# CCTF: Resilient Server - USC/ISI

## Created by: Jelena Mirkovic, USC/ISI, sunshine@isi.edu

## Contents



## Teams

| Role | Sharks | Lions | Tigers | Rockets | Jets |
|---|---|---|---|---|---|
| **Students** | Garima Aggarwal<br>David Collier<br>David Ivan<br>Jingul Kim<br>Christian Vanderwall | Kristine Brown<br>Adrian Cagaanan<br>Andrew Lee<br>George Li<br>Brian Nguyen | Wai Hung Cheung<br>Keith DeRuiter<br>Phillip Lee<br>Juanchen Li<br>Joseph Nitz | James Dalton<br>Cristopher Hanford<br>Yunan Lin<br>Javis McGee<br>Angelica Tran | Johnson Hsieh<br>Danbo Lai<br>Jefferson Luu<br>Runxuan Wei<br>Hanliang Xu |
| **Mentor** | Xiyue Deng | Abdulla Alwabel | Vinod Sharma | Hao Shi | Simon Woo |

## Overview

This exercise lets students practice monitoring network for denial-of-service attacks and devising appropriate actions. Students will be divided into 3-4 person teams. Each team will play the defender role (Blue team) for their own system and the attacker role (Red team) for another team's system. Each network in the exercise will consist of six machines - a `server` and a `gateway` machine under the control of the Blue Team, three `client` machines under the control of the Red Team and a `router` machine that neither team controls. Links between clients and the `router` and the link between the `gateway` and the `server` are 100 Mbps. The link between the `gateway` and the `router` is 1 Gbps. This allows the clients to attempt to create DoS conditions by flooding the link between the `gateway` and the`server`. The network for the exercise is shown below and the NS file for it resides at `/share/education/CTF2_USC/ctf2.ns`.

## Blue Team Tasks

This team will control the `server` and the `gateway` machine, connected by 100 Mbps link. The `server` should be a classical LAMP server (you can create it by typing:

```
sudo apt-get update
sudo apt-get install lamp-server^
```

The server should be able to serve 10 static Web pages, whose names should be:

```
1.html
2.html
3.html
4.html
5.html
6.html
7.html
8.html
9.html
10.html
```

The Blue team can make up the content on these pages.

The Blue team should also develop a monitoring program for the `gateway` machine and for the `server` so that they can quickly spot if the Red team launches denial-of-service and so that they can defend from it. One way to defend from it is to implement some filtering at the `gateway` machine via iptables. Blue tam should be ready to implement filtering on the fly but they should develop a sophisticated monitoring program that helps them quickly figure out what to filter. The Blue team can also modify Apache's configuration and code to make the server more resilient to some attacks, or they can craft packets to end connections that behave suspiciously (e.g., TCP RST).

Make sure you understand how `iptables` command works before you use it as you may cut off your access to a given machine in DeterLab if you filter out some specific traffic to/from it, e.g., all outgoing traffic. The only way to recover from this is to reboot the machine using Web portal for DeterLab. Click on your Experiment, then click on the machine's name in the Node List (e.g., pc133) and then choose "Reboot node" from the top left menu. It usually takes 5-10 minutes for the machine to come up again.

The goal of the Blue team is to keep the server up and running. When the server gets attacked, the Blue team should strive to bring it back up quickly (if it is down) and to install filters to get rid of attack traffic.
When developing and practicing swap in an experiment using `/share/education/CTF2_USC/ctf2.ns` file. This will lead to the identical setup as the one during CTF2 exercise.

## Assumptions and Requirements

You can borrow code from online sources but you need to understand what it does and how.

## Milestones

Here are some milestones that your team must reach BEFORE the exercise.

1. **(4/6)** Develop monitogring at the `server` that will let you automatically check the content of

HTTP requests you are getting and who is sending them.
2. **(4/6)** Develop monitoring software on the `gateway` machine that will let you automatically check if `server` is getting slow.
3. **(4/10)** Extend your monitoring software so you can automatically get statistics on number of packets and bytes sent to the server in TCP data, TCP SYN, UDP and ICMP and Total categories so you can diagnose various DDoS attacks. Make sure the software monitors the correct interface.
4. **(4/16)** Extend your monitoring software so you can detect number of packets and bytes sent to the server by each client IP. Make sure the software monitors the correct interface.
5. **(4/10)** Learn how you would write rules for `iptables` to filter traffic with some characteristics, e.g., by protocol, sender IP, length, TCP flags, etc. You may need to write those rules manually during the exercise but make sure you have tried to write them while preparing for the exercise and that they work correctly. You can check correctness by generating attack traffic with some signature (e.g., packet length, sender IP, protocol, etc.), writing a rule to filter it and checking that that traffic is dropped. You can check for drops in two ways. First, you could run your monitoring software on the interface leading to the `server`. Second, you could use an option with `iptables` that lets you see counts of times a rule was matched. It may be advisable to try both methods for measuring correctness as the first measures what goes to the server and the second shows you that the rule was activated by attack traffic.

Tasks 1 and 2 can be done in parallel, tasks 3 and 4 should be done after 1 and 2. Task 5 can be done in parallel with all this.

# Red Team Tasks

The Red Team will have control over the three client machines. They should program one of those to send only legitimate traffic, and two can be used for various attacks. *Only the Red Team will know which machine is legitimate*. The legitimate client machine must:

- Run an automated program to generate requests
- Generate at most 1 request per second
- Generate at least 1 request per 10 seconds

Attack machines can act as legitimate machines some times and attack other times.

The goal of the Red Team is to make the server unable to serve its legitimate client, either through compromise or through denial of service. Any attack is allowed, not just denial of service, even breaking Blue team's passwords.
When developing and practicing swap in an experiment using `/share/education/CTF2_USC/ctf2.ns` file. This will lead to the identical setup as the one during CTF2 exercise.

### Assumptions and Requirements

You can borrow code from online sources but you need to understand what it does and how.

### Milestones

Here are some milestones that your team must reach BEFORE the exercise.

1. **(4/10)** Develop attacks that may crash the `server` because they require it to process too many requests or because requests are malformed. This may or may not be possible but give it a try.
2. **(4/10)** Develop attacks that flood the link between the `gateway` and the `server`. It may be advisable to use raw sockets here to craft packets. It may also be advisable to parameterize attack software so that you can easily change spoofing technique, if any, packet type, packet length, etc.
3. **(4/16)** Develop attacks that flood the link or the server with too many HTTP requests.
4. **(4/16)** Develop attacks that use slow HTTP flood.
5. **(4/18)** Test ALL your attacks and make sure they do work against your server implementation. Then iterate between trying to handle those that work against your server and trying to craft new attacks that will bring that even more hardened server down.

Tasks 1, 2, 3 and 4 can be done in parallel and task 5 should be done in the end.

## Scoring

The Blue Team receives a point for each legitimate client's request that the server processes and responds to within 500 ms. Red Team gets the point otherwise.

## Exercise Dynamics

Teams will need to simultaneously act as Blue Team and Red Team throughout the 2h exercise. We will then have a 10 min break followed by a post-mortem discussion and selection of a winning team.

## Grading

Each team member will be graded based on their contribution to the team effort, not based on the team's performance. After the exercise each team member will submit a report containing the list of contributions they made to the team effort - e.g., modules that they coded, testing and setup they performed, etc. All team members must sign each report. Reports will be delivered to the instructor in class. The grades will be assigned based on the report.

## Useful Links

You can use any programming language you like for any part of your assignment.

1. You can use `netcat` to send packets in a DDoS attack. To install do apt-get install `netcat`. Also see netcat manual
2. You can use `tcpdump` to record network traffic. You can develop your own scripts to analyze it.
3. You can also look at Web server logs at `/var/log/apache2`