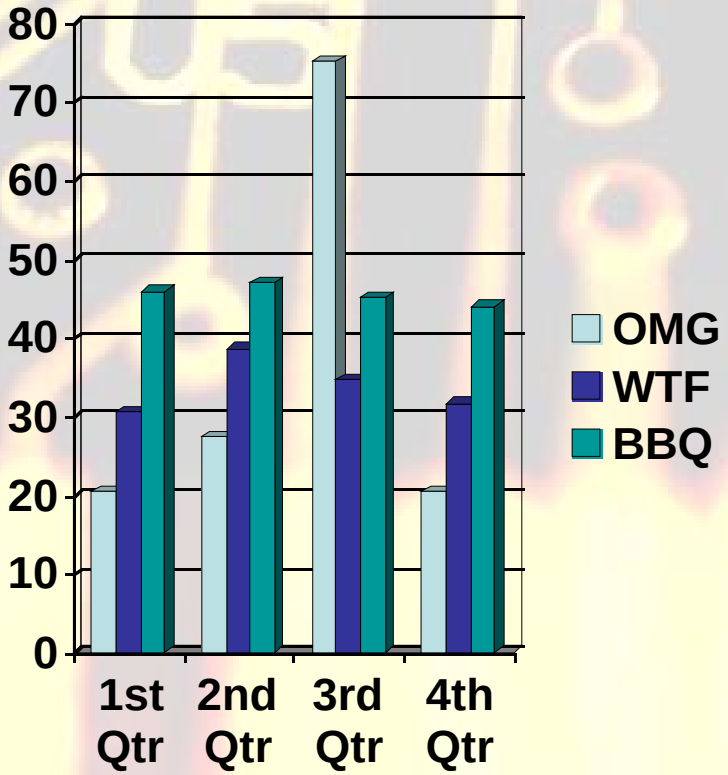
- Obstruct the communication media between the intended users and the victim so that they can no longer communicate adequately.
- Force the victim computer to reset or consume its resources such that it can no longer provide its intended service.

How is This Accomplished?

- Complete consumption of a resource such as bandwidth, memory, CPU, file handles, or any other finite asset.
- Exploiting a weakness in the service to stop it functioning or causing the service to crash.

So? Put a Firewall in Front of Your Machines!

- Sometimes FreeBSD is your firewall (OMG!)
- Sometimes you don’t always have the luxury of a firewall (WTF?)
- The graph on the side contributes nothing, but looks impressive from far away (BBQ?!?)



What's the Goal Then?

- Understand how a denial of service attack works.
- Configure a FreeBSD machine to mitigate the effects of a denial of service attack.
- Try not to break anything in the process, and make it as transparent to the end user as possible.

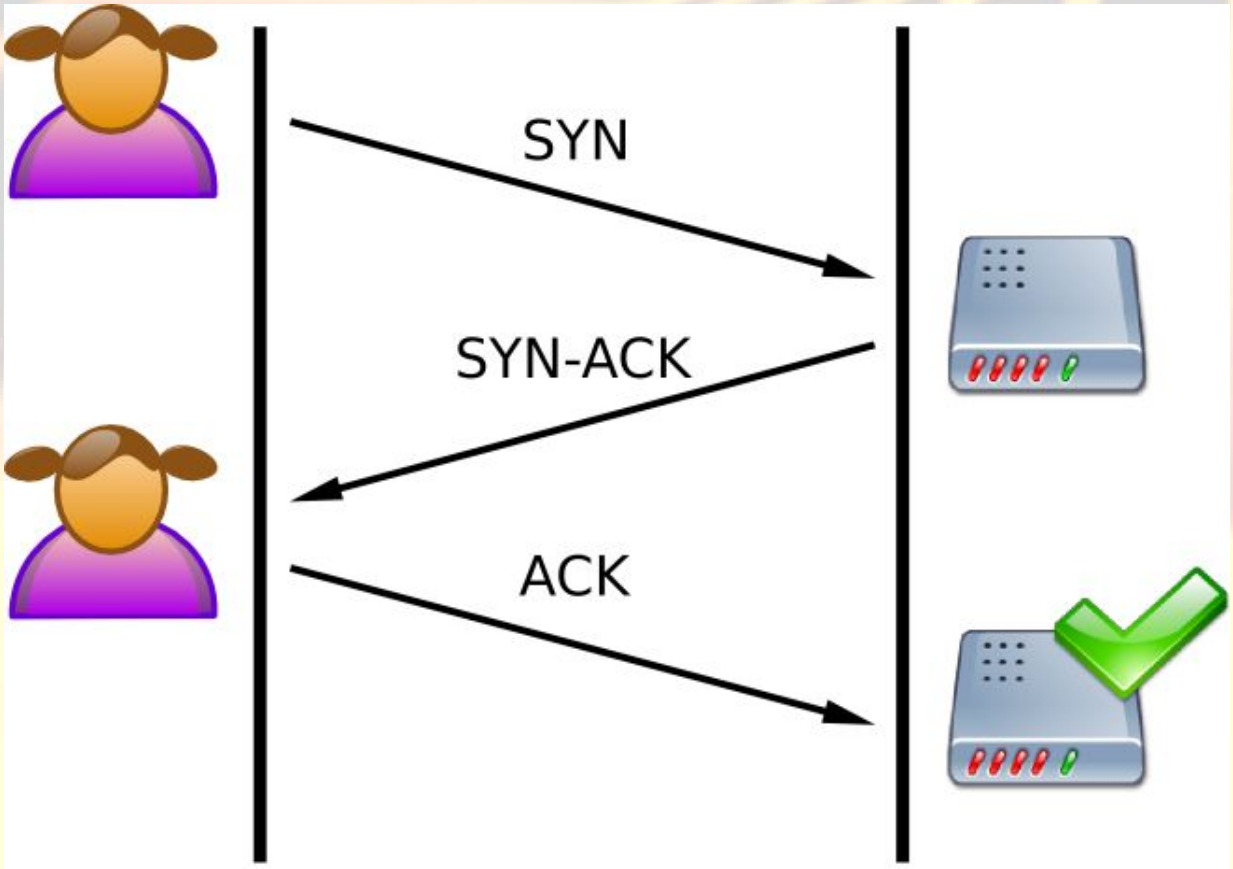
Common Attack Vectors

- Sending a large number of UDP packets to a target system
- Sending a succession of SYN requests to a target system.

What is UDP Flooding?

- Initiated by sending a large number of UDP packets to random ports
- The server will have to check for the daemon listening at that port and:
  - Determine nothing is listening on that port
  - Reply with an ICMP Destination Unreachable
- The system will be forced into sending many ICMP packets, eventually making it unreachable.

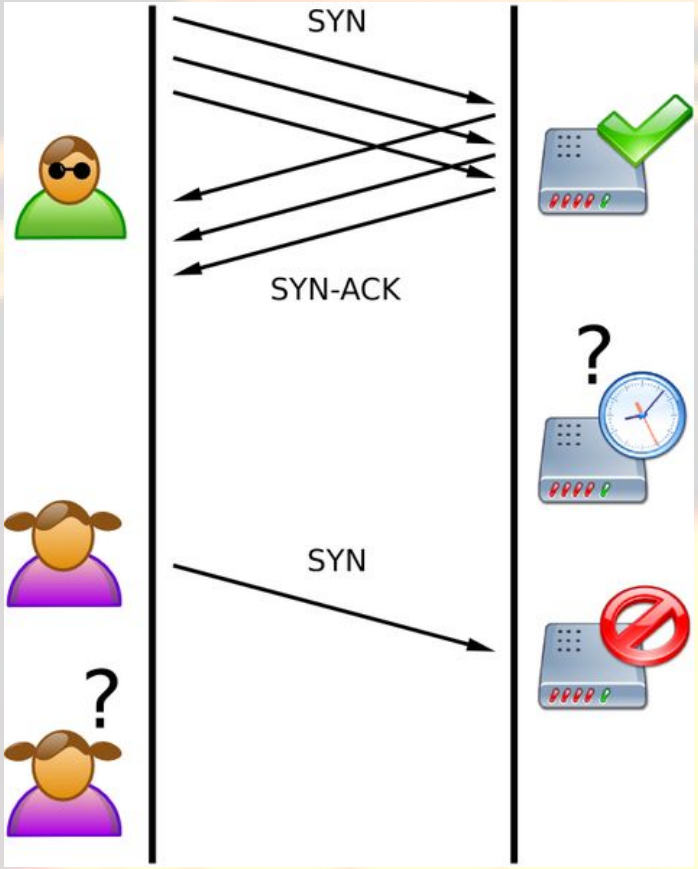
TCP Connection Establishment



Behind the Scenes?

- Three memory structures are allocated for TCP connections:
  - Socket Structure (socket{})
  - IP Control Block Structure (inpcb{})
  - TCP Control Block Structure (tcpcb{})
- Every time a client SYN arrives on a valid port these memory structures must be allocated.

What is a SYN Flood?



- Attackers sends a SYN packet to server
- Server responds with a SYN/ACK
- Attacker doesn’t respond with ACK

Dealing With Flooding

- Increase the amount of resources available for handling connections.
- Limit resources wasted processing bogus requests.

SYN Cache

- Protects against flooding by minimizing the amount of state kept on the server.
- Holds TCP options from the SYN and enough state to perform a SYN/ACK retransmission.
- Takes up less space than a TCP control block structure.
- An incoming ACK for the SYN/ACK that matches a syncache entry causes the system to:
    - Create a TCP control block with the options stored in the syncache entry
    - Discard the syncache entry

How SYN Cache Works

- Replaces per socket linear chain of incomplete queued connections with a global hash table.
- Provides an upper boundary in the amount of memory it takes up.
- Limits the number of entries in a given hash bucket.

How SYN Cache Works

- Hash value is computed on incoming packet using:
  - Source and destination address
  - Source and destination port
  - Randomly chosen secret
- Result is an index in a hash table

How SYN Cache Works

- If a new entry overflows the per-bucket limit, the oldest entry in the bucket is dropped.
- If the total number of entries in the hash table is exceeded, oldest entry is dropped.
- Memory and CPU required is bounded.

SYN Cache Tunables

- net.inet.tcp.syncache.cachelimit
  - Determines the maximum number of syncache entries that may be allocated.
- net.inet.tcp.syncache.hashsize
  - Controls the size of the hash table.
- net.inet.tcp.syncache.bucketlimit
  - Caps the size of each hash chain.

SYN Cache Tunables

- net.inet.tcp.syncache.rexmtlimit
  - Determines how many times a SYN/ACK should be retransmitted.
- net.inet.tcp.syncache.count
  - Indicates how many entries are currently present in the syncache.

SYN Cookies

- When a syncache bucket overflows, a fallback mechanism exists.
- Avoids dropping connections by keeping initial SYN state in the network.
- Sends a cryptographic value in SYN/ACK which is returned in the ACK.

How SYN Cookies Work

- Client sends a SYN packet
- Server responds but discards the SYN queue entry.
- Server reconstructs queue entry using information encoded in the sequence number upon receiving ACK.

How SYN Cookies Work

- The first sequence number sent by an endpoint can be any value as decided by that endpoint.
- Many implementations use zero as the initial sequence number
- SYN Cookies initial sequence numbers are carefully constructed according to defined rules.

How SYN Cookies Work

- Basis is a table of 32 bit values obtained from arc4random()
- The source and destination address and port and a secret are hashed using md5.
- 25 bits from the result of the hash are sent back as the cookie.

How SYN Cookies Work

- Upon receiving an ACK with a valid cookie from the client will establish a connection.
- From this point forward, the connection proceeds as normal.
- If ACK contains an invalid or expired cookie, the packet is discarded.

SYN Cookie Drawbacks

- Does not break any protocol specifications.
- Should be compatible with all TCP implementations.
- However…
  - The server is limited to only 4 predefined MSS values.
  - The server must reject all TCP options because the server discards the SYN queue entry where that information would otherwise be stored.

Decrease Maximum Segment Life

- Amount of time to wait for an ACK in reply to a SYN/ACK or FIN/ACK
- If an ACK is not received in this time, the segment can be considered "lost" and the network connection is freed.

Decrease Maximum Segment Life

- The FreeBSD TCP/IP stack holds on to half open connections for 30 seconds.
- This is extremely generous considering most connections take less than one second to establish.
- /sbin/sysctl net.inet.tcp.msl=7500

TCP/UDP Blackhole

- Control what happens when a packet is received on a closed port
- Packets arriving on a closed port will be dropped without notification being sent.
- This saves CPU time and bandwidth.

Blackhole Tunables

- /sbin/sysctl net.inet.tcp.blackhole=1
  - Drop SYN packets
- /sbin/sysctl net.inet.tcp.blackhole=2
  - Drop all TCP packets
- /sbin/sysctl net.inet.udp.blackhole=1
  - Drop all UDP packets

Optimizing Device Performance

- Traditionally, each time the network card needs attention it generates an interrupt request.
- The request causes a context switch and a call to an interrupt handler.

Context Switching

- When the CPU and kernel have to switch between user and kernel land.
- It i$ an extremely expen$ive operation.

(see what I did there?)

DEVICE_POLLING

- Changes the method through which data gets from your network card to the kernel.
- Provides more control to the operating system on how and when to handle devices.
- Kernel polls the network card itself at certain predefined times.
- Kernel decides when it is most efficient to poll for updates and for how long

Supported Network Cards

- DEC/Intel 21143 and clone 10/100 Ethernet driver; dc(4)
- Intel EtherExpress PRO/100 Ethernet device driver; fxp(4)
- RealTek 8129/8139 Fast Ethernet device driver; rl(4)
- SiS 900, SiS 7016 and NS DP83815/DP83816 Fast Ethernet device driver; sis(4)

Disable Auto-Negotiation

- If your switch or server is set to use auto-negotiation, every few moments it stops transferring network traffic in order to renegotiate its speed.

ACCEPT_FILTER_HTTP

- Buffers incoming connections until a complete HTTP request arrives
- The server will not have to context switch several times before performing the initial parsing of the request.
- This reduces the amount of required CPU utilization to handle incoming requests by keeping active processes in preforking servers such as Apache

Where to go From Here

- Read tuning(7). Its quite fascinating!
- Post questions to talk@lists.nycbug.org
- Email me directly: skreuzer@exit2shell.com

Any (preapproved) Questions?

Happy Birthday Sonia!

FINALLY!
He's DONE!

Wasn't That Bad. Right?


HB
MÜNCHEN
Let's Go Drinking!