UNR RET Research Experience in Cybersecurity for Nevada Teachers

Frankie Clark | Lab Assistance Mackenzie Zappe and Ignacio Astaburuaga

Program Leads Shamik Sengupta, Jay Thom and Dave

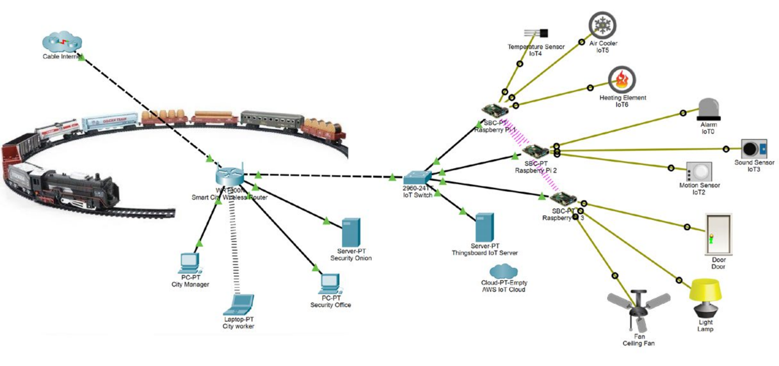
Introduction

This project is meant to layout a blueprint for building out UNR's Smart City to enterprise level proportions for the purpose of in-depth Cybersecurity research and testing.

Currently, UNR's Smart City is composed of a head unit Asus PC, a Direct Logic 205 PLC, and a few sensors and actuators connected to a N scale train set.

Our goal is to build the Smart City out to mimic a real-life Smart City network for Cybersecurity study.

Let's Get Down and City



The model incorporates real-life elements, such as AD, DNS, employee clients, Security, and IoT Management.

Major Networking Elements

- Windows AD, DNS, DC placed in UNR DMZ
- Wireless router acts as DMZ internal firewall
- Security Onion taps are placed in line to monitor network traffic (also may SPAN router interfaces)
- Thingsboard IoT server is setup and programmed in Python to connect/manage the RPi's GPIO pins connected to the IoT sensors, actuators.
- City employee clients are connected to AD.
- Cisco Firewall and VPN may be added later.

Security Onion

- Security Onion was overhauled in the past 18 months from its Ubuntu platform to CentOS, docker and containers.
- Tools like Snort and Squert/Sguil have been replaced by Suricata, Kibana, Zeek, and Wazuh.
- The core search tool in Security Onion is Elasticsearch, a RESTful search and analytics engine.
- Real-time IDS duties are performed by Wazuh and Suricata, enterprise Cybersecurity toolset.
- Kibana is an open source tool to help you visualize Elasticsearch data.
- Taps are preferrable to capture interface traffic but SPAN (mirroring) works fine.

Thingsboard IoT Server

- Thingsboard server is setup either on the ESXi stack or standalone.
- It enables device connectivity via industry standard IoT protocols (MQTT, CoAP, HTTP, ModBus) and supports both cloud and on-site deployments.
- It is owned and maintained in the Ukraine while at war with Russia.
- The programming language of choice to manage and control the single board computers (Raspberry Pi, Arduino) will be Python.
- Allows for UI direct integration and control of the Pi's GPIO pins connected to various sensors and actuators.
- Replicates a good attack surface for Cybersecurity research.

ESXi Hosts and Guests

- VMware's ESXi bare metal hypervisor will be installed on two enhanced Dell Precision workstations with three SSDs and 82GB of RAM.
- Each Dell Precision has a 6 core i7 enabling up to five VMs .
- Additional PCI NIC cards will be installed for the VMs.
- Security Onion and Thingsboard may be installed as guest VMs or on dedicated computers.
- Bare metal hypervisors separate the Guest OS from the underlying hardware to eliminate dependence on hardware and drivers.
- Snapshots will be taken to backup guests.

UNR RET Research Experience in Cybersecurity for Nevada Teachers

Frankie Clark | Lab Assistance Mackenzie Zappe and Ignacio Astaburuaga

Program Leads Shamik Sengupta, Jay Thom and Dave

Introduction

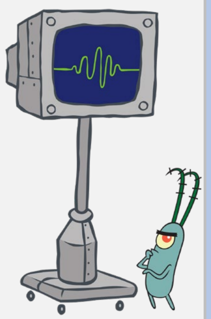
In this lesson (multi-class), students will already have a grasp of the OSI model and basic networking concepts. The lesson will occur at the end of the Cisco Packet Tracer mini-certification course taught to incoming Cybersecurity I students (Freshmen) at the beginning of the school year.

Students will utilize Packet Tracer to reverse engineer their Smart Home networks. If none exists, an example will be provided.

Lesson Essential Questions

The essential questions students will be able to answer:

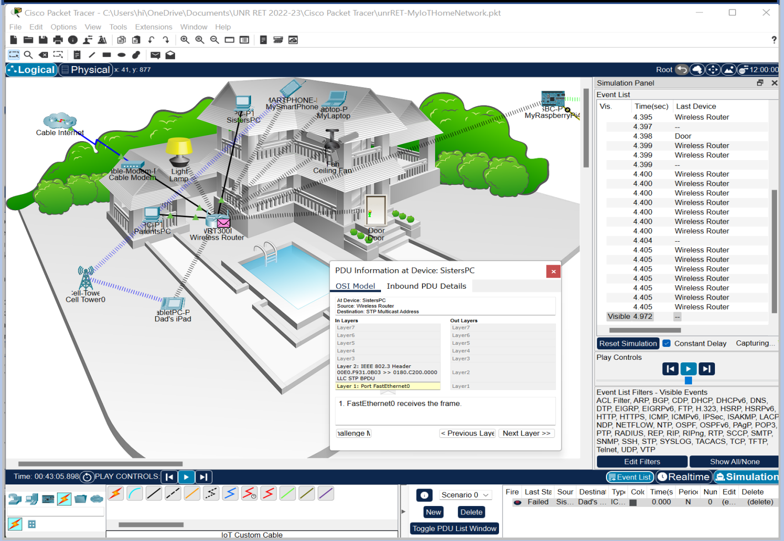
- What are the main components of my Smart Home network?
- What role does each component perform in the network and how are they networked?
- How can I use Simulation Mode in PT to follow packets and understand headers?
- How can I use PT to program my room trip wire and alarm?
- How can I use this model to improve my home Cybersecurity?



Reverse Engineer Your Home Network

Step 1: Inventory and map out all networked devices in your Smart Home network.
Step 2: Retrieve all network information related to each device (IP, Gateway, DNS, MAC address, SSID, Bluetooth)
Step 3: Use what you have learned in PT to reverse engineer an exact likeness of your Smart Home network.
Step 4: Test all settings, connections
Step 5: Use PT's Simulation mode to take a deep dive into networking protocols and concepts
Step 6: Program in JS/Python your room trip wire alarm

Simulation Mode – Trace Steps



PDU Information at Device: SistersPC

OSI Model | Inbound PDU Details

At Device: SistersPC
Source: Wireless Router
Destination: STP Multicast Address

In Layers	Out Layers
Layer7	Layer7
Layer6	Layer6
Layer5	Layer5
Layer4	Layer4
Layer3	Layer3
Layer 2: IEEE 802.3 Header 00E0.F931.0B03 >> 0180.C200.0000 LLC STP BPDU	Layer2
Layer 1: Port FastEthernet0	Layer1

1. FastEthernet0 receives the frame.

Challenge M | < Previous Layer | Next Layer >>

Simulation Panel

Event List

Vis.	Time(sec)	Last Device
	4.395	Wireless Router
	4.397	--
	4.398	Door
	4.399	Wireless Router
	4.399	Wireless Router
	4.399	--
	4.400	Wireless Router
	4.400	Wireless Router
	4.400	Wireless Router
	4.400	Wireless Router
	4.400	Wireless Router
	4.400	Wireless Router
	4.400	Wireless Router
	4.404	--
	4.405	Wireless Router
	4.405	Wireless Router
	4.405	Wireless Router
	4.405	Wireless Router
	4.405	Wireless Router
	4.405	Wireless Router
	4.405	Wireless Router
Visible	4.972	--

Reset Simulation ☑ Constant Delay Capturing...

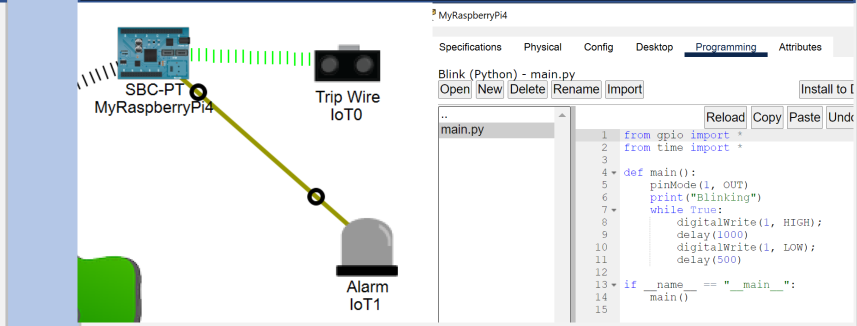
Play Controls

Event List Filters - Visible Events
ACL Filter, ARP, BGP, CDP, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Edit Filters | Show All/None

Fire	Last Sta	Sour	Destinat	Typ	Colo	Time(s	Perio	Num	Edit	Delete
●	Failed	Sis...	Dad's ...	IC...	■	0.000	N	0	(e...	(delete)

Program My Trip Wire Alarm System



Blink (Python) - main.py

main.py

```
from gpio import *
from time import *

def main():
    pinMode(1, OUT)
    print("Blinking")
    while True:
        digitalWrite(1, HIGH);
        delay(1000)
        digitalWrite(1, LOW);
        delay(500)

if __name__ == "__main__":
    main()
```

Students will create a Raspberry PI in PT and then utilize it's GPIO pins to connect to the Alarm triggered by the Trip Wire when someone enters the student's bedroom. The Trip Wire is connected to the single board computer via Bluetooth. The PI is programmed in Python to trigger the Alarm "1" upon the Trip Wire's light beam being interrupted.

Assessment

Pre-Assessment:
Students will participate in a whole class review based on all of the major PT concepts learned in the unit and how to apply them.

Post- Assessment:
Students will reverse engineer and build their Smart Home networks in PT and demonstrate full functionality via Simulation Mode, such as ping, tracert, etc. Students will add a single board computer and program it in Python to actuate their bedroom trip wire and alarm system.