Setup Manual: Buffer Overflow CTF Competition

1 Installation and Setup

During the competition, instructors run a CTF web server, through which, they can start vulnerable servers for students to attack. Each team will be assigned a vulnerable server, and its goal is to launch buffer-overflow attacks on the server.

1.1 Step 1: Environment Setup

The CTF web server uses several software packages that have not been installed on our VM (they will be added in future VM releases). These packages can be easily installed using the following commands:

```
$ sudo pip3 install Flask==1.0.3 watchdog==0.9.0
$ sudo apt-get install python3-gevent
```

1.2 Step 2: Download the Zip File

Download the file `seedctf.zip` from the website, unzip it, and you will see a folder called `seedctf/`.

1.3 Step 3: Prepare Team flags

Once a team's attack is successful, its flag will be raised. To add more fun to it, we ask students to submit their own flags. Instructors need to manually copy those images files (`png` or `jpg` formats) to the `seedctf/assets/flags` folder. The flag name needs to have the following format: `teamN.png` or `teamN.jpg`, where `N` is the team number. For example, `team1.jpg` and `team17.png` are valid filenames for Team 1 and 17, respectively. We have already placed some generic flags in the folder.

1.4 Step 4 (Optional Step): Create Victory Sound Effect

A victory sound will be played if a team got the flag. If you would like to play a different sound/music, you can replace the sound file (`seedctf/assets/victory.wav`) with your own file.

By default, modern browsers block JavaScript from playing audio without human interaction (i.e., if the play event is not a direct effect of a user interaction, such as clicking buttons, the browser will not play the sound). Since the victory sound effect is not triggered by a user interaction, the sound effect will not play unless the browser explicitly allows it. Please refer to your browser's documentation to see how to enable webpage from playing sounds automatically. If you do not want the sound effect, you can always skip this step.

2 Manage the CTF Competition

The CTF competition management is conducted through a web server. We show how to start this server, and then how to use the web interfaces to manage the CTF system.

2.1 Step 1: Start the CTF Web Server

First, let us get the web server started. The following command line arguments are available to configure the CTF server:

Listing 1: Buffer-Overflow Server Usage

```
usage: buffer-overflow.py [-h] [--config CONFIG] [--host HOST] [--port PORT]
                          [--ssl] [--cert CERT] [--key KEY] [--disable-aslr]
                          --token TOKEN

Optional arguments:
  -h, --help         show this help message and exit
  --config CONFIG    path to CTF configration file
  --host HOST        the host for frontend to bind to (default: 0.0.0.0)
  --port PORT        the port for frontend to bind to (default: 80 for http,
                     443 for https)
  --ssl              enable https for frontend
  --cert CERT        path to ssl certificate file
  --key KEY          path to ssl key file
  --disable-aslr     disable ASLR if enabled. ASLR will be re-enabled when the
                     program quit if it was previously on
  --token TOKEN      the instructor token
```

The command line arguments should be quite self-explanatory. To set up a minimum buffer overflow CTF competition without any preconfigured sessions on port `8080`, use the following command:

```
$ sudo -u nobody ./buffer-overflow.py --token "secret" --port 8080
```

It should be noted that we run the server using `nobody` as the user ID, instead of using the `seed` user. Essentially, we are running the server with a reduced privilege.

Once the server started, you can navigate to the instructor interface and add CTF sessions there. If you are running a browser inside the VM, the instructor page should be available on `http://127.0.0.1:8080/instructor`. When the page prompts for token, use `"secret"` (without the quotation marks). We suggest that instructors change it to something else; otherwise, students can also access the server using this default token.

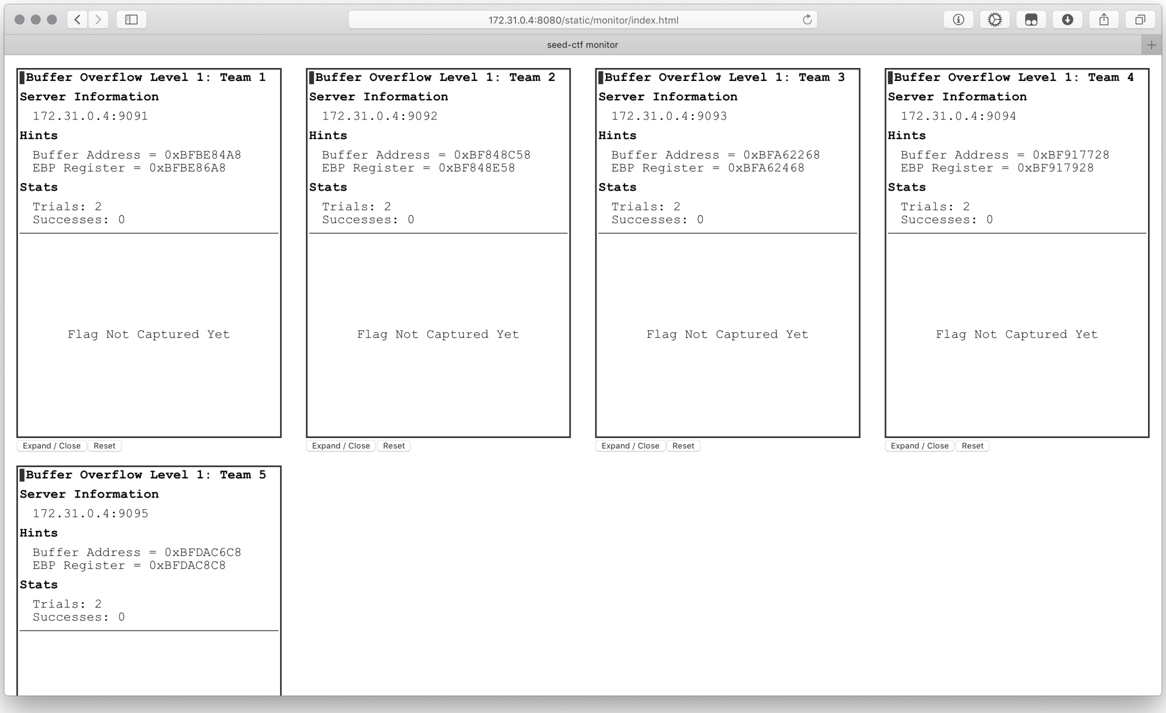
Notes: If the competition server failed to start, or you cannot add new sessions when running the server with dropped permission, please check your `/tmp` folder and remove any CTF related files.

2.2 Step 2: Project the Monitor Interface

Once the server is started, a web page will be available on `http://127.0.0.1:8080/`. The webpage is called “Monitor Interface.” The instructor will project the interface to the class. Figure 1 shows what the monitor interface looks like.

The monitor interface should be available for students to open on their own computers (they need to replace `127.0.0.1` with the server’s actual IP address). The monitor interface will have multiple server cards, showing information about each server (we call it session). Figure 2 shows an example of a server card.

The Server Card. There are five sections in a server card: title, server information, hints, stats, and flag. The title area shows the level and team that this session belongs to. The server information section



```
Buffer Overflow Level 1: Team 1
Server Information
  172.31.0.4:9091
Hints
  Buffer Address = 0xBFBE84A8
  EBP Register = 0xBFBE86A8
Stats
  Trials: 2
  Successes: 0

Flag Not Captured Yet
```

```
Buffer Overflow Level 1: Team 2
Server Information
  172.31.0.4:9092
Hints
  Buffer Address = 0xBF848C58
  EBP Register = 0xBF848E58
Stats
  Trials: 2
  Successes: 0

Flag Not Captured Yet
```

```
Buffer Overflow Level 1: Team 3
Server Information
  172.31.0.4:9093
Hints
  Buffer Address = 0xBFA62268
  EBP Register = 0xBFA62468
Stats
  Trials: 2
  Successes: 0

Flag Not Captured Yet
```

```
Buffer Overflow Level 1: Team 4
Server Information
  172.31.0.4:9094
Hints
  Buffer Address = 0xBF917728
  EBP Register = 0xBF917928
Stats
  Trials: 2
  Successes: 0

Flag Not Captured Yet
```

```
Buffer Overflow Level 1: Team 5
Server Information
  172.31.0.4:9095
Hints
  Buffer Address = 0xBFDAC6C8
  EBP Register = 0xBFDAC8C8
Stats
  Trials: 2
  Successes: 0
```

Figure 1: The monitor interface

displays the server details. The hint section provides hints; the stats section indicates the number of trials and successes, and the flag area displays flags once an attack has succeed.

Two controls are available on the bottom of the card. The `"Expand/Close"` button will expand or close a server card. Students might expand the card of their team. An expand card will fill the browser window and have an extra console section available. The console section, however, was not used in this particular CTF competition. The `"Reset"` button resets the flag (only on the local browser), allowing teams to try the attack again after a successful attack.

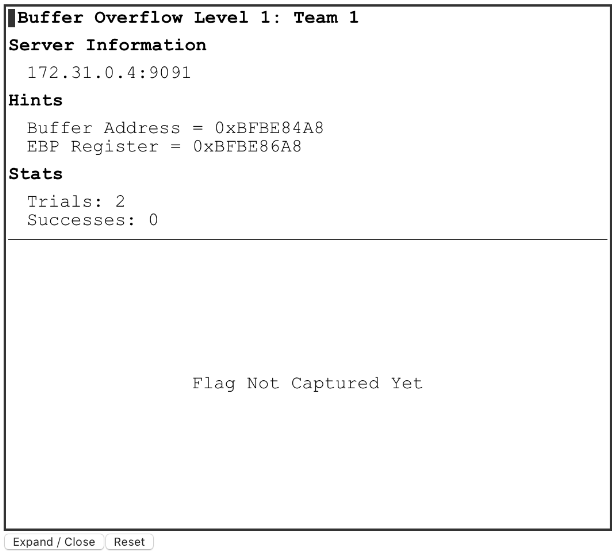
2.3 Step 3: Start Attack Sessions

Instructors can use `http://127.0.0.1:8080/instructor` to manage the CTF system. This is an instructor-only control interface that is protected by a secret token specified when the instructor starts the CTF server. Figure 3 shows what the control interface looks like.

Using this interface, instructors can create new sessions for teams or stop existing sessions. They can also click the `"Get Answer"` button to see the parameters that are hidden to students. The `"Add session"` side panel is used to create new sessions. The meaning of each field in this panel can be found from Table 1.

3 Set Up Port Forwarding

The instructor needs to let students know the IP addresses of the vulnerable servers. If these servers run on a machine with a public IP address, the instructor can skip this section. We do recommend that the instructor



```
Buffer Overflow Level 1: Team 1
Server Information
  172.31.0.4:9091
Hints
  Buffer Address = 0xBFBE84A8
  EBP Register = 0xBFBE86A8
Stats
  Trials: 2
  Successes: 0

Flag Not Captured Yet
```

Figure 2: The server card

runs these vulnerable servers inside a virtual machine. In this case, the IP address of the virtual machine will likely be a private address, and the VM is not accessible directly from the outside. To solve this problem, we need to ask students to send their packets to the host machine (which has a public IP address), and the host machine will forward packets to the VM running inside. This is called port forwarding, and we can set it up in VirtualBox.

For example, if a vulnerable server program needs to receive requests on port `9090` on the VM, we will map port `NNN` of the host OS to port `9090` on the VM, so when the host OS receives packets on its port `NNN`, it will forward the packets to port `9090` on the VM. Here, `NNN` can be any port number that is available on the host, but to avoid confusion, we always use the same number, i.e., `NNN = 9090`, if we want to forward to the `9090` port of the VM. Figure 4 illustrates how port forwarding allows outside to communicate with server programs in the VM. In the following, we will show how to set up port forwarding.

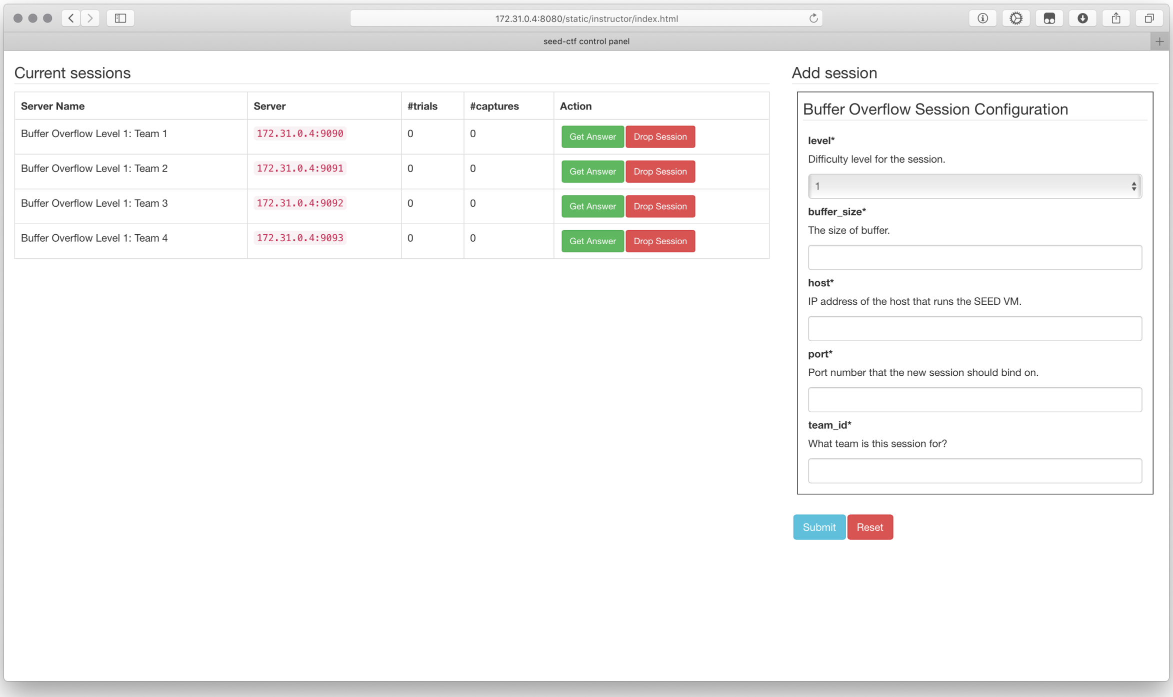
Step 1: Disable Firewall. Before the port forwarding is set up, we should make sure that the firewall on the host machine is turned off. Port forwarding might be affected by firewalls, such as Windows Defender Firewall, MCafee, etc.

Step 2a: Configure Using Command Line. We use an example to show how to add a port forwarding rule.

```
  Host IP:   10.1.162.10
  Host Port: 9090
  VM IP:     10.0.2.70
  VM Port:   9090
```

We can use the `VBoxManage` tool (installed with VirtualBox) and use its command line options as shown below to add a port forwarding rule (we name the rule "`ruleA`"). This step can be repeated to set up port forwarding for other port numbers as well.

```
$ VBoxManage natnetwork modify --netname NatNetwork --port-forward-4
             "ruleA:tcp:[10.1.162.10]:9090:[10.0.2.70]:9090"
```



Current sessions

Server Name	Server	#trials	#captures	Action
Buffer Overflow Level 1: Team 1	172.31.0.4:9090	0	0	Get Answer Drop Session
Buffer Overflow Level 1: Team 2	172.31.0.4:9091	0	0	Get Answer Drop Session
Buffer Overflow Level 1: Team 3	172.31.0.4:9092	0	0	Get Answer Drop Session
Buffer Overflow Level 1: Team 4	172.31.0.4:9093	0	0	Get Answer Drop Session

Add session

Buffer Overflow Session Configuration

level*
Difficulty level for the session.
1

buffer_size*
The size of buffer.

host*
IP address of the host that runs the SEED VM.

port*
Port number that the new session should bind on.

team_id*
What team is this session for?

Figure 3: The instructor interface

Step 2b: Configure Using User Interface. Port forwarding can also be enabled using VirtualBox's user interface. To do this, we need to navigate to File ->Preferences ->Network. Click one of the networks for which port forwarding rules need to be added, and then click the Port Forwarding button (shown in Figure 5(a)). To add a rule, click the plus button as shown in Figure 5(b), and then fill in the forwarding rule. Click Ok to save the rule. In Figure 5(c), we port forward from the host OS's (10.1.162.10) port 9090 to a VM's (10.0.2.70) port 9090.

Appendix

A Start the competition server with Pre-configured Sessions

Other than using the web interface to start competition sessions manually, instructors can also use a file to pre-configure multiple sessions, and start them from the command line. The session configuration file uses the JSON (JavaScript Object Notation) format. It contains an array of session configuration objects. The configuration object is defined in Table 1.

Note: For Level 3, the complete address of the buffer is not given. We use address_mask to specify how many bits of the buffer address should be hidden. For example, if the return address is 0xBFFFC7EC and the address mask is 0xFFFFF000, the range of the buffer address provided to students will be from 0xBFFFC000 to 0xBFFFCFFF.

An Example. We give an example of the configuration file in the following:

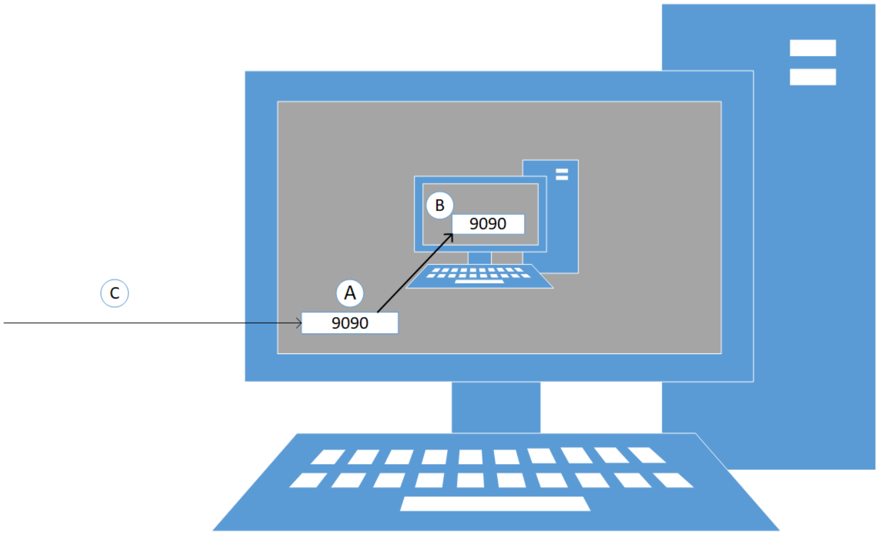


Figure 4: Port-Forwarding Setup

Table 1: Fields in session configuration

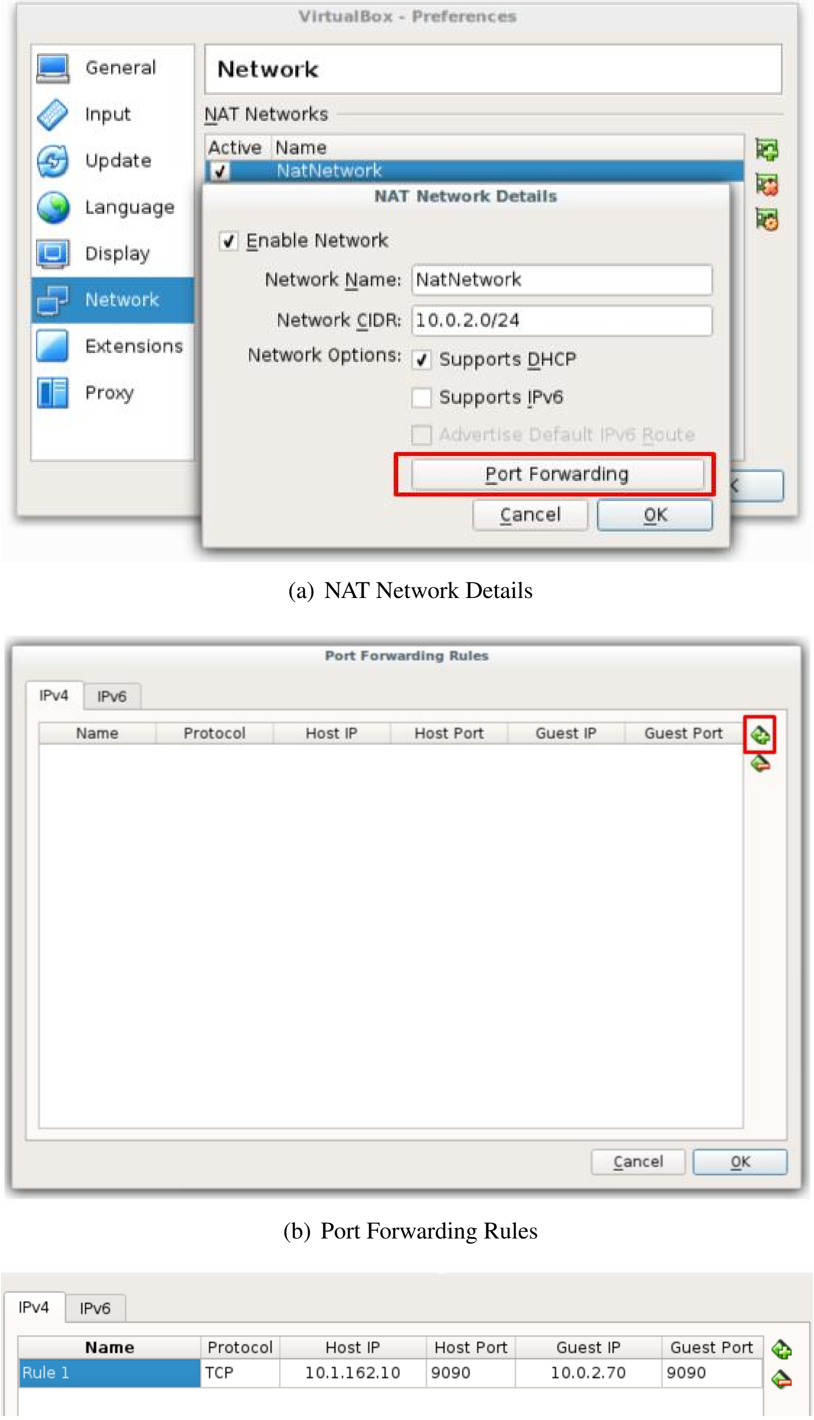
Field	Type	Required?	Description
level	integer	yes	1 to 4. The difficulty level for this session.
buffer_size	integer	yes	The size of buffer for this session.
host	string	yes	IP address of the host that runs the SEED VM.
port	integer	yes	The port number for this session to bind.
team_id	integer	yes	The ID of team for this session.
buffer_high	integer	for levels 2 and 3	The value to be added to the buffer size.
buffer_low	integer	for levels 2 and 3	The value to be deducted from the buffer size.
address_mask	string	for level 3	The address mask.

Listing 2: Sample configuration

```
[
   {
      "level": 1,
      "team_id": 1,
      "buffer_size": 500,
      "port": 9091,
      "host": "0.0.0.0",
      "host_friendly": "172.31.0.4"
   },
   {
      "level": 2,
      "team_id": 2,
      "buffer_size": 500,
      "port": 9092,
      "host": "0.0.0.0",
      "host_friendly": "172.31.0.4",
      "buffer_high": 200,
      "buffer_low": 100
   },
   {
      "level": 3,
      "team_id": 3,
      "buffer_size": 500,
      "port": 9093,
      "host": "0.0.0.0",
      "host_friendly": "172.31.0.4",
      "buffer_high": 200,
      "buffer_low": 100,
      "address_mask": "0xfffffc00"
   },
   {
      "level": 4,
      "team_id": 4,
      "buffer_size": 500,
      "port": 9094,
      "host": "0.0.0.0",
      "host_friendly": "172.31.0.4"
   }
]
```

The example above shows a sample configuration file with four sessions configured, each for a different team and for a different difficulty level. To start the CTF server using this configuration, use the following command (`config.json` is the configuration file):

```
$ ./buffer-overflow.py --token secret --config config.json
```



☑ Enable Network

Network Name: NatNetwork

Network CIDR: 10.0.2.0/24

Network Options: ☑ Supports DHCP

□ Supports IPv6

□ Advertise Default IPv6 Route

(a) NAT Network Details

(b) Port Forwarding Rules

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
Rule 1	TCP	10.1.162.10	9090	10.0.2.70	9090

(c) Add New Rule

Figure 5: Port Forwarding