Layer 7 DoS attack with slowloris

CPRE 431

M04 Lab HW

Assignments will be submitted in PDF format via Canvas.

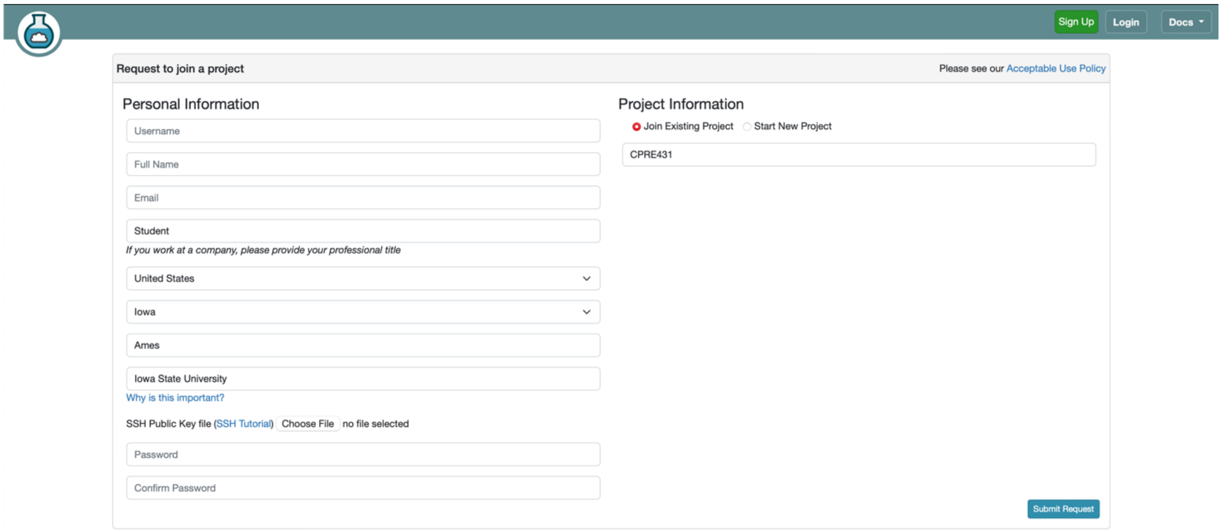
Please submit your homework online through Canvas. Late homework will not be accepted.
Important: Your submission must be in .pdf format ONLY! Make sure to read every instruction carefully as misstep can affect results in successive steps
Please ensure that you support all your answers with the correct screenshots of the major steps (reserving resources on CloudLab, SSHing to the nodes, installing tools/packages, any plots,... etc).

Introduction

This experiment explores slowloris, a denial of service attack that requires very little bandwidth and causes vulnerable web servers to stop accepting connections to other users. This experiment should take about 60 minutes to run.

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. Users of CloudLab are responsible for ensuring compliance with the CloubLab Acceptable Use Policy

To reproduce this experiment on CloubLab, you will need an account on the CloudLab Portal, and you will need to join a project. For the purposes of this class, the project is called **CPRE431**. Make sure to follow the join link on Canvas to create an account and automatically submit a request to join the CPRE431 project. When creating your account, fill in the username, full name, email, and password boxes for your CloudLab account and the rest of the boxes can be filled as seen below.



Request to join a project

Please see our Acceptable Use Policy

Personal Information

Username

Full Name

Email

Student

If you work at a company, please provide your professional title

United States

Iowa

Ames

Iowa State University

Why is this important?

SSH Public Key file (SSH Tutorial) Choose File no file selected

Password

Confirm Password

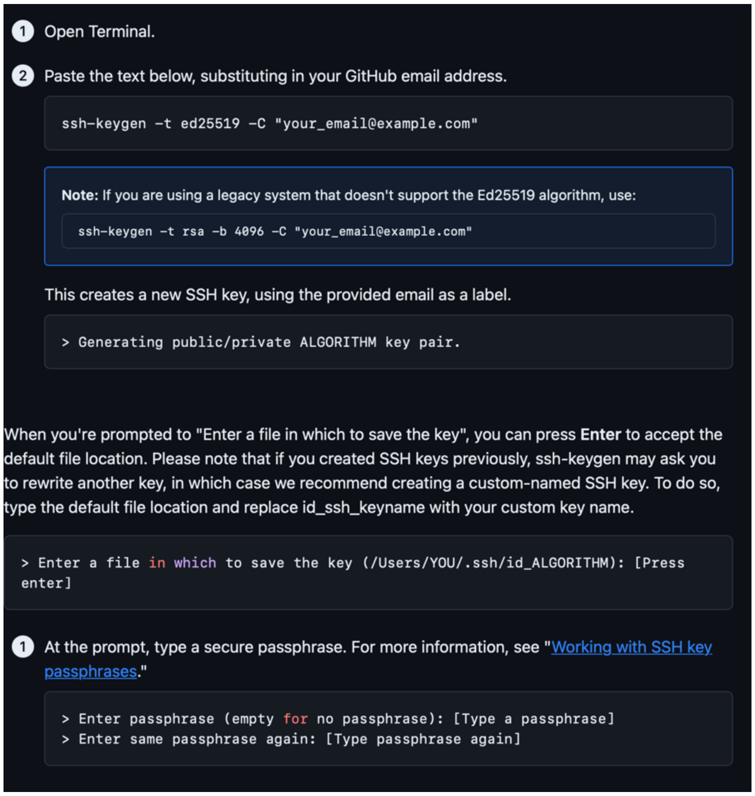
Project Information

● Join Existing Project ○ Start New Project

CPRE431

Submit Request

CloudLab requires uploading generated SSH keys. Uploading the SSH key makes it easier to connect to the experiment's nodes and execute the steps on multiple nodes seamlessly without using the browser console. To generate an SSH key, follow the GitHub Tutorial here to create an SSH key-pair. Essentially, the main instructions for key generation are in Figure 2. Once the keys are generated, upload your public key to CloudLab on the account creation portal and submit the join request. Once the request is approved, you can start setting up your nodes and experiment.



1 Open Terminal.

2 Paste the text below, substituting in your GitHub email address.

```
ssh-keygen -t ed25519 -C "your_email@example.com"
```

Note: If you are using a legacy system that doesn't support the Ed25519 algorithm, use:

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

This creates a new SSH key, using the provided email as a label.

```
> Generating public/private ALGORITHM key pair.
```

When you're prompted to "Enter a file in which to save the key", you can press **Enter** to accept the default file location. Please note that if you created SSH keys previously, ssh-keygen may ask you to rewrite another key, in which case we recommend creating a custom-named SSH key. To do so, type the default file location and replace id_ssh_keyname with your custom key name.

```
> Enter a file in which to save the key (/Users/YOU/.ssh/id_ALGORITHM): [Press
enter]
```

1 At the prompt, type a secure passphrase. For more information, see "Working with SSH key passphrases."

```
> Enter passphrase (empty for no passphrase): [Type a passphrase]
> Enter same passphrase again: [Type passphrase again]
```

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.

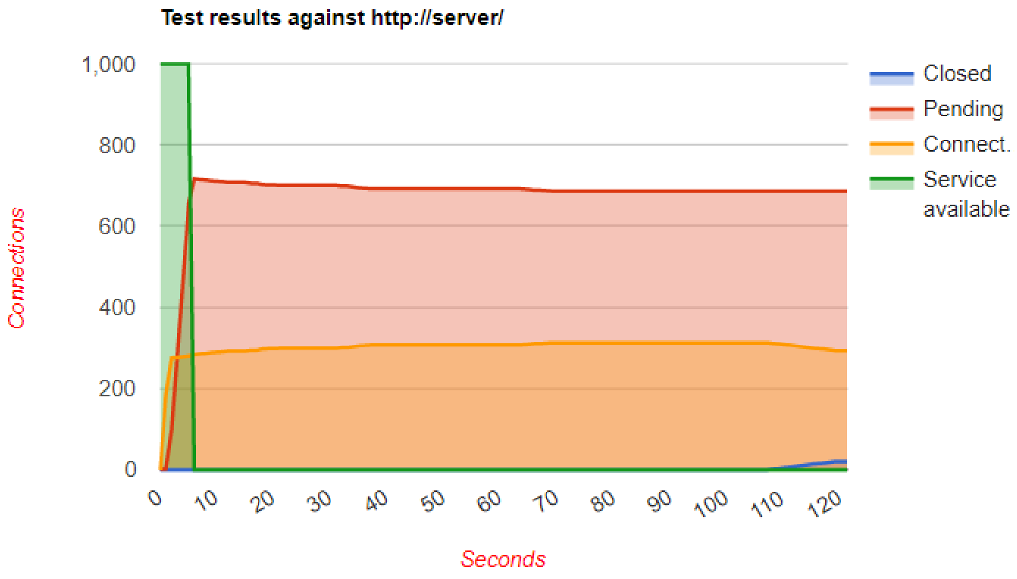
Slowloris is a type of denial of service attack that operates at Layer 7 (the application layer). It exploits a design approach of many web servers, allowing a single machine to take down another machine's vulnerable web server with minimal bandwidth.

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.

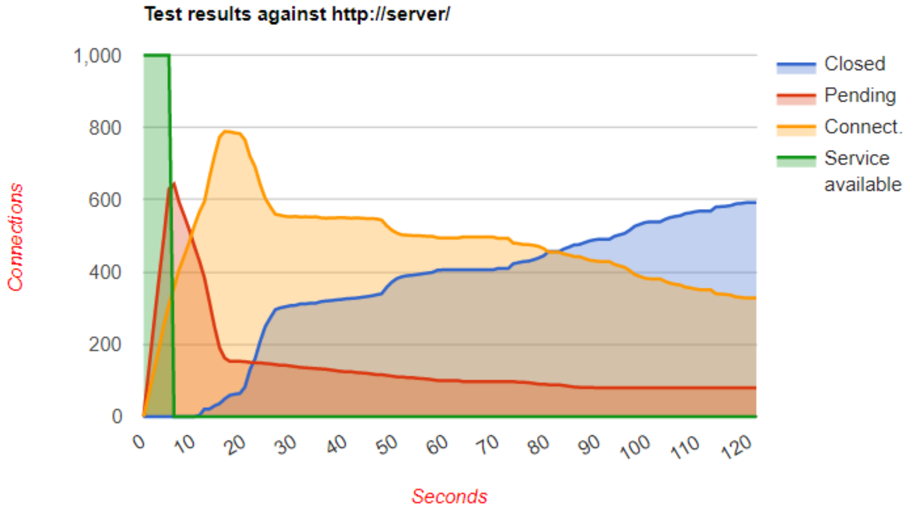
Slowloris was used in 2009 against Iranian government servers during protests related to the elections that year.

Results

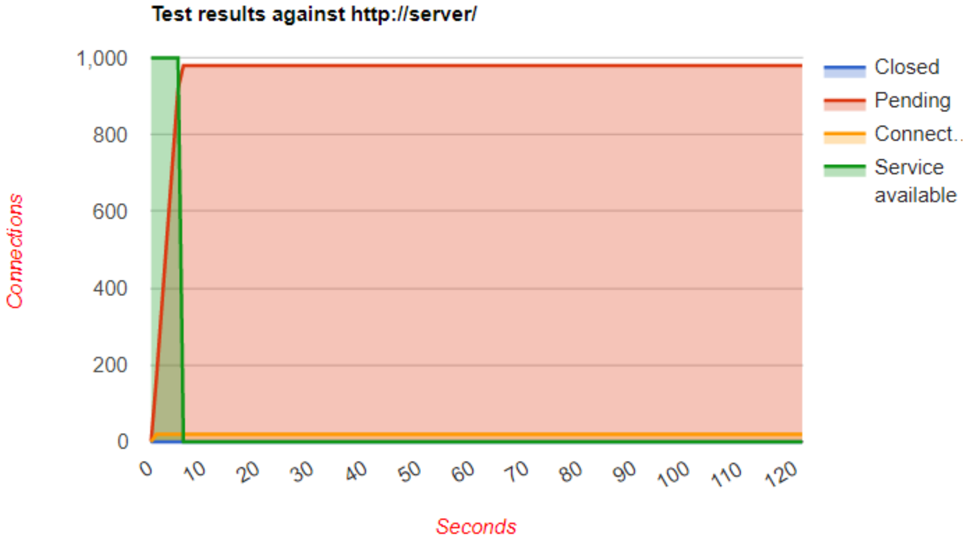
The following image shows the response of an Apache webserver to a slowloris attack. We see that when there are a large number of established connections, the service becomes unavailable (green line goes to zero.)



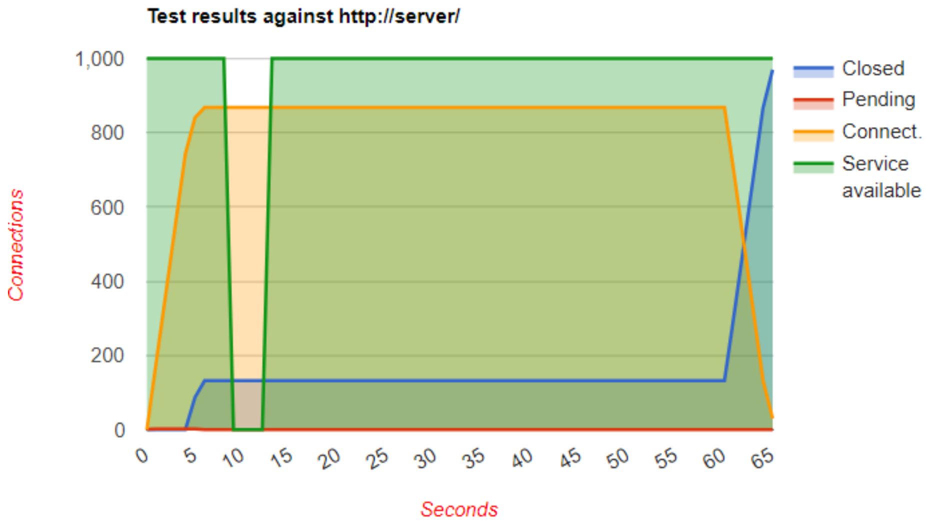
When we limit the rate of traffic from the attacker to 100 kbps, the attack is still successful:



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:

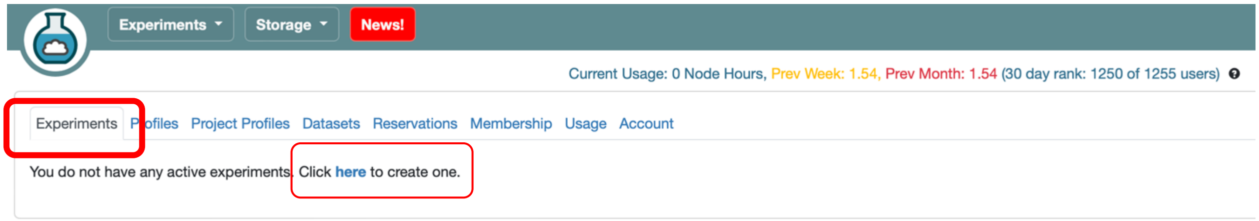


Finally, we found that the nginx web server is resistant to slowloris (even without a firewall limiting the number of connections per host) because of its non-blocking approach, which supports a higher level of concurrency:



Running the Experiment

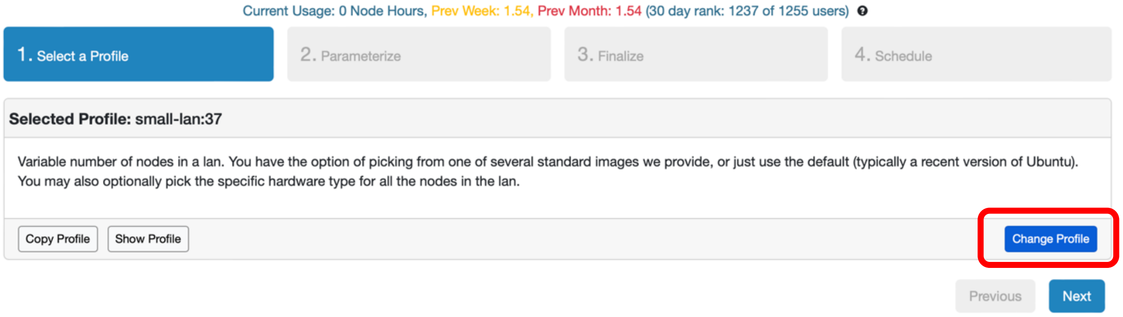
In the CloudLab Portal, go to the “Experiments” tab and click on create a new experiment.



Current Usage: 0 Node Hours, Prev Week: 1.54, Prev Month: 1.54 (30 day rank: 1250 of 1255 users)

You do not have any active experiments. Click here to create one.

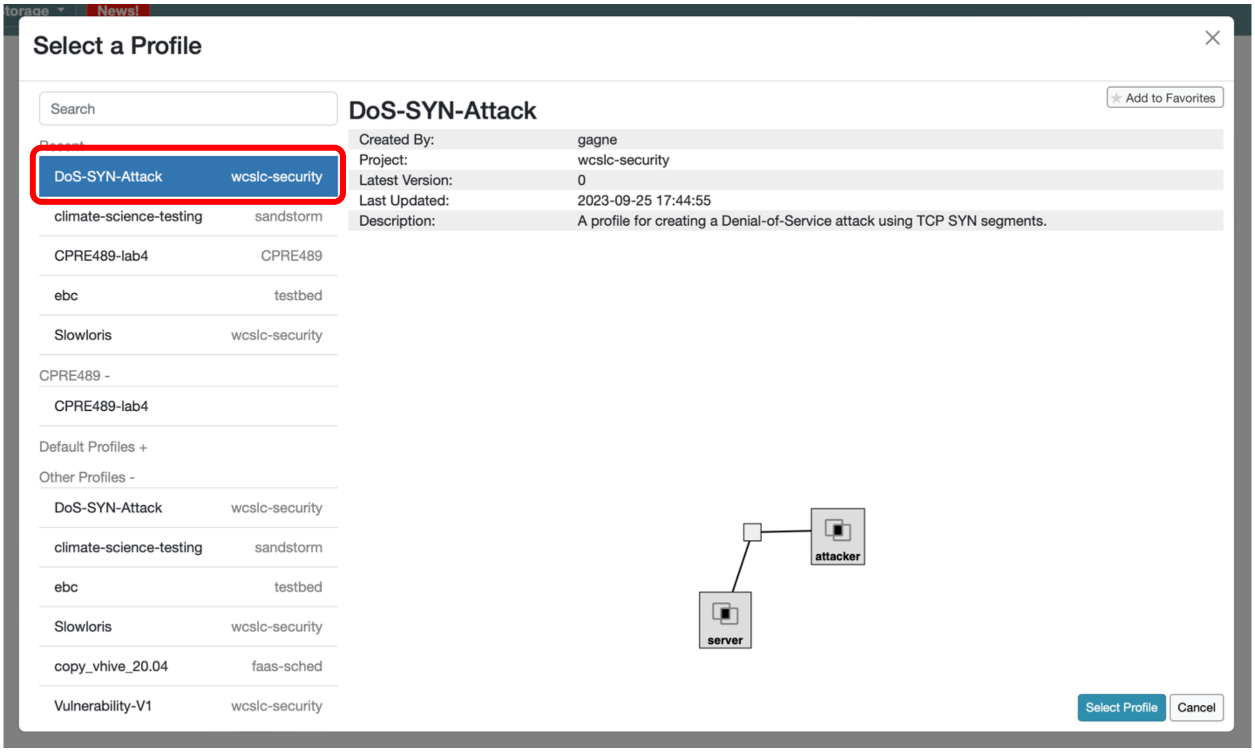
Next, click on “Change Profile” and pick “DoS-SYN-Attack.” If you cannot find it on the suggested profiles list, search for the name in the search bar and it should pop up. Click “Select Profile” and click “Next.



Current Usage: 0 Node Hours, Prev Week: 1.54, Prev Month: 1.54 (30 day rank: 1237 of 1255 users)

Selected Profile: small-lan:37

Variable number of nodes in a lan. You have the option of picking from one of several standard images we provide, or just use the default (typically a recent version of Ubuntu). You may also optionally pick the specific hardware type for all the nodes in the lan.



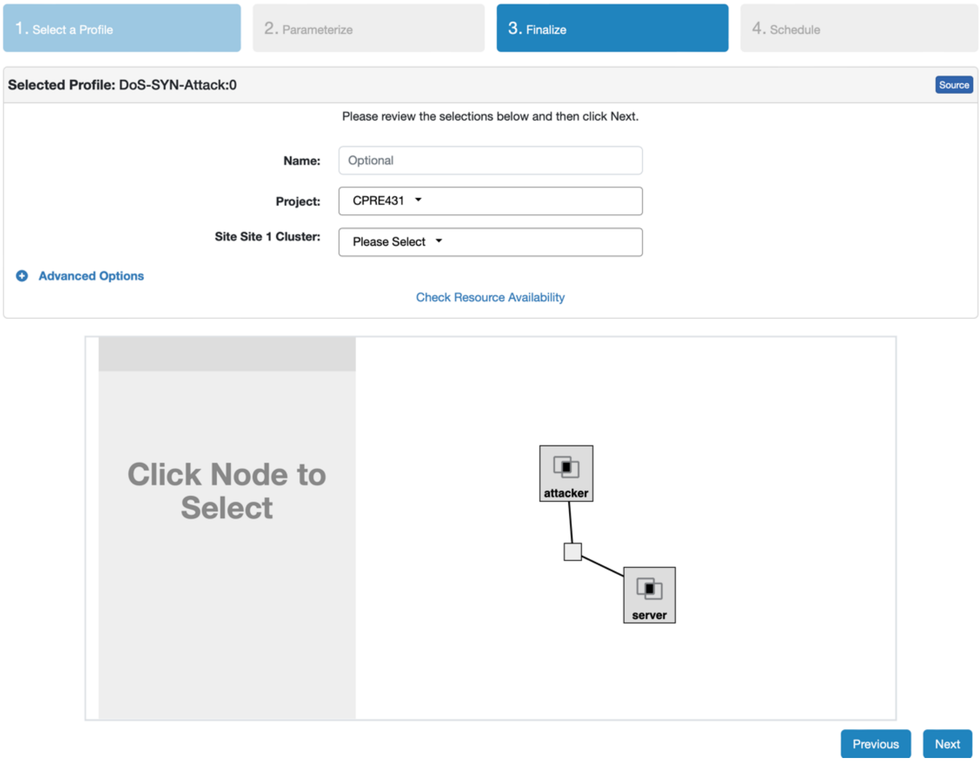
Select a Profile

DoS-SYN-Attack	wcslc-security
climate-science-testing	sandstorm
CPRE489-lab4	CPRE489
ebc	testbed
Slowloris	wcslc-security
CPRE489-lab4	
DoS-SYN-Attack	wcslc-security
climate-science-testing	sandstorm
ebc	testbed
Slowloris	wcslc-security
copy_vhive_20.04	faas-sched
Vulnerability-V1	wcslc-security

DoS-SYN-Attack

Created By:	gagne
Project:	wcslc-security
Latest Version:	0
Last Updated:	2023-09-25 17:44:55
Description:	A profile for creating a Denial-of-Service attack using TCP SYN segments.

For the name option in the next screen, name your experiment in the following format: “firstName-lastName-CPRE431-LAB4” and when picking the site clusters, pick any of the resources which have a green check mark nearby their names.



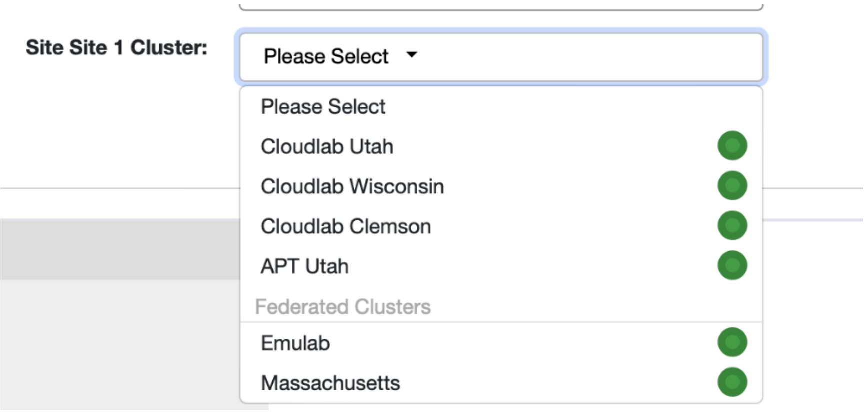
Selected Profile: DoS-SYN-Attack:0

Please review the selections below and then click Next.

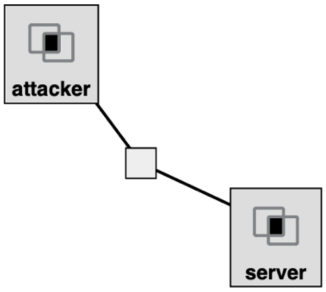
Name: Optional

Project: CPRE431

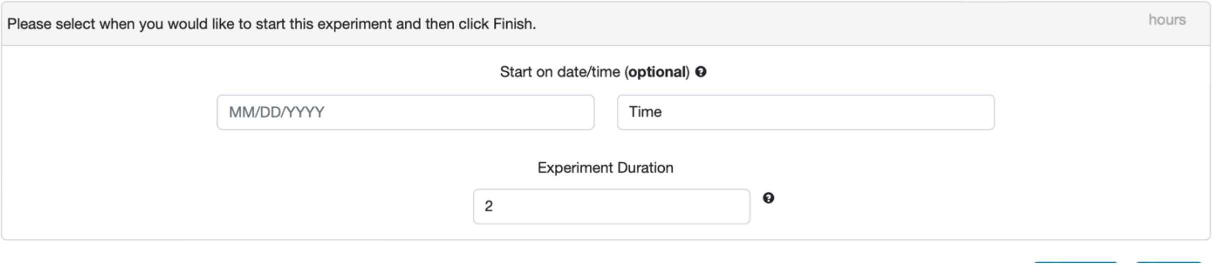
Site Site 1 Cluster: Please Select



This will load a topology with two nodes connected by a link, like this:



Schedule a time for your experiment, it should not take longer than an hour, however it is important to note that there are limited resources, so make sure to start early on your experiment. Include a screenshot in your report once the resources are reserved.



Please select when you would like to start this experiment and then click Finish.

Start on date/time (**optional**)

MM/DD/YYYY

Time

Experiment Duration: 2 hours

When your nodes are ready to log in, click on the Experiments tab again, and select your experiment. Once you are in your experiment page, click list view to show the SSH commands. Important Note: add the red portion to the end of the command you copy to SSH into the nodes (this is an example format of the expected SSH command, yours may look different)

```
ssh -p 12345 username@host.cloudlab.us -i /private/key/location/name_of_key
```

SSH into the Server Node and run to install the Apache webserver (and Lynx, a text-based web browser for use in terminal sessions).

```
sudo apt-get update
sudo apt-get -y install lynx apache2
```

Verify that the webserver is running by connecting to it from a browser; run

```
lynx http://server
```

on the server node and you should see the Apache2 Ubuntu Default Page.

In a second terminal, SSH into the client node and run

```
sudo apt-get update
sudo apt-get -y install slowhttptest
```

to install the slowhttptest tool. This tool implements several Layer 7 DoS attacks, including slowloris.

Then, on the client, run

```
slowhttptest -c 1000 -H -g -o apache_no_mitigation -i 10 -r 200 -t GET -u
http://server -x 24 -p 3 -l 120
```

In the terminal output, you will see the test parameters, e.g.

```
test type:                          SLOW HEADERS
number of connections:              1000
URL:                                http://server/
verb:                               GET
Content-Length header value:        4096
follow up data max size:            52
interval between follow up data:    10 seconds
connections per seconds:            200
probe connection timeout:           3 seconds
test duration:                      120 seconds
using proxy:                        no proxy
```

and you'll also see the current connections and their states, as well as the availability of the server. The message

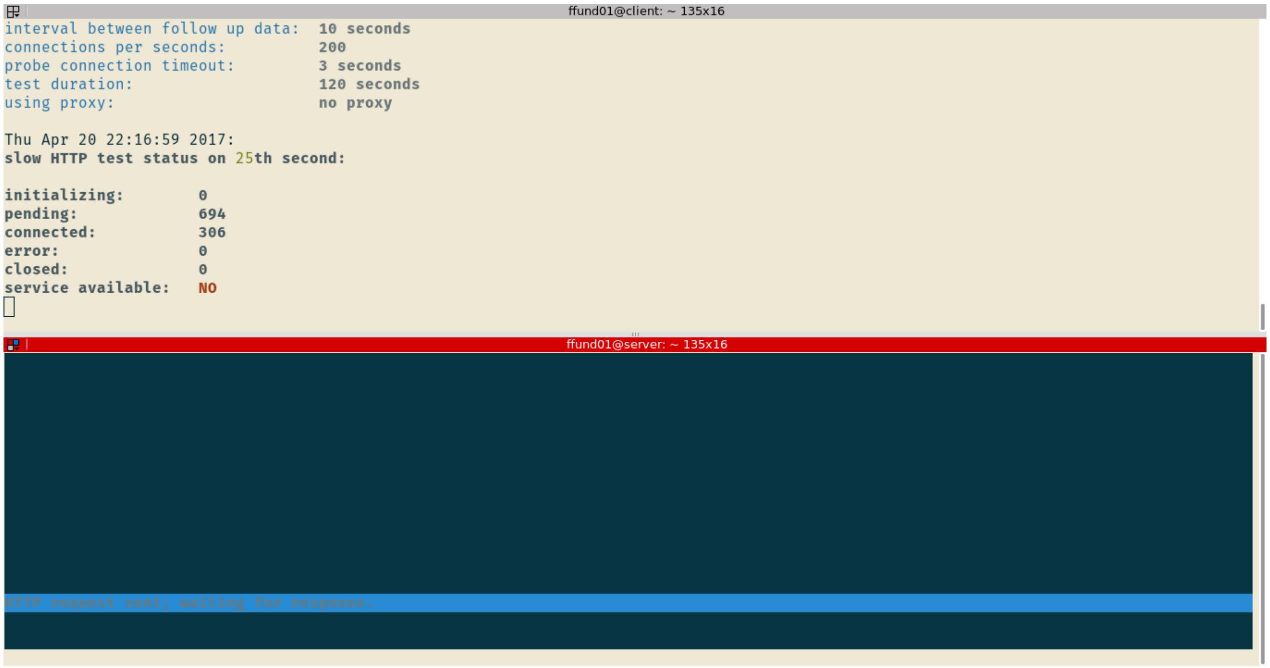
```
service available:   NO
```

means that the DoS attack on the webserver was successful.

This test will run for 120 seconds. After about half a minute, while the test is still running, try to access the web page again on the server node by running

```
lynx http://server
```

and verify that it is not responsive:



```
interval between follow up data:  10 seconds
connections per seconds:          200
probe connection timeout:         3 seconds
test duration:                    120 seconds
using proxy:                      no proxy

Thu Apr 20 22:16:59 2017:
slow HTTP test status on 25th second:

initializing:        0
pending:             694
connected:           306
error:               0
closed:              0
service available:   NO
```

```
HTTP request sent; waiting for response.
```

Also, if you run

```
netstat -anp | grep :80 | grep ESTABLISHED
```

on the server, you will see many TCP connections to port 80 in the ESTABLISHED state, a hallmark of this kind of attack.

After the test finishes running, transfer the "apache_no_mitigation.html" to your laptop with scp.

Please refer to the following video for how to SCP: https://iastate.box.com/s/0d5ohi6awzcracupthtn9c2qeihs1jr8

Open this file with a web browser. You should see an image similar to the first one in the Results section, indicating that a large number of established connections have made the service unavailable.

Let us explore several ways to mitigate this kind of attack.

First, let's see if this attack is still feasible when the client has very limited bandwidth. On the client node, run

```
ifconfig
```

and find the name of the network interface that is connected to the server. Then, run

```
sudo tc qdisc replace dev eth1 root netem rate 100kbit
```

substituting the name of the interface you have found in the previous step for eth1. This will limit the rate of outgoing traffic on this interface to 100 kbps.

Now we'll run the slowloris attack again. On the client, run

```
slowhttptest -c 1000 -H -g -o apache_lowrate_client -i 10 -r 200 -t GET -u
http://server -x 24 -p 3 -l 120
```

Wait about 30 seconds and then check the service availability by running lynx again on the server. When the test finishes (after 120 seconds), transfer the "apache_lowrate_client.html" file to your laptop with SCP and open it in a browser. Open this file with a web browser. You should see an image similar to the second one in the Results section, indicating that even when the attacker has very little available bandwidth, the attack can still be successful.

Remove the rate-limiting traffic shaper on the client with

```
sudo tc qdisc delete dev eth1 root
```

substituting the correct interface name in the command above.

Next, we will try using firewall rules to mitigate this attack. Specifically, we will create a rule that says that any single host is limited to 20 connections to port 80 on the server.

On the server, run

```
sudo iptables -I INPUT -p tcp --dport 80 -m connlimit --connlimit-above 20 --
connlimit-mask 40 -j DROP
```

to set up this rule.

On the client, run

```
slowhttptest -c 1000 -H -g -o apache_iptables -i 10 -r 200 -t GET -u
http://server -x 24 -p 3 -l 120
```

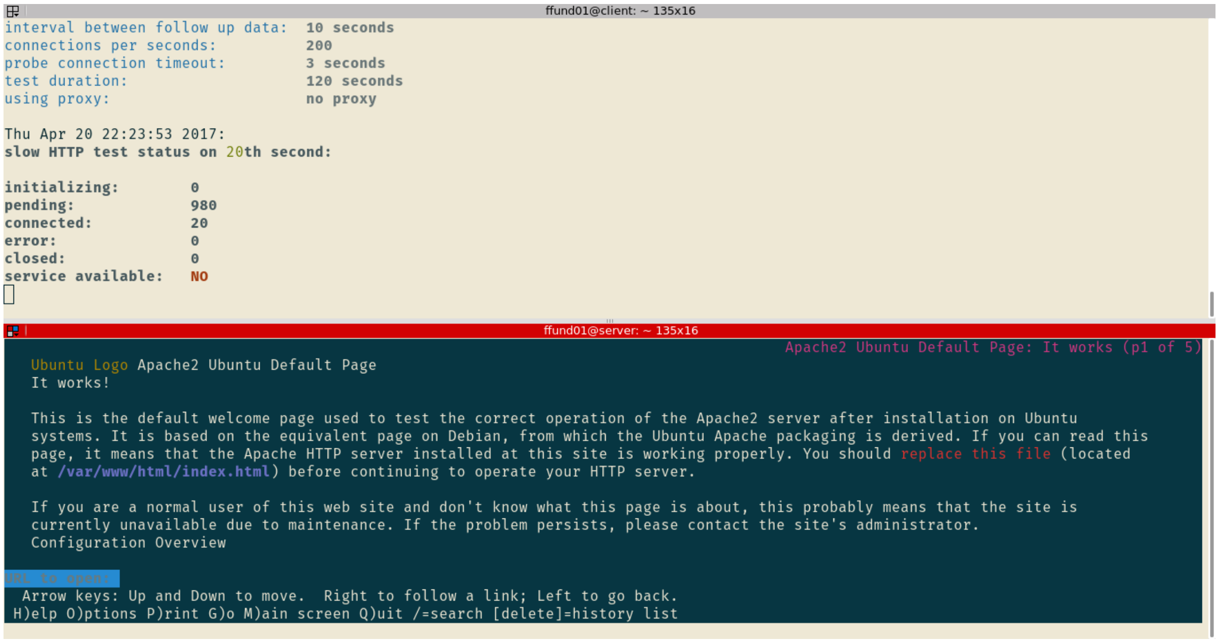
and then, after half a minute, run

```
lynx http://server
```

on the server to check the availability of the service. Even when slowhttptest reports

```
service available:   NO
```

we can still load the page in lynx on the server:



```
interval between follow up data:  10 seconds
connections per seconds:          200
probe connection timeout:         3 seconds
test duration:                    120 seconds
using proxy:                      no proxy

Thu Apr 20 22:23:53 2017:
slow HTTP test status on 20th second:

initializing:        0
pending:             980
connected:           20
error:               0
closed:              0
service available:   NO
```

```
                                                                          Apache2 Ubuntu Default Page: It works (p1 of 5)
   Ubuntu Logo Apache2 Ubuntu Default Page
   It works!

   This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu
   systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this
   page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located
   at /var/www/html/index.html) before continuing to operate your HTTP server.

   If you are a normal user of this web site and don't know what this page is about, this probably means that the site is
   currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
   Configuration Overview

URL to open:
  Arrow keys: Up and Down to move.  Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

This is because the service is only unavailable to the malicious user. The firewall does not affect a non-malicious user.

Transfer the file "apache_iptables.html" to your laptop with SCP and open it in a browser. Compare it to the third figure in the Results section.

While this mitigation prevents a slowloris attack that is launched from only one host, it still would not protect against a distributed slowloris attack, with many participants each consuming a smaller number of connections. Also, if the number of allowed connections per host is set too low, it might limit connections from clients behind a NAT or a proxy, which share the same IP address. Use:

```
sudo iptables --flush
```

to remove the firewall rule on the server.

We are going to try one more way to mitigate this attack: changing the application design.

The Apache web server allocates a worker thread for each connection, allowing a slow or idle connection to block an entire thread. When the total number of worker threads is exhausted, then no new connection is accepted.

In contrast, the nginx web server has a non-blocking design, in which worker threads are not assigned to connections on a one-to-one basis. Instead, a thread will dynamically serve a connection only when there is data to send or receive for that connection. This makes it more resistant to the slowloris attack at Layer 7 (although it may still be possible to launch a low-rate attack that exhausts the total number of connections possible at a lower level, such as the total number of file descriptors available to the operating system.)

On the server node, stop the Apache server with

```
sudo service apache2 stop
```

Then install and start nginx:

```
sudo apt-get update
sudo apt-get -y install nginx
sudo service nginx restart
```

If you still have issues stopping apache2 try to delete the resources and reserve them again (don't forget to install slowhttptest again on the client side) then install and start nginx as follows:

```
sudo apt-get -y install lynx nginx
```

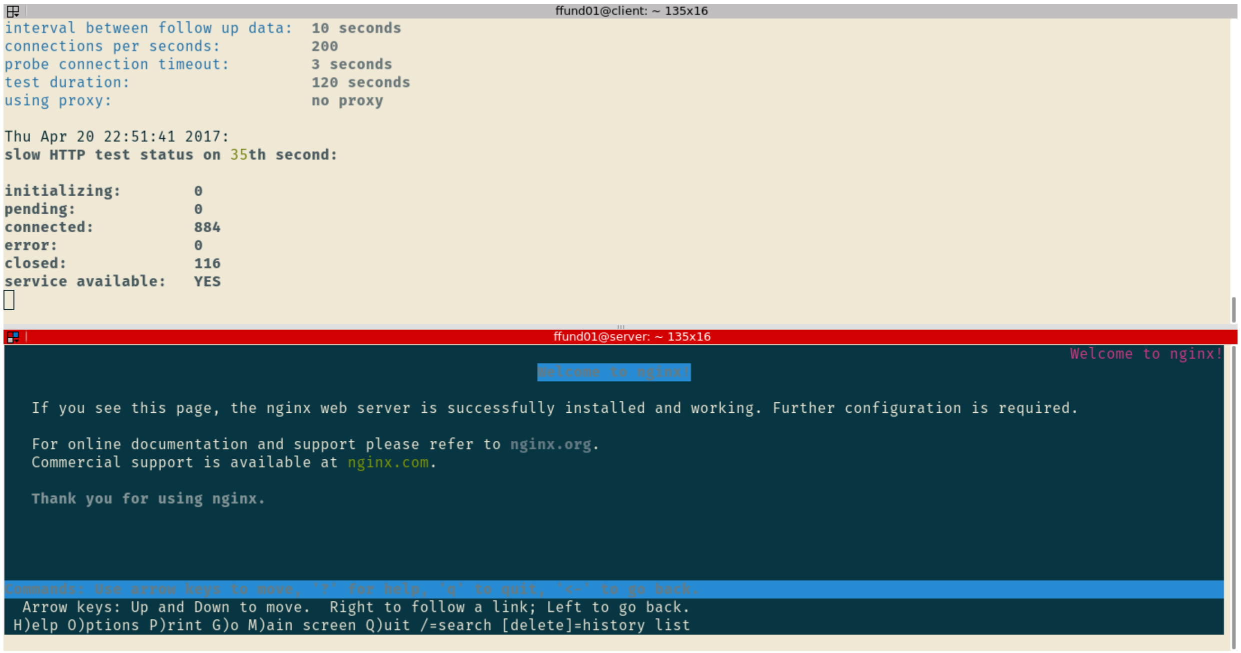
Run the attack from the client again, with

```
slowhttptest -c 1000 -H -g -o nginx_no_mitigation -i 10 -r 200 -t GET -u
http://server -x 24 -p 3 -l 120
```

and after 30 seconds, run

```
lynx http://server
```

on the server. Here, you should see a "Welcome to nginx" page:



ffund01@client: ~ 135x16

```
interval between follow up data:  10 seconds
connections per seconds:          200
probe connection timeout:         3 seconds
test duration:                    120 seconds
using proxy:                      no proxy

Thu Apr 20 22:51:41 2017:
slow HTTP test status on 35th second:

initializing:        0
pending:             0
connected:           884
error:               0
closed:              116
service available:   YES
```

ffund01@server: ~ 135x16

```
                                                                                                                      Welcome to nginx!
                                                          Welcome to nginx!

   If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

   For online documentation and support please refer to nginx.org.
   Commercial support is available at nginx.com.

   Thank you for using nginx.



Commands: Use arrow keys to move, '?' for help, 'q' to quit, '<-' to go back.
  Arrow keys: Up and Down to move.  Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

When the attack finishes, transfer the "nginx_no_mitigation.html" file to your laptop with SCP and open it in a browser. While you may see some brief outage, you should find that the service generally remains available (even to the malicious attacker) despite a large number of established connections (as in the fourth figure in the Results section.) Due to the difference in application design, this web server is less vulnerable to the slowloris attack.

Please terminate your resources in the CloudLab Portal when you're done, to free them up for other experimenters!