

CAPTURE THE PI HAT - NETWORKING EXERCISES

2019 GenCyber Camps at Texas A&M University, College Station, TX 77843

OVERVIEW - Students log into remote devices and transfer files securely using technologies common to large scale computing. By gaining access and executing “malicious” python code a partner’s RPI students learn that their machine are susceptible to attacks by the adversary. A LED equipped sense-hat adds a visual component to this hands-on activity.

Student Learning Objectives -

- Demonstrate basic networking skills related to file transfer and remote access
- Ensure (or exploit) weak user accounts and password management systems
- Leverage python scripting skills to identify and exploit vulnerabilities in open-access devices on public networks

Expected Student Knowledge -

- Familiarity with basic linux commands such cp, mv, rm, mkdir, clear, cat
- basic python scripting on the RPI sense hat
- Familiarity with installing and updating packages using “apt-get”
- Familiarity with a text editor such as “gedit” or a python emulator

Hardware and Software Prerequisites -

- RPI and RPI sense hat.
- Linux OS and Python
- Network (wireless or connected) that allows connections between RPIs.

Linux Commands Introduced -

- top, ifconfig, ping, passwd, ssh, scp, sudo, mv, shutdown, chmod, nmap, apt-get, cat, more, tcp dump

Ethical issues to be Addressed -

- As part of this exercise, students share passwords and allow others access to their computers. Students should be strongly cautioned against doing so in real life.
- This activity introduces students to transfer and communications methodologies that may be mistaken to be related to hacking. This should be clarified

- Students learn about vulnerabilities in systems during this exercise. It is critical that students understand the need for these technologies and be aware of the repercussions of using them in unethical and/or illegal practices.

GenCyber Cybersecurity Concepts Covered -

- Confidentiality
- Defense in depth
- Think like an adversary

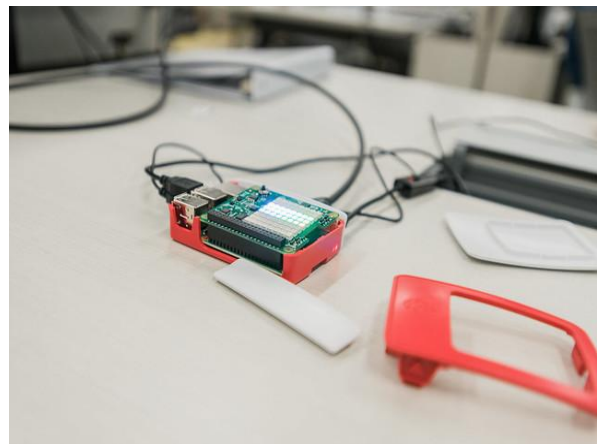


Figure 1. A sense hat on a raspberry pi (RPI) microcontroller. The sense-hat includes a variety of sensors and a programmable LED display.

SEQUENCE OF ACTIVITIES -

1. Setting up the RPI

Difficulty level: Introductory

At the start of the exercise, students reset their RPI user password to “raspberrypi” using the “passwd” command on the command line, or by using the systems’ settings. Using the user-interface, students enable incoming connections on the machine and connect to a common wireless network. At the start of the exercise the students log in as the default user (pi) on their machines. While the common password is known to all students, the I.P. address space is not known.

Once the RPIs have been set up, each student partners with another student. These

students have an agreement that (a) they will allow each other to remotely access their machines, and (b) not do any harm to each other's machines.

In the following steps, we will outline how students will identify their IP addresses, access their partner's machines remotely and transfer code that activates the sense hat, and shows that they have "captured" their partner's RPI hat.

2. Ping the remote host using ping

Difficulty Level: Introductory

Students are next asked to ping a remote host, preferably a website such as google.com. This allows students to verify that there is a communication channel between their machine and the target host. The instructor can next ask the students find the slowest website (host server), i.e. one that has the longest packet round-trip time. It is suggested that students only use main-stream websites during this competition. Once the rules have been established, we suggest that the competition be run for 5 minutes.

```

tripham@main-10-230-188-175: ~ (zsh)
Last login: Tue Sep 24 06:31:08 on console
➔ ~ ping google.com
PING google.com (172.217.9.14): 56 data bytes
64 bytes from 172.217.9.14: icmp_seq=0 ttl=55 time=10.138 ms
64 bytes from 172.217.9.14: icmp_seq=1 ttl=55 time=16.077 ms
64 bytes from 172.217.9.14: icmp_seq=2 ttl=55 time=15.380 ms
64 bytes from 172.217.9.14: icmp_seq=3 ttl=55 time=16.092 ms
64 bytes from 172.217.9.14: icmp_seq=4 ttl=55 time=16.800 ms
64 bytes from 172.217.9.14: icmp_seq=5 ttl=55 time=12.633 ms
64 bytes from 172.217.9.14: icmp_seq=6 ttl=55 time=16.717 ms
64 bytes from 172.217.9.14: icmp_seq=7 ttl=55 time=16.990 ms
64 bytes from 172.217.9.14: icmp_seq=8 ttl=55 time=15.875 ms
64 bytes from 172.217.9.14: icmp_seq=9 ttl=55 time=16.622 ms
64 bytes from 172.217.9.14: icmp_seq=10 ttl=55 time=14.051 ms
64 bytes from 172.217.9.14: icmp_seq=11 ttl=55 time=16.827 ms
^C
--- google.com ping statistics ---
12 packets transmitted, 12 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 10.138/15.350/16.990/2.001 ms
➔ ~ |

```

Figure 2. Sample output generated by the ping command while attempting to reach google.com

3. Run ifconfig

Difficulty Level: Introductory

```

threeine.co.uk
pi@raspberrypi ~ $ ifconfig
eth0:
Link encap:Ethernet HWaddr b8:27:eb:7a:be:1d
inet addr:192.168.0.31 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:83747 errors:0 dropped:0 overruns:0 frame:0
TX packets:1722 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:41095463 (39.1 MiB) TX bytes:5995932 (5.7 MiB)

lo:
Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:10 errors:0 dropped:0 overruns:0 frame:0
TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:752 (752.0 B) TX bytes:752 (752.0 B)

pi@raspberrypi ~ $

```

Figure 3. Output produced by running the ifconfig command

Students find their RPI's IP address by running the "ifconfig" command. Participants will exchange their RPI's IP address with their designated partner. The paired students are next asked to ping their own machine, followed by their partner's machine.

4. Connect to a remote machine via SSH

Difficulty Level: Intermediate

```

Utilizadors-MacBook-Pro: ~ Rui$ ssh pi@192.168.1.98
The authenticity of host '192.168.1.98 (192.168.1.98)' can't be established.
ECDSA key fingerprint is SHA256:xaT0ApFlZ2In-raSkJzaymZ8Jg-Qny91fy3J/hlvjI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.98' (ECDSA) to the list of known hosts.
pi@192.168.1.98's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Dec 19 16:44:48 2015 from desktop-c5bdf0f.lan
pi@raspberrypi:~$

```

Figure 4. Connecting to a remote RPi using ssh.

Students should use "ssh" to first log into their RPi followed by their partners' (remote) RPi using their respective IP addresses and default passwords. This activity illustrates the importance of having secure passwords. Having an insecure passwords will allow adversaries to compromise computing systems. This is best demonstrated by a follow-up activity in which students can use this information to shut down (using sudo shutdown command) their remote machine without the owner's knowledge.

```

top (top)
Processes: 338 total, 2 running, 336 sleeping, 1413 threads
Load Avg: 1.26, 1.18, 1.22 CPU usage: 2.3% user, 1.46% sys, 96.50% idle
SharedLibs: 381M resident, 71M data, 50M linkedit.
MemRegions: 69619 total, 4167M resident, 205M private, 1896M shared.
PhysMem: 18G used (2333M wired), 5946M unused.
VM: 1548K vszize, 1373M framework vszize, 0(0) swaptins, 0(0) swaptouts.
Networks: packets: 182331/136M in, 262362/50M out. Disks: 285654/3267M read, 128538/1800M written.

PID COMMAND %CPU TIME #TH #WQ REPORT MEM PURG CMPR DGRP PPID STATE
3338 screencaptur 0.8 00:00.18 5 3 196- 13M- 84K 0B 3338 1 sleeping
3337 screencaptur 7.1 00:00.44 2 1 54- 3300K- 620K 0B 290 290 sleeping
3336 backupd 0.0 00:00.03 2 1 44 1812K 0B 0B 3336 1 sleeping
3325 top 2.6 00:02.66 1/1 0 28 3936K 0B 0B 3325 3283 running
3328 Google Chrom 0.0 00:00.08 13 1 111 14M 4096B 0B 579 579 sleeping
3325 com.apple.iC 0.0 00:00.09 2 1 55 3700K 0B 0B 3325 1 sleeping
3283 zsh 0.0 00:00.29 1 0 21 2740K 0B 0B 3283 3282 sleeping
3282 login 0.0 00:00.01 2 1 31 1252K 0B 0B 3282 3281 sleeping
3281 ifern2 0.0 00:00.02 2 1 33 3728K 0B 0B 3281 634 sleeping
3280 CoreServices 0.0 00:00.10 3 1 161 4620K 0B 0B 3280 1 sleeping
3272 Google Chrom 0.0 00:00.22 14 1 165 127M 4096B 0B 579 579 sleeping
3269 distroconf 0.0 00:00.01 2 0 33 1956K 0B 0B 3269 1 sleeping

```

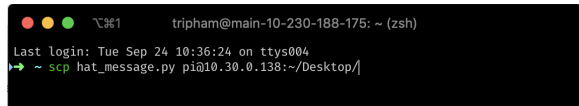
Figure 5. Sample output produced by running top

Here, we suggest introducing the "top" program as a means to monitor resources on unix system and how it can be used to identify running SSH processes. Since every student in the camp login to their peer's Raspberry Pi using the root password, it is not possible to identify who is using SSH to login to machine. This identifies a blind spot for when adversaries get access to machine root account and use it to access computing

resources without the knowledge of legitimate users.

5. Securely transferring files using scp

Difficulty Level: Intermediate



```
triphm@main-10-230-188-175: ~ (zsh)
Last login: Tue Sep 24 10:36:24 on ttys004
➔ ~ scp hat_message.py pi@10.30.0.138:~/Desktop/
```

Figure 6. Using scp to send file from a local machine to a remote host

In this activity, students use the “scp” command to securely transfer files between two machines (RPIs in this case). Using the scp command, it is also possible to transfer the data from the remote machine to the student’s local machine. Once students have learned the syntax of the command, they practice by exchanging arbitrary files between their RPIs. Students next ssh into their partner’s machines to ensure that the files have indeed transferred.

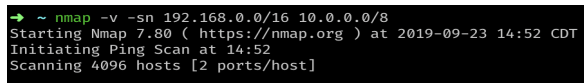
Prior to starting the advanced section of this activity, students should become comfortable with using scp to transfer files, and remotely logging into their partner’s machine’s remotely. In this activity, students transfer a previously created “malicious file” that flashes a message on a RPI sense hat. A simple sensor hat code is seen in Figure 7. Students are asked to transfer their malicious code using to their partner’s RPI and execute it after logging into their machine. On executing the program a custom message flashes on the partner’s RPI sense hat, informing the partner that her/his machine has been compromised. ***The pi hat has is captured!*** This is a race and every team will have an instant winner. As a fun activity, students can change the code to flash different messages on their partner’s Sense Hat .

```
1 #!/usr/bin/env python
2 # this script will show a scrolling message on the Pi HAT
3 from sense_hat import SenseHat
4 sense = SenseHat()
5
6 # send the text Hello, <your name>! to be displayed on the
7 Sense Hat.
# You should replace <your name> with the actual name of the
other student.
sense.show_message("Hello, <your name>!")
```

Figure 7. A sample python script that can be used during the file transfer exercise. This file contains instructions to display a specific message on the partner’s RPI sense hat.

6. Perform a network scan using nmap

Difficulty Level: Advanced



```
➔ ~ nmap -v -sn 192.168.0.0/16 10.0.0.0/8
Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-23 14:52 CDT
Initiating Ping Scan at 14:52
Scanning 4096 hosts [2 ports/host]
```

Figure 8. An example of host scanning using nmap

On completion of exercise 5, students should realize that knowing the I.P. address of a machine is a critical aspect to accessing a remote machine. Building on this knowledge, in this activity students next learn how to discover existing machines connected to the same network (subnet) using network mapper (nmap). The intention is to make students aware of one of many possible methods that other people, especially adversaries, can use to discover the existence of connected devices to a local network. Nmap also provides the ability to fingerprint devices which can potentially expose flaws of vulnerable machines.

As an advanced exercise, students can be asked to write a python script to ping and ssh into identified vulnerable machines. This should only be done if the students are working on a closed or isolated network. Students in the classroom can be advised to practice defensive maneuvers such as changing their password or taking their RPI off the network. In contrast, an enterprising student can take over all the RPIs in a room before others

7. Closing out the exercise

Difficulty Level: Introductory

To close out the exercise the students are asked to change the passwords on their RPIs via the GUI or command line using the “passwd” command.

CAMP CONTACT INFORMATION

Support from NSA GenCyber is gratefully acknowledged. Instructional materials are on request. Please send us your questions, comments and suggestions to the Texas A&M GenCyber camp team at:

- Gencyber camp email address
help-gencyber@hprc.tamu.edu
- Director, Prof. Dilma Da Silva
dilma@cse.tamu.edu
- Lead Instructor, Dr. Dhruva Chakravorty
chakravorty@tamu.edu