(Pre) GENI Lab 2

Due: There is nothing to submit. However, make sure you complete Section 1 below by 3:30pm on Sept 11, 2019

0. Introduction

0.1. Overview

This experiment explores **slowloris**, a denial of service attack running on the application layer that requires very little bandwidth and causes vulnerable web servers to stop accepting HTTP connections from other users.

This experiment involves running a potentially disruptive application over a private network, in a way that does not affect infrastructure outside of your slice. Take special care not to use this application in ways that may adversely affect other infrastructure.

0.2. Background

Denial-of-service (DoS) attacks aim to block access by "legitimate" users of a website or other Internet service, typically by exhausting the resources of the service (e.g. bandwidth, CPU, memory) or causing it to crash.

Slowloris is a type of denial of service attack that operates at the application layer. It exploits a design approach of many web servers, allowing a single machine to take down another machine's vulnerable web server with minimal bandwidth.

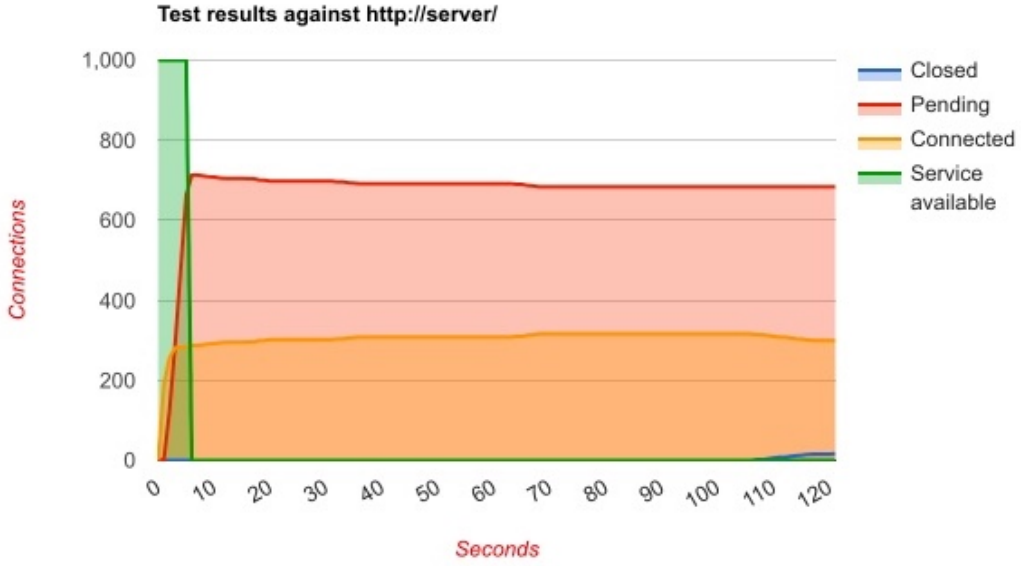
It achieves this by opening as many connections to the target web server as it can, and holding them open as long as possible by sending a partial request, and adding to it periodically (to keep the connection alive) but never completing it. Affected servers use threads to handle each concurrent connection, and have a limit on the total number of threads. Under slowloris attack, the pool of threads is consumed by the attacker and the service will deny connection attempts from legitimate users.

Slowloris was used in 2009 against Iranian government servers during protests related to the elections that year.

0.3. Objectives/Expected Results

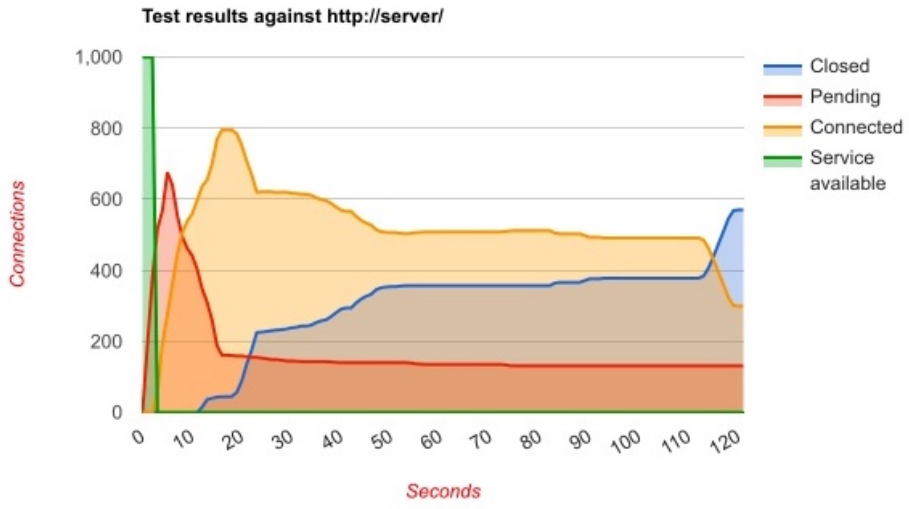
The following image shows the response of an Apache web server to a slowloris attack. We see that when there are a large number of established connections, the service becomes unavailable (green line goes to zero.)

Test parameters	
Test type	SLOW HEADERS
Number of connections	1000
Verb	GET
Content-Length header value	4096
Extra data max length	52
Interval between follow up data	10 seconds
Connections per seconds	200
Timeout for probe connection	3
Target test duration	120 seconds
Using proxy	no proxy



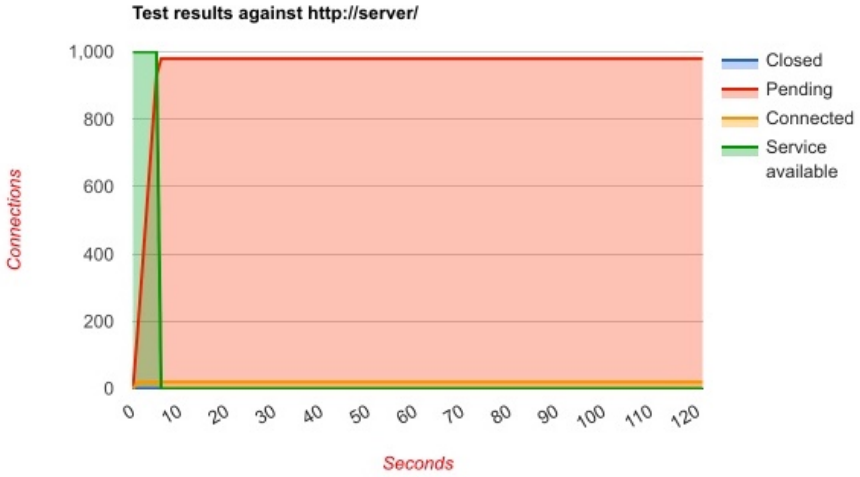
When we limit the rate of traffic from the attacker to 100 kbps, the attack is still successful:

Test parameters	
Test type	SLOW HEADERS
Number of connections	1000
Verb	GET
Content-Length header value	4096
Extra data max length	52
Interval between follow up data	10 seconds
Connections per seconds	200
Timeout for probe connection	3
Target test duration	120 seconds
Using proxy	no proxy



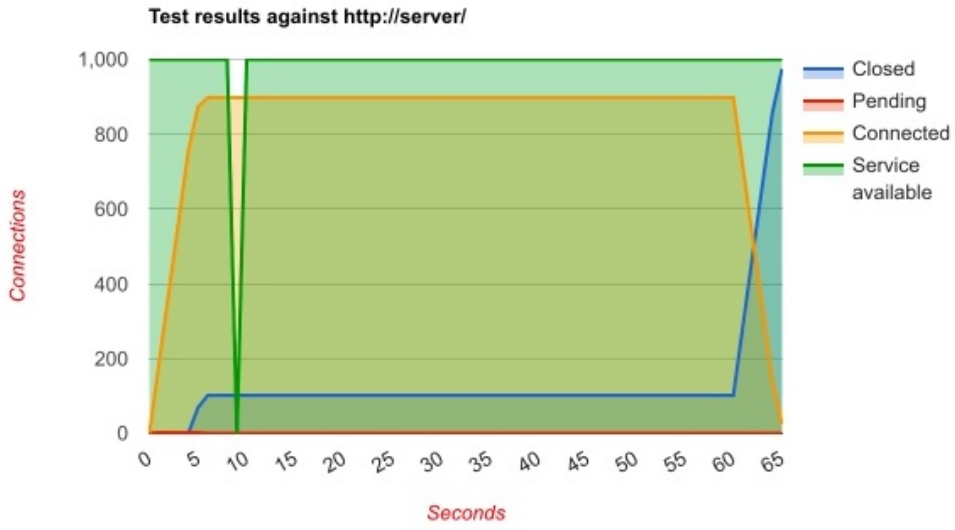
Using a firewall to limit the number of connections from a single host is more successful. While slowhttptest still reports that the service is unavailable, in fact, it is only unavailable to the malicious attacker (which we can see is limited to 20 connections) and other hosts are able to access the service:

Test parameters	
Test type	SLOW HEADERS
Number of connections	1000
Verb	GET
Content-Length header value	4096
Extra data max length	52
Interval between follow up data	10 seconds
Connections per seconds	200
Timeout for probe connection	3
Target test duration	120 seconds
Using proxy	no proxy



Finally, we found that the nginx web server is resistant to slowloris (even without a firewall limiting the number of connections per host) because of its non-blocking approach, which supports a higher level of concurrency:

Test parameters	
Test type	SLOW HEADERS
Number of connections	1000
Verb	GET
Content-Length header value	4096
Extra data max length	52
Interval between follow up data	10 seconds
Connections per seconds	200
Timeout for probe connection	3
Target test duration	120 seconds
Using proxy	no proxy



0.4. Key Commands

`ifconfig` , `slowhttptest` , `lynx` , `tc` , and `nginx`

1. Lab Configurations

1.1. New Slice

Create a new slice (please name the new slice "**lab2-your-initial**") and request resources by loading the RSpec from the following URL: https://git.io/v9ftJ.

Be sure you have one ssh terminal to the client node and one ssh terminal to the server node.

1.2. File Transfer with SCP (Secure Copy)

As what you will see later, you will need to transfer some files from your GENI nodes to your local computer with the SCP protocol (a.k.a. sftp on Mac/Linux or WinSCP on Windows).

1.2.1. Use sftp on Mac/Linux

- On your local computer, create and go to a folder (e.g. CS4121/GENI/Lab2) where you want to store files for this lab.
- In the ssh terminal to your **client** node, create a dummy file `cat > test.txt`, type some random text, and press Ctrl-d to quit.
- Then follow the instructions to use sftp on Mac/Linux here.
    - In step 4, if you got some permission error, your might have to add the `-i ~/.ssh/id_geni_ssh_rsa` option to the sftp command.
    - In step 6, do `> mget test.txt`.
- On your local computer, check if the dummy file has been successfully transferred. If yes, you are all set.

1.2.2. Use WinSCP on Windows

- (If you are using one of the lab computers in Nevins Hall, skip this step because WinSCP has already been installed on it.)

  On your personal Windows laptop, download WinSCP from the following link.

- (If you already downloaded the PuTTY key to your computer from GENI, skip this step.)

  Download the PuTTY key to your computer from the GENI portal as what you did in lab 0.

- On your local computer, create a folder (e.g. CS4121/GENI/Lab2) where you want to store files for this lab.

- In the ssh terminal to your **client** node, create a dummy file `cat > test.txt`, type some random text, and press Ctrl-d to quit.

- Then follow the instructions to use WinSCP on Windows here

    - In the last step, if you don't see your dummy file in the right area, right-click and refresh.

      Drag and drop the dummy file to the local folder you designated to this lab in the left area.

- On your local computer, check if the dummy file has been successfully transferred. If yes, you are all set.

TO BE CONTINUED....