Weapon Systems Cyber Protection

Manav Shah, Christopher LaManna, Ratan Nambiar, Sanila Tabassum
Customer : Northrop Grumman
SME: Chris Hummel

Problem Statement

Weapons systems not only depend on hardware, but also on software to function effectively and properly. The software capabilities being used in various weapons systems are often exposed and vulnerable to different types of attacks. Successfully hacking into the weapon system can lead to catastrophic consequences. Millions of dollars can be lost, and people could potentially die.

Background

As mentioned the problem statement, weapon systems serve as a critical part of national security and a compromise to weapon system can lead to drastic losses. Our solution for this problem is to use a single board computer to properly and effectively secure a weapon system. In designing the device, we chose to use the Raspberry Pi as the single board computer. We are configuring it with a virtual private network, firewall, intrusion detection system, and a Linux operating system.

The stakeholder for this project is Northrop Grumman, one of the largest defense contractors in the nation. The point of contact from Northrop Grumman that we regularly interacted with was Chris Hummel.

Business Plan

Our business plan for this project would be for Northrop Grumman to acquire our product and implement it in their existing solutions. The cost of this device is less than $100. This is very cheap and if it saves one weapons system it will break-even and have a large return on the investment made. We intentend to give this to Northrop Grumman and have them implement it on the needed weapons systems.

Concept of Operations

The user of the single board computer will connect it directly to a system and be able to use the resources provided by the single-board computer to secure the network. Figure 1 displays the CONOPS for the device.

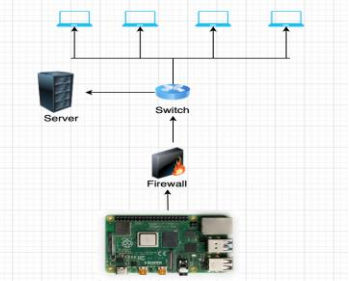


Figure 1: CONOPS Diagram

Architecture

The system architecture, as shown in Figure 2, is the fundamental underlying design of the device. The single-board computer is the hardware component of the device and will contain open-source software such as a firewall, an intrusion detection and prevention system, and a network monitoring application.

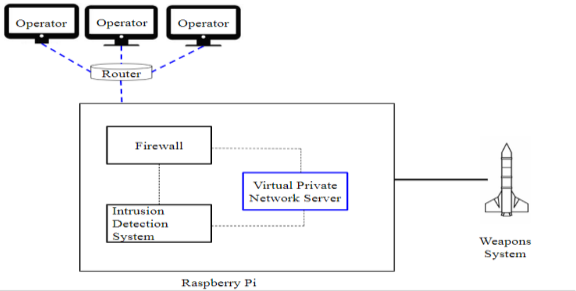


Figure 2: System Architecture

Data Model

The data model, Figure 3, illustrates how elements of the system are structured and defines the relationships between those elements as well as the flow of information in the system.

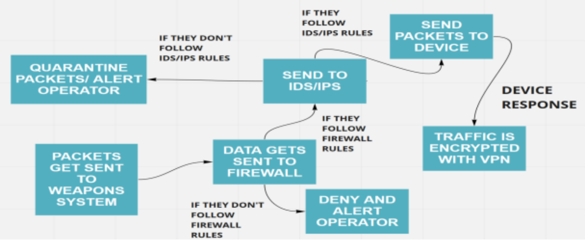

PACKETS GET SENT TO WEAPONS SYSTEM
DATA GETS SENT TO FIREWALL
IF THEY FOLLOW FIREWALL RULES
IF THEY DON'T FOLLOW FIREWALL RULES
DENY AND ALERT OPERATOR
SEND TO IDS/IPS
IF THEY DON'T FOLLOW IDS/IPS RULES
QUARANTINE PACKETS/ ALERT OPERATOR
IF THEY FOLLOW IDS/IPS RULES
SEND PACKETS TO DEVICE
DEVICE RESPONSE
TRAFFIC IS ENCRYPTED WITH VPN

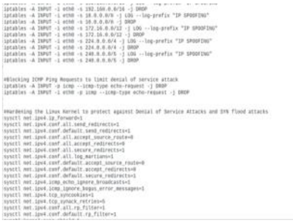
Figure 3: Data Model

Design Tradeoffs

- Single-Board Computer
  - UDOO X86 II
    - Good memory capacity and performance capabilities
    - Sold out, so we opted for the Raspberry Pi 4
  - Raspberry Pi 4
    - Comparable performance to UDOO X86 II.
    - Hardware was ver new, incompatible with a lot of the open source software
    - We opted for an older model, the Raspberry Pi 3b+
- Firewall
  - pfSense
    - great open source firewall
    - pfSense has to be loaded as the operating system on the pi
    - wanted more flexibility with the operating system to download additional security modules
    - we opted for iptables as the firewall.

Implementation

In order to configure the iptables (firewall), we wrote and executed a bash script that would protect against denial of service attacks, SYN flood attacks, IP spoofing, and other configurations to protect the network. Figure 4 shows a snippet of the firewall script and figure 5 displays the firewall rules.

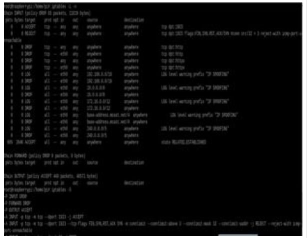


```
iptables -A INPUT -i eth0 -s 192.168.0.0/16 -j DROP
iptables -A INPUT -i eth0 -s 10.0.0.0/8 -j LOG --log-prefix "IP SPOOFING"
iptables -A INPUT -i eth0 -s 10.0.0.0/8 -j DROP
iptables -A INPUT -i eth0 -s 172.16.0.0/12 -j LOG --log-prefix "IP SPOOFING"
iptables -A INPUT -i eth0 -s 172.16.0.0/12 -j DROP
iptables -A INPUT -i eth0 -s 224.0.0.0/4 -j LOG --log-prefix "IP SPOOFING"
iptables -A INPUT -i eth0 -s 224.0.0.0/4 -j DROP
iptables -A INPUT -i eth0 -s 240.0.0.0/5 -j LOG --log-prefix "IP SPOOFING"
iptables -A INPUT -i eth0 -s 240.0.0.0/5 -j DROP

#Blocking ICMP Ping Requests to limit denial of service attack
iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
iptables -A INPUT -i eth0 -p icmp --icmp-type echo-request -j DROP

#Hardening the Linux Kernel to protect against Denial of Service Attacks and SYN flood attacks
sysctl net.ipv4.ip_forward=1
sysctl net.ipv4.conf.all.send_redirects=1
sysctl net.ipv4.conf.default.send_redirects=1
sysctl net.ipv4.conf.all.accept_source_route=0
sysctl net.ipv4.conf.all.accept_redirects=0
sysctl net.ipv4.conf.all.secure_redirects=1
sysctl net.ipv4.conf.all.log_martians=1
sysctl net.ipv4.conf.default.accept_source_route=0
sysctl net.ipv4.conf.default.accept_redirects=0
sysctl net.ipv4.conf.default.secure_redirects=1
sysctl net.ipv4.icmp_echo_ignore_broadcasts=1
sysctl net.ipv4.icmp_ignore_bogus_error_messages=1
sysctl net.ipv4.tcp_syncookies=1
sysctl net.ipv4.tcp_synack_retries=5
sysctl net.ipv4.conf.all.rp_filter=1
sysctl net.ipv4.conf.default.rp_filter=1
```

Figure 4: Firewall Script



```
root@raspberrypi:/home/pi# iptables -L -v
Chain INPUT (policy DROP [illegible] packets, [illegible] bytes)
 pkts bytes target     prot opt in     out     source               destination
[illegible]
Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT [illegible] packets, [illegible] bytes)
 pkts bytes target     prot opt in     out     source               destination
root@raspberrypi:/home/pi# iptables -S
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT ACCEPT
[illegible]
```

Figure 5: Firewall Rules

Summary

The objective of this project was for us to design and build a board-level communications and network protection capability for a deployed weapon system. In order to accomplish this, we configured a Raspberry Pi with a firewall, IDS, and VPN in order to detect threats. The next part of the project would have been to test the configurations, however, due to COVID-19, we were unable to do so. In order to test it, we would have generated malicious network traffic and analyzed the number of false positives and negatives in order to further improve the configurations for the device.

Acknowledgements: We would like to thank Northrop Grumman, our customer point of contact and subject matter expert, Chris Hummel, Professor Gino Manzo and the GMU CYSE Department for their continuous feedback and mentorship throughout the project.

April 27, 2020