

Closing the Floodgates: DDoS Mitigation Techniques

Matthew Tanase 2003-01-07

Introduction

To be on the receiving end of a [distributed denial of service](#) (DDoS) attack is a nightmare scenario for any network administrator, security specialist or access provider. It begins instantly, without warning, and continues relentlessly: machines down, jammed bandwidth, overloaded routers. An effective, immediate response is often difficult and may depend on third parties, such as ISPs. With these challenges in mind, this article will explore some techniques that systems administrators and security professionals can employ should they ever find themselves in this rather undesirable situation.

Identifying a DDoS Attack

It's easy to recognize a DDoS attack: slow networks and servers crashing catch the eye of every administrator (or their users!). The first step in our defense is to identify and qualify what type of traffic is bombarding the network. Most DDoS attacks send a very specific type of traffic over and over: ICMP, UDP, TCP - albeit with forged source addresses. But an unusually large quantity of a certain packet type should be an accurate starting point for characterizing the attack and useful information for crafting filters in the future. The exception to this rule, to be addressed later, would be DDoS attacks directed at specific services, such as HTTP, using valid traffic and requests.

To identify and inspect the packets, we need to analyze the network traffic. This can be accomplished using two different methods depending on where the traffic is being examined. The first method can be used on a machine located on the target network. [tcpdump](#) is a widely available [sniffer](#) that works well for our purposes. Analyzing the output in real-time is impossible on a busy network, so the first step is to use the "-w" option to write the data to a file. Next, use a tool such as David Dittrich's modified version of [tcpdstat](#) or [tcptrace](#) to summarize the findings.

The output from a sample tcpdump file parsed using tcpdstat is below:

```
DumpFile: test
FileSize: 0.01MB
Id: 200212270001
StartTime: Fri Dec 27 00:01:51 2002
EndTime: Fri Dec 27 00:02:15 2002
TotalTime: 23.52 seconds
TotalCapSize: 0.01MB CapLen: 96 bytes
# of packets: 147 (12.47KB)
AvgRate: 5.56Kbps stddev:5.40K PeakRate: 25.67Kbps
```

```
### IP flow (unique src/dst pair) Information ###
```

```
# of flows: 9 (avg. 16.33 pkts/flow)
```

```
Top 10 big flow size (bytes/total in %):
```

```
26.6% 16.5% 14.7% 11.6% 9.8% 7.6% 5.4% 5.4% 2.5%
```

```
### IP address Information ###
```

```
# of IPv4 addresses: 7
```

```
Top 10 bandwidth usage (bytes/total in %):
```

```
97.5% 34.1% 31.2% 21.4% 10.7% 2.5% 2.5%
```

```
### Packet Size Distribution (including MAC headers) ###
```

```
<<<<
```

```
[ 32- 63]: 79
```

```
[ 64- 127]: 53
```

```
[ 128- 255]: 8
```

```
[ 256- 511]: 6
```

```
[ 512- 1023]: 1
```

```
>>>>
```

```
### Protocol Breakdown ###
```

```
<<<<
```

protocol	packets	bytes	bytes/pkt
[0] total	147 (100.00%)	12769 (100.00%)	86.86
[1] ip	147 (100.00%)	12769 (100.00%)	86.86
[2] tcp	107 (72.79%)	6724 (52.66%)	62.84
[3] telnet	66 (44.90%)	3988 (31.23%)	60.42
[3] pop3	41 (27.89%)	2736 (21.43%)	66.73
[2] udp	26 (17.69%)	4673 (36.60%)	179.73
[3] dns	24 (16.33%)	4360 (34.15%)	181.67
[3] other	2 (1.36%)	313 (2.45%)	156.50
[2] icmp	14 (9.52%)	1372 (10.74%)	98.00

It's easy to see how these useful tools can help you quickly determine what type of traffic is most prevalent. They save a lot of time by parsing and processing the captured packets.

A router can also be used to monitor incoming traffic. Through the use of specific access lists, a router can serve as a basic packet filter. It will also be the device you are likely focused on since it's the gateway between your network and the Internet. The following example from Cisco illustrates a very simple way of using access lists to monitor incoming traffic:

```
access-list 169 permit icmp any any echo
access-list 169 permit icmp any any echo-reply
access-list 169 permit udp any any eq echo
```

```
access-list 169 permit udp any eq echo any
access-list 169 permit tcp any any established
access-list 169 permit tcp any any
access-list 169 permit ip any any
```

```
interface serial 0
ip access-group 169 in
```

Using the "show access-list" command will summarize the match count for each rule:

```
Extended IP access list 169
permit icmp any any echo (2 matches)
permit icmp any any echo-reply (21374 matches)
permit udp any any eq echo
permit udp any eq echo any
permit tcp any any established (150 matches)
permit tcp any any (15 matches)
permit ip any any (45 matches)
```

The output is simple, but effective - note the high number of ICMP echo-reply packets. More information about suspect packets can be gathered by appending the "log-input" command to the proper rule. This rule will log information about any ICMP traffic:

```
access-list 169 permit icmp any any echo-reply log-input
```

The router logs now capture more detail (view using "show log") about the matching packets. In the example below, the log shows several packets that matched a DENY ICMP rule:

```
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.142 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.113 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.72 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.154 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.15 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.142 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.47 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
```

```
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.35 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.113 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.59 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.82 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.56 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.84 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.47 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.45.35 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 192.168.212.15 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
%SEC-6-IPACCESSLOGDP: list 169 denied icmp 172.16.132.33 (Serial0 *HDLC*) -> 10.2.3.7 (0/0), 1
packet
```

Note the information contained in each entry: source and destination address, interface and the rule that it matched. This type of detailed information will help determine our defense.

Reaction

Having looked at how to identify suspect traffic, it's time to examine how to respond. Unfortunately, the options are somewhat limited because most DDoS attacks used spoofed source addresses that are likely generated at random. So what be done?

Tracing the Attackers

One of the most common reactions would be an attempt to "trace" the attacker. However, a DDoS, unlike a traditional DoS, comes from multiple sources. The best chance at alleviating the attack using this method would be determining which of the routers several hops up from your network is handling the most packets. Unfortunately, this will require cooperation from several sources, since you won't be able to examine packets on an upstream router. Each participant in the process (mostly ISPs) will, however, follow very similar steps. Having identified the offending traffic type using the techniques above, a new, specific access list will be constructed to match it. Appended to the rules, which are applied to the interface sending traffic to the target, will again be the "log-input" keyword. Why? The logs will record details about the source interface and MAC address - which is valid information. That data can be used to determine the IP address of the router forwarding the malicious traffic. The process is then repeated on the next router up the chain. After several iterations, the

source (or one of them) will be located. At that time, the proper filter can be put in place to block the attacker. The drawback of tracing a DDoS attack is time and difficulty. Rooting out several sources could require working with multiple parties and even law enforcement.

Rate Limiting

A better option for immediate relief, one available to most ISPs, would be to "rate limit" the offending traffic type. Rate limiting restricts the amount of bandwidth a specific type of traffic can consume at any given moment. This is accomplished by dropping the limited packets received when the threshold is exceeded. It's useful when a specific packet is used in the attack. Cisco provides this example for limiting ICMP packets used in a flood:

```
interface xy
rate-limit output access-group 2020 3000000 512000 786000 conform-action
transmit exceed-action drop
access-list 2020 permit icmp any any echo-reply
```

This example brings up an interesting problem, which was noted earlier. What if the offending traffic appears to be completely legitimate? For instance, rate limiting a SYN flood directed at a Web server will reject both good and bad traffic, since all legitimate connections require the initial 3-way handshake of TCP. It's a difficult problem, without an easy answer. Such concerns make DDoS attacks extremely tricky to handle without making some compromises.

Black Hole Filtering

ISPs have other options available that depend on routing changes, such as black hole filtering. Black hole filtering works by forwarding malicious traffic to an imaginary interface known as Null0 – similar to /dev/null on Unix machines. Since it's not a valid interface, traffic routed to Null0 is essentially dropped. Moreover, this technique minimizes performance impact - a useful feature during the DDoS investigation so the rest of the network remains stable under the heavy load.

An important point to keep in mind when addressing a DDoS attack is that filtering at the target is not the best option. Whether it is a firewall or border router stopping the offending packets, a huge portion of incoming bandwidth is still being consumed - delaying legitimate traffic. To truly alleviate the effects of a DDoS flood, the traffic will have to be blocked at a point higher up the chain - likely a device under a large providers control. This means that many of the products that claim to prevent DDoS attacks are ultimately useless for smaller networks and their end users. Moreover, it means that solving a DDoS attack, at some point, is out of our hands. It's a frustrating truth realized by anyone who has ever dealt with the problem.

Prevention

If you are lucky to have escaped the wrath of DDoS attack to this point, make sure you employ the following precautions to keep it that way and prevent your own network from participating in one.

The "ip verify unicast reverse-path" (or non-Cisco equivalent) command should be enabled on the input interface of the upstream connection. This feature drops spoofed packets, a major difficulty in defeating DDoS attacks, before they can be routed. Additionally, make sure incoming traffic with source addresses from reserved ranges (i.e., 192.168.0.0) is blocked. This filter will drop packets whose sources are obviously incorrect .

Ingress and egress filtering techniques are also crucial to the prevention of DDoS attacks. These simple ACLs, if properly and consistently implemented by ISPs and large networks, could eliminate spoofed packets from reaching the Internet, greatly reducing the time involved in tracking down attacker. The filters, when placed on border routers, ensure that incoming traffic does not have a source address originating from the private network and more importantly, that outbound traffic does have an address originating from the internal network.

[RFC2267](#) is a great foundation for such filtering techniques.

Lastly, it's crucial to have discussed DDoS attacks and operational procedures with your IT teams and ISP. Time is of the essence under these unfortunate circumstances - a plan and contact list should be in place BEFORE it happens.

Conclusion

DDoS attacks are a difficult challenge for the Internet community. The reality of the situation is that only the biggest attacks are fully investigated, the smaller ones that happen everyday slip through the cracks. And while a bevy of products exist, most are not practical for smaller networks and providers. Ultimately, you are in charge of dealing with and defending against a DDoS. This means knowing how to respond when under attack: identifying traffic, designing and implementing filters, the follow-up investigation. Preparation and planning are, by far, the best methods for mitigating DDoS attacks and risks.

Matthew Tanase CISSP, is President of [Qaddisin](#), a network security company based in St. Louis. His company provides nationwide consulting services for several organizations. Additionally, he produces [The Security Blog](#), a daily Weblog dedicated to security.

Relevant Links

[Barbarians at the Gate: An Introduction to DDoS](#)

Matt Tanase, SecurityFocus

[Sniffers: What They Are and How to Protect Yourself](#)

Matt Tanase, SecurityFocus

[Cisco - Strategies to Protect Against DDoS Attacks](#)

Cisco White Papers

[Cisco - Characterizing and Tracing Packet Floods Using Cisco Routers](#)

Cisco White Papers

Cisco - IOS Essential Features

Cisco White Papers

ISP Security: Real World Techniques

The North American Network Operators' Group (NANOG)

David Dittrich's DDoS site

[Privacy Statement](#)

Copyright 2006, SecurityFocus