

IoT Security Instructor Training Academy-Day 2019

it bildungsnetz



BERUFLICHE SCHULE ITECH
Elbinsel Wilhelmsburg



abacus
welten verbinden



IoT Security Instructor Training Academy-Day 2019

Agenda:

Part 1: Final Exam preparation and Final Exam

- Chapter keys content and
- fokus statements for the Final
- Examinees change rooms to for a quiet working environment

Part 2: Mini Lab's inside the course

- short lab overview
- small IoT example and security scan



Chapter 0
Course Introduction

Chapter 1
The IoT Under Attack

Chapter 2
IoT Systems and Architectures

Chapter 3
The IoT Device Layer Attack Surface

Chapter 4
IoT Communication Layer Attack Surface

Chapter 5
IoT Application Layer Attack Surface

Chapter 6
Vulnerability and Risk Assessment in an IoT System

Chapter 3 The IoT Device Layer Attack Surface

3.2 Vulnerabilities and Attacks at the Hardware Layer

3.2.1 Hardware Security

3.2.1.4 Lab - Investigate the FC

3.2.1.5 Lab - Compromise IoT I

3.2.2 Firmware Vulnerabilities

3.2.2.7 Lab - Compromise IoT I

Chapter 4 IoT Communication Layer Attack Surface

4.1 The IoT Communication Layer

4.1.2 Wireless Protocols

4.1.2.3 Lab - Sniffing Bluetooth with the Raspberry Pi

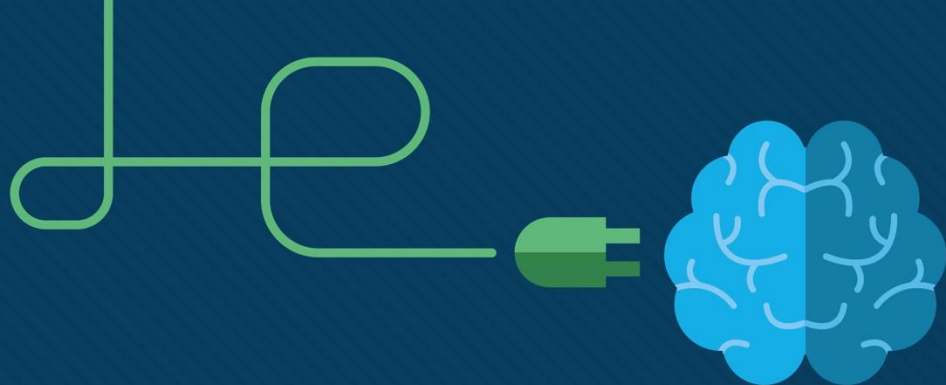
4.2 TCP/IP Vulnerabilities in IoT Networks

4.2.2 TCP and UDP Vulnerabilities

4.2.2.5 Lab - Port Scanning an IoT Device

4.2.2.6 Lab - Packet Crafting to Exploit Unsecured Ports

Chapter 5 IoT Application Layer Attack Surface



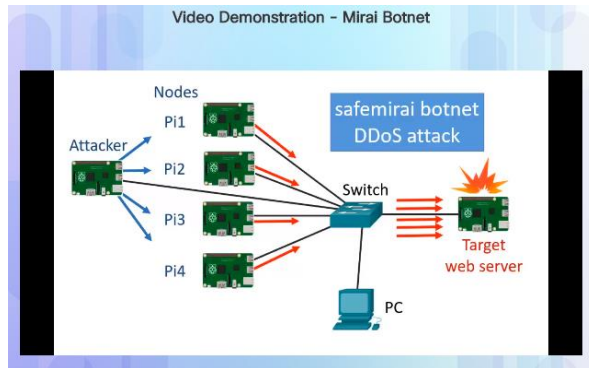
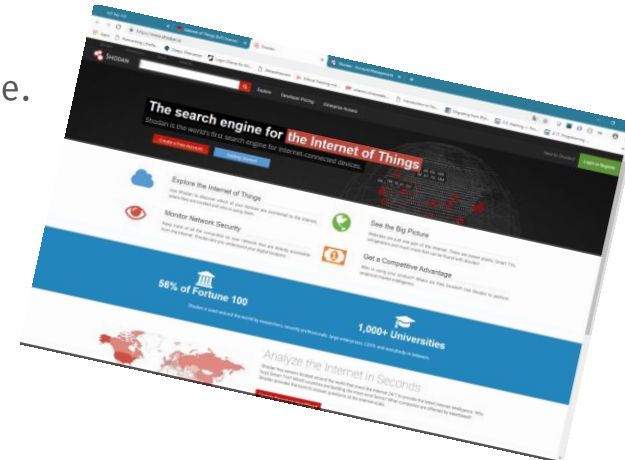
Chapter 1: The IoT Under Attack

IoT Security 1.0 v2.0



Chapter 1 - Sections & Objectives

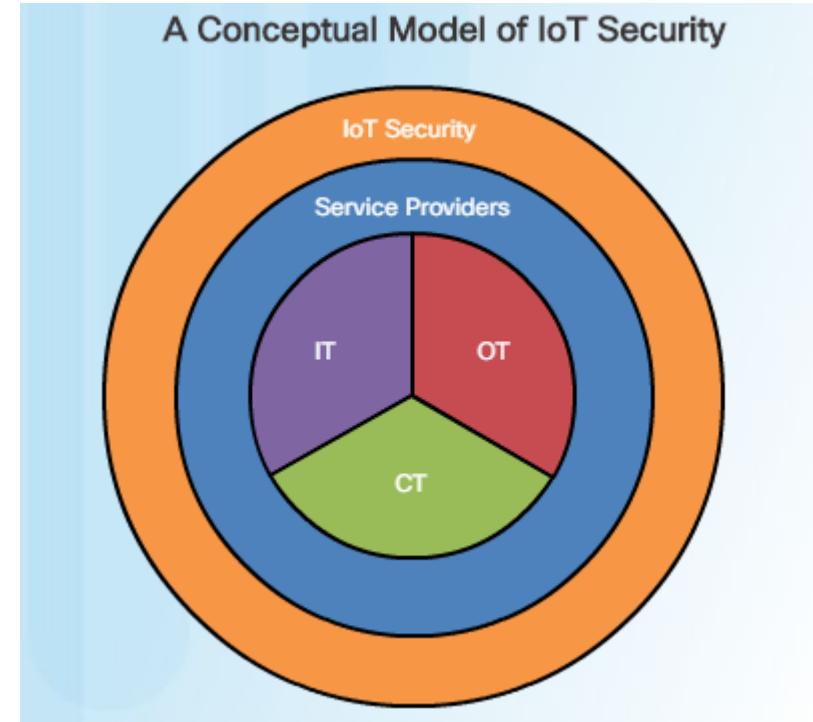
- 1.1 Explain the need for IoT security in several IoT environments.
 - Explain why security should be a focus of the IoT
 - Explain how the unique security risks of the IoT differ from standard IT security.
- 1.2 Evaluate potential risks in various IoT use cases.
 - Explain the unique security requirements of the IoT in the smart home.
 - Explain the unique security requirements of the IoT in healthcare.



Chapter 1 - Sections & Objectives

The Unique IoT Risk IoT Security Model

- Whether the IoT device belongs to IT, OT, CT, or some combination of the three, strong security is required.
- Service providers are organizations that connect our devices to the Internet.
 - They are in a position to offer services to address the IoT security needs of their clients.



Chapter 1 - Sections & Objectives

Microwelle in einem Botnet ?

- IoT security includes devices and applications from information technology (IT), operational technology (OT), and consumer technology (CT).
-
- **IT** - includes devices in the data center, in the cloud, bring your own devices (BYODs), and thousands of sensors and actuators connected in the field
- **OT** - includes industrial control systems (ICSs), supervisory control and data acquisition (**SCADA**) systems, and all the devices that connect to these systems
- **CT** - includes connected devices in the home, wearable technology, smart cars, and more

Chapter 1 - Sections & Objectives

Use Case:

Attacking a Smart Home WiFi, by all wifi-devices lose connectivity!

• In general, the security requirements for a smart home should include the following:

• **WPA2** - The wireless network should use the latest Wi-Fi security, which is currently WPA2.

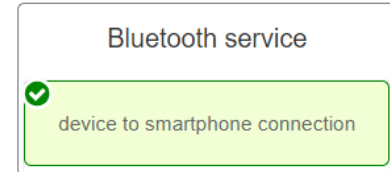
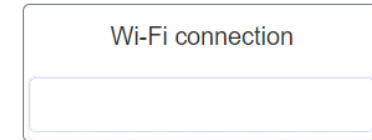
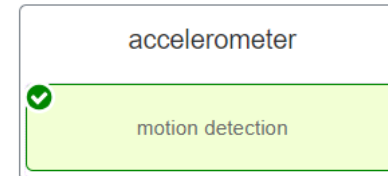
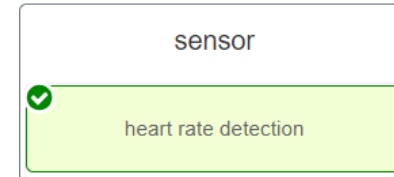
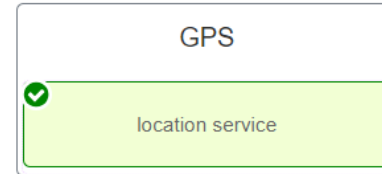
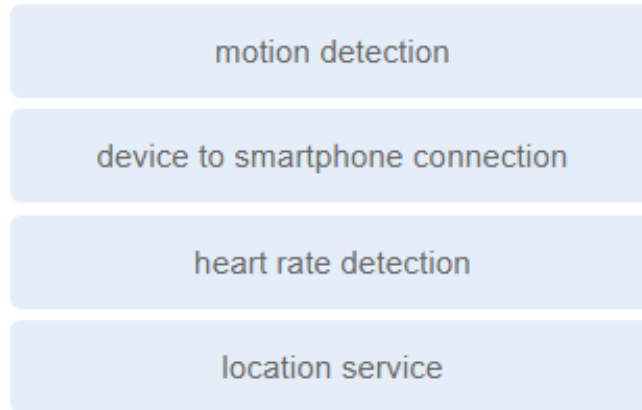
• **Encryption** – It protects the confidentiality and integrity of information transmitted over a network.

• **Authentication** - Strong authentication protects the device from unauthorized use or reconfiguration and prevents disclosure or modification of the data stored on the device.

• **Firmware** – The IoT device manufacturers should update the firmware for any newly discovered vulnerabilities. The home IoT device users should enable the checking of updates automatically.

Chapter 1 - Sections & Objectives

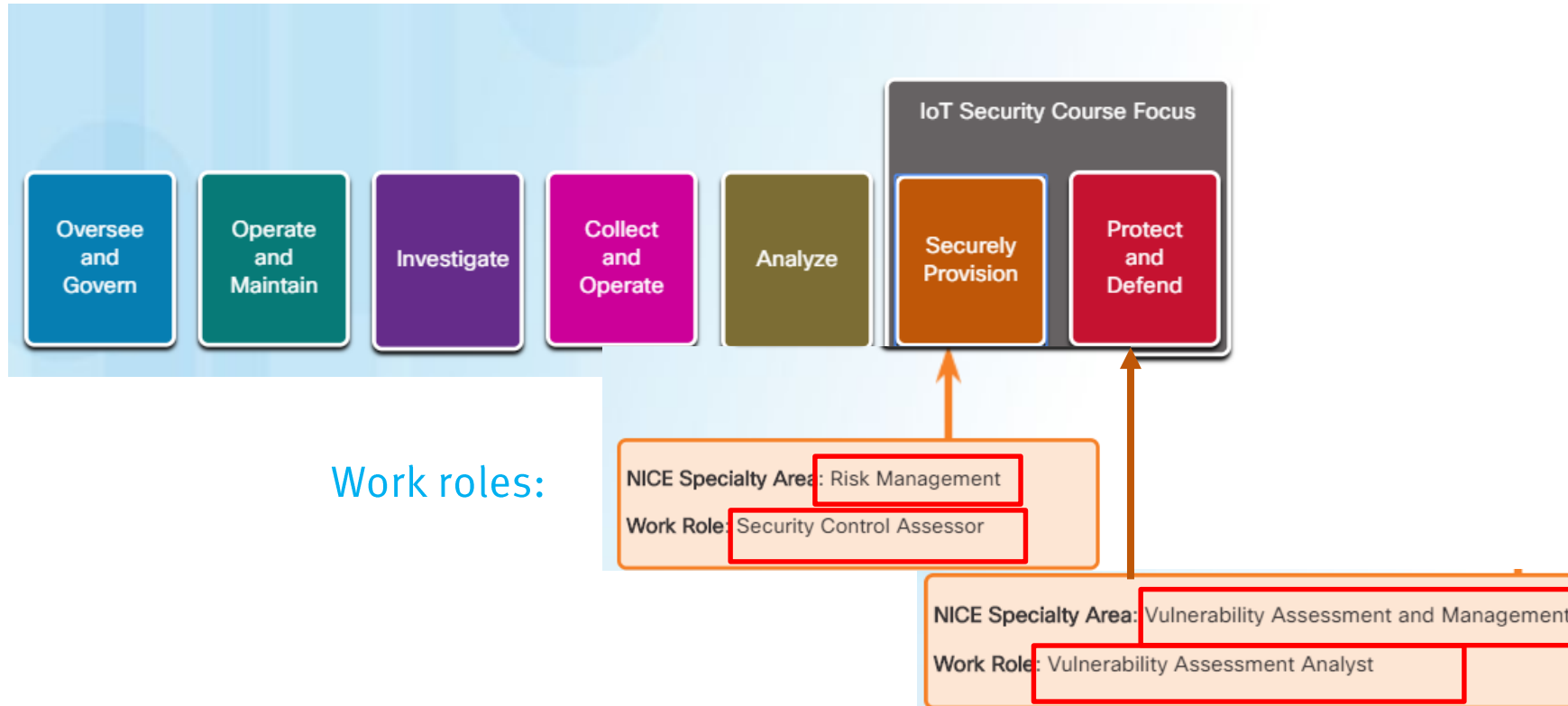
Use Case: Healthcare, like a smart fitness device



Chapter 1 - Sections & Objectives

NICE Cybersecurity Workforce Framework

National Initiative for Cybersecurity Education





Chapter 2: IoT Systems and Architectures

IoT Security 1.0

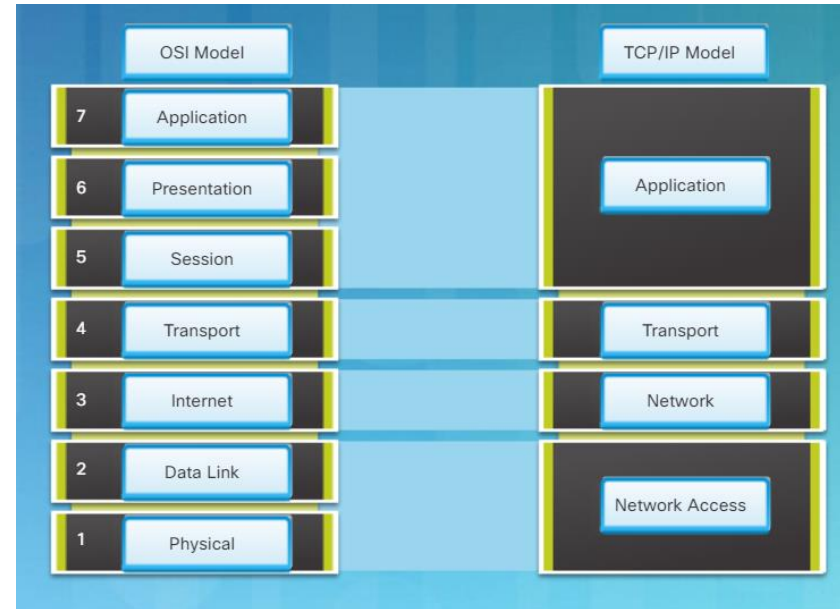


- 2.1 Use Industry standard models to explain IoT systems.
 - Explain the value of IoT industry standards.
 - Explain the value of IoT industry standard models.
- 2.2 Use a common model to explain IoT Security.
 - Explain how a layered security model is useful in understanding IoT security requirements.
- 2.3 Create an IoT threat model.
 - Explain the cybersecurity job roles.
 - Explain how threat models are constructed.

Networking Models and Layers

New Model

Intention of the IoT reference model!



The intent of the IoT reference model **is to provide common** terminology and help clarify how information flows and is processed for a unified IoT industry.

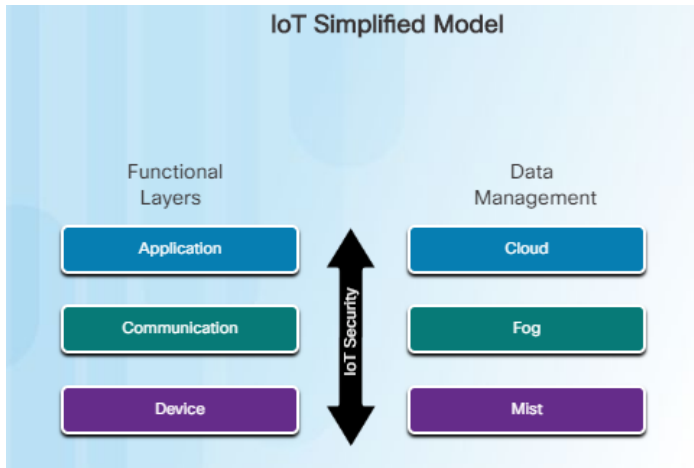
It is an **ETSI** standard.

IoT Reference Model

Level	Description	
7	Collaboration & Processes (Involving people and business processes)	Transcends multiple applications to include the communication and collaboration required between people and business processes.
6	Application (Reporting, analytics, control)	Information interpretation based on the nature of the device data and business needs.
5	Data Abstraction (Aggregation and access)	Focused on rendering the data and its storage in ways to enable application development.
4	Data Accumulation (Storage)	Data in motion is converted to data at rest. The data is also transformed so that it can be consumed by upper levels.
3	Edge (Fog) Computing (Data element analysis and transformation)	Converts data into information that is suitable for storage and higher level processing.
2	Connectivity (Communication and processing units)	Responsible for reliable and timely data transmission between devices and the network, across networks, and between the network and data processing in Level 3.
1	Physical Devices & Controllers (The "Things" of IoT)	Includes a wide range of endpoint devices that send and receive information.

IoT Security Layers

A Simple IoT Model

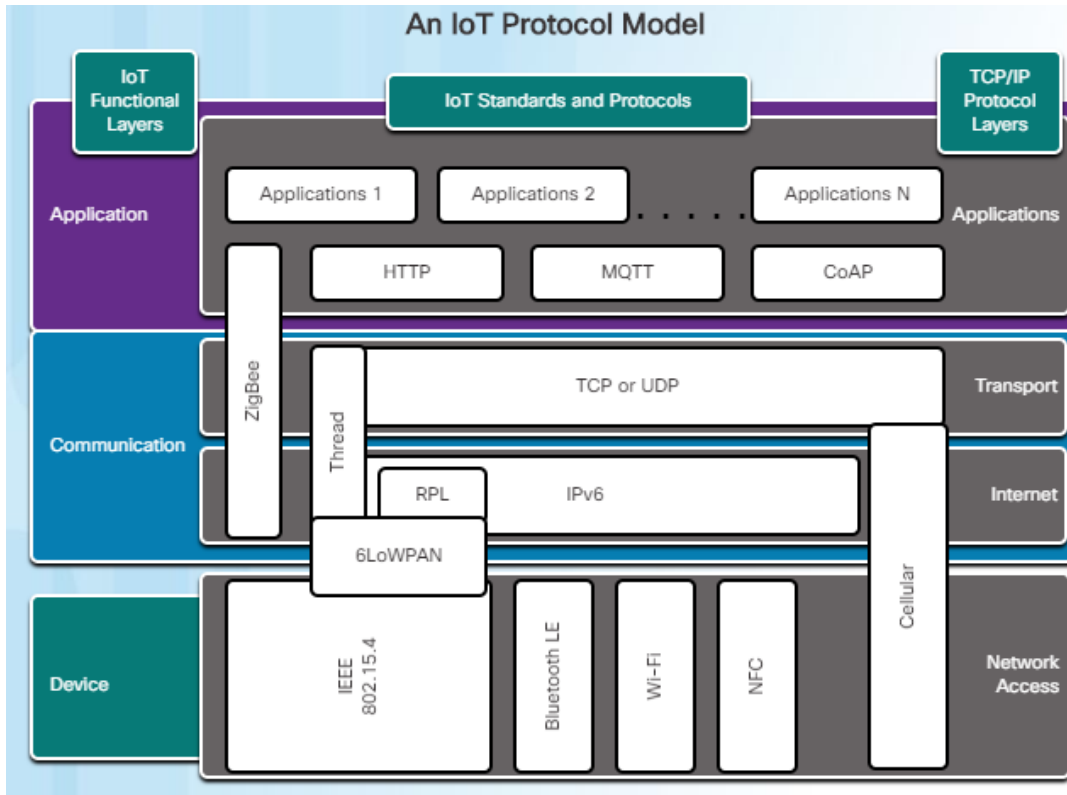


- **Domains - Application, Communication, and Device layers.**
 - **Device** layer of an irrigation system might include individual sprinkler heads, moisture sensors, temperature sensors, and actuators.
 - **Communication** layer, these devices might all be connected to a local irrigation control panel that monitors the state of the system.
 - **Application** layer, the control panel may be connected to a remote data center where all the control panels for multiple irrigation systems are aggregated.
- **For data management, interested in when and where data is processed.**
 - **Mist layer**, close to the ground where things are connected to the network.
 - **Fog** layer on a local device that has more power, such as irrigation system's control panel.
 - Can a supervisor remotely override the autonomous actions of the control panel using a mobile or desktop application in the **Cloud**?

IoT Security Layers

IoT Security Model

Matching the IoT standard or protocol with a category.



- This course uses a combination of the functional layers of the IoT simplified model overlaid with the TCP/IP model.

Application

- **ZigBee**,
- Hypertext Transfer Protocol (HTTP/HTTPS), Message Queuing Telemetry Transport (MQTT),
- **Constrained Application Protocol (CoAP)**

Communication

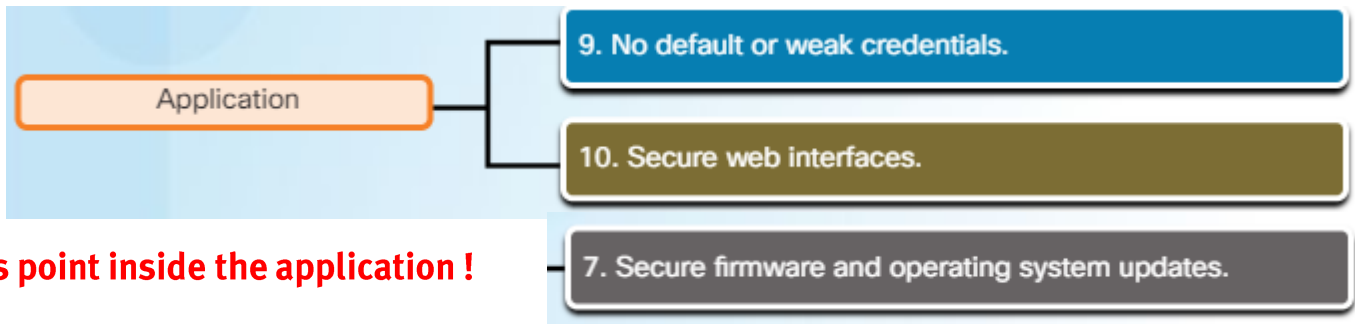
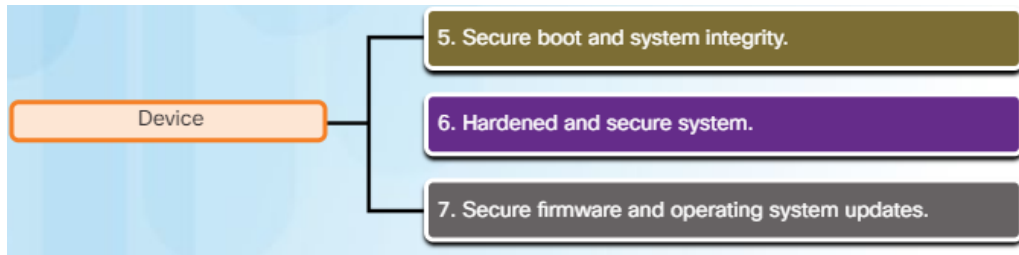
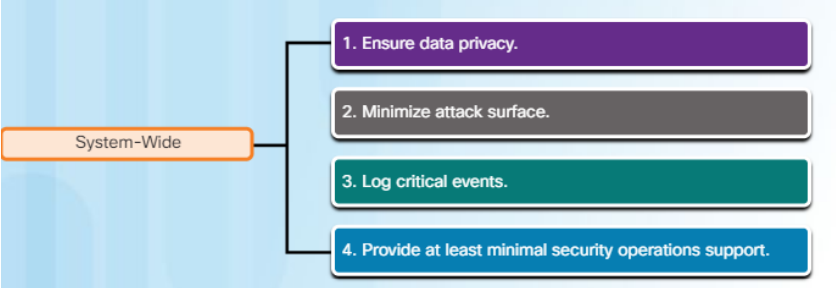
- Thread, Transport Control Protocol (TCP), UDP, IPv6,
- **RPL, thread**

Device

- **6LoWPAN, IEEE 802.15.4**,
- Bluetooth Low Energy (BLE), Wi-Fi, Near Field Communication (NFC), Cellular

Security requirement with the appropriate layer of the IoT functional model.

Ten Critical IoT Security Requirements

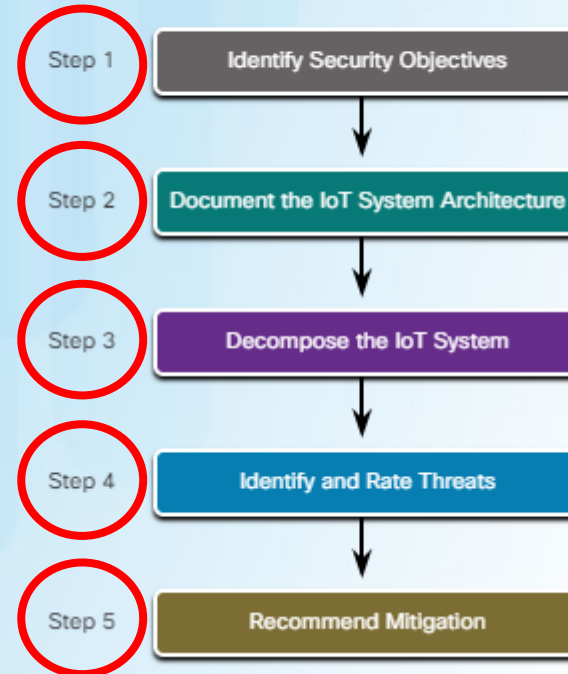


Final expected this point inside the application !

Threat Model Analysis for an IoT System

- Threat modeling - tool used to conduct tasks for risk management and vulnerability assessments.
- Structured approach for analyzing the security and vulnerability of a system, whether that system be a **device's hardware, software, or the networks used to communicate** with other devices.

Threat Modeling Process for Vulnerability Assessment



DREAD is an acronym that stands for the variables used to quantify, compare, and prioritize the amount of risk in each threat:

DREAD Risk Rating = (**D**amage + **R**eproducibility + **E**xploitability + **A**ffected Users + **D**iscoverability) / 5

STRIDE is a vulnerability assessment tool used to identify threats. STRIDE is an acronym that stands for the following categories of threats:

- **S**poofing Identity
- **T**ampering with Data
- **R**epudiation
- **I**nformation Disclosure
- **D**enial of Service
- **E**levation of Privilege



Chapter 3: The IoT Device Layer Attack Surface

IoT Security 1.0 v2.0



Chapter 3 - Sections & Objectives



- 3.1 Explain the operation of IoT device hardware and firmware.
 - Explain the operation of IoT device hardware components.
 - Explain the operation of IoT device software components.
- 3.2 Perform threat modeling activities to evaluate IoT device hardware and firmware.
 - Perform threat modeling activities to evaluate IoT device hardware.
 - Perform threat modeling activities to evaluate IoT device firmware.
- 3.3 Recommend measures to mitigate threats to IoT devices.
 - Recommend measures to mitigate threats at the device layer.
 - Recommend measures to mitigate protocol security threats on an IoT Device.

- Hardware Sensors
 - Environment manipulation
 - Tampering
 - Damage
- Potential vulnerabilities towards device Memory
 - Default username and password
 - Sensitive data
 - Plaintext usernames and passwords
 - Encryption keys
- Device Physical Interfaces
 - Removal of storage media
 - Reset to insecure state
 - Device ID/Serial number
 - Serial interface connections
 - User and Administrative access
 - Privilege escalation
- Device Firmware
 - Backdoor Accounts
 - Hardcoded credentials
 - Encryption keys
 - Firmware version display
 - Firmware version last update date
 - Vulnerable services
 - Security related function API exposure
- Firmware Update Mechanism
 - Update sent without encryption
 - Updates not signed
 - Update location writable
 - Update verification and authentication
 - Malicious update
 - Missing update mechanism
 - No manual update mechanism

IoT devices typically use a trimmed down version of an operating system. Developers can choose from open source and commercial options.

Busybox

is an open source compiled executable that contains many of the core utilities that are usually found in Linux distributions. Most of the utilities are scaled-down versions that have fewer options than their full equivalents.

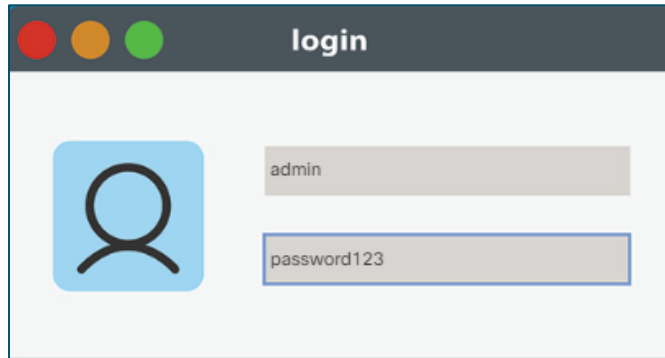
command

```
Type 'busybox' to see the list of available commands.
# busybox
BusyBox v1.20.0 (2012-04-22 12:29:58 CEST) multi-call binary.
Copyright (C) 1998-2011 Erik Andersen, Rob Landley, Denys Vlasenko
and others. Licensed under GPLv2.
See source distribution for full notice.

Usage: busybox [function] [arguments]...
or: busybox --list[-full]
or: busybox --install [-s] [DIR]
or: function [arguments]...
```

Firmware Vulnerabilities

- IoT devices require firmware to run.
- Firmware is basically embedded software that contains a minimal operating system and related programs to control the IoT device.
- IoT device firmware can contain security vulnerabilities that are discovered after their release. Firmware-related vulnerabilities for IoT devices are similar to those of other computers or networking devices.



Most general IoT attacks and distributed denial-of-service (DDoS) attacks are made possible via the use of default or weak login credentials.

- Constrained devices are often placed in remote locations where physical security may be difficult to implement.
- Potential vulnerabilities could include:
 - Theft of the device.
 - Physical damage to the device.
 - Disabling the device, removing power source.
 - Disabling communication, disconnecting cables or other means of disruption.
 - **SD-card could be stolen**
- Provide some type of video surveillance where possible.
- **Provide a tamper proof enclosure or tamper resistant type of housing.**



Network Access Control Concepts

Access Control Models



- A security analyst should be familiar with different basic access control models to have a better understanding of how threat actors can break the access controls.
- **Mandatory access control (MAC)** - Applies the strictest access control and is typically used in **military or mission critical applications**. Assigns security level labels to information and provides users with access based on their security level clearance.
- **Discretionary access control (DAC)** - It allows users to control access to their data as owners of that data.
- **Non-Discretionary access control** - Access decisions are based on an individual's roles and responsibilities within the organization, also known as role-based access control (RBAC).
- **Attribute-based access control (ABAC)** - Allows access based on attributes of the object (resource) to be accessed, the subject (user) accessing the resource, and environmental factors regarding how the object is to be accessed, such as time of day.
- **Principle of least privilege** - users should be granted the minimum amount of access required to perform their work function.
- **Privilege escalation exploit** - vulnerabilities in servers or access control systems are exploited to grant an unauthorized user, or software process, higher levels of privilege than they should have.

Encryption

Public Key Cryptography

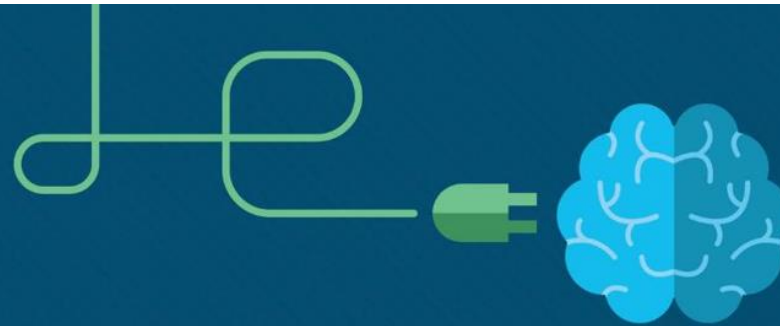


- **Cryptography** - based on the sender and receiver of a message knowing and using the same secret key.
 - **Symmetric Cryptography** - sender uses the secret key to encrypt the message, and the receiver uses the same secret key to decrypt the message.
 - **Because all keys in a secret-key cryptosystem must remain secret, the challenge has been providing secure key management.**
- **Public-key cryptography** was introduced in 1976 by Whitfield Diffie and Martin Hellman in order to solve the secure key management problem.
 - Each person gets a pair of keys: one called the public key and the other called the private key.
 - Each person's public key is published while the private key is kept secret.
 - With this system, anyone can send a confidential message by using public information (public key of the recipient), but the message can only be decrypted using the private key of the intended recipient.
 - Can be used not only for privacy (encryption), but for authentication (digital signatures).



Chapter 4: IoT Communication Layer Attack Surface

IoT Security 1.0 v2.0



Chapter 4 - Sections & Objectives



- 4.1 Determine vulnerabilities of the IoT communication layer.
 - Explain the functions of the IoT network communication layer.
 - Determine vulnerabilities in IoT wireless network protocols.
- 4.2 Determine vulnerabilities in TCP/IP that impact IoT systems.
 - Determine vulnerabilities in IP that impact IoT systems.
 - Determine vulnerabilities in TCP and UDP that impact IoT systems.
- 4.3 Propose measures to mitigate threats at the IoT network layer.
 - Implement access control in IoT networks.

OWASP Communication Layer Vulnerabilities

Attack Surface	Vulnerability
Device Network Services	Information disclosure
	Injection
	Denial of Service
	Unencrypted Services
	Poorly implemented encryption
	Test/Development Services
	Vulnerable UDP Services
	DoS
	Replay attack
	Lack of payload verification
Lack of message integrity check	
Network Traffic	LAN traffic
	LAN to Internet traffic
	Short range
	Non-standard protocols
	Wireless (Wi-Fi, Z-wave, XBee, Zigbee, Bluetooth, LoRA)
	Packet manipulation (protocol fuzzing)

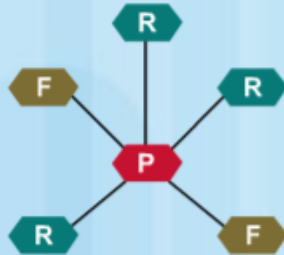
Communication Channels in IoT wireless network

Wireless Network	Use Case
WBAN: Wireless Body Area Network	A network of wireless sensor devices that are either worn or implanted into the body. May use various wireless protocols to communicate with a gateway to post data to cloud applications.
WPAN: Wireless Personal Area Network	Frequently employs Bluetooth to connect audio devices, personal fitness trackers, and smart watches to a cell phone that serves as a gateway.
WFAN: Wireless Field (or Factory) Area Network	Ruggedized network components connect sensors and actuators at dispersed locations in challenging manufacturing environments.

IEEE 802.15.4 Topologies

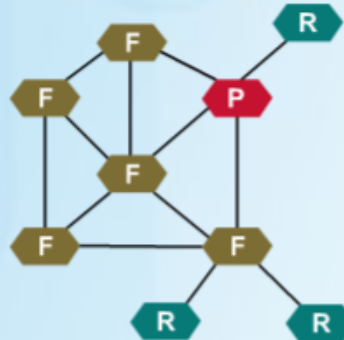
Operates at Layer 2

Star Topology



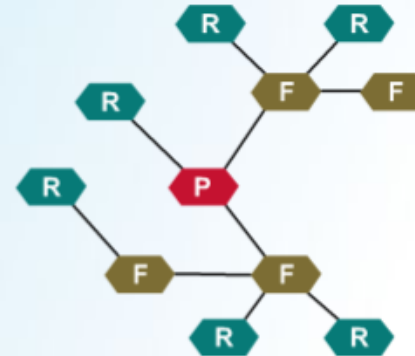
- All devices communicate to PAN coordinator which uses fixed power
- Other devices can be battery/scavenger

Mesh Topology



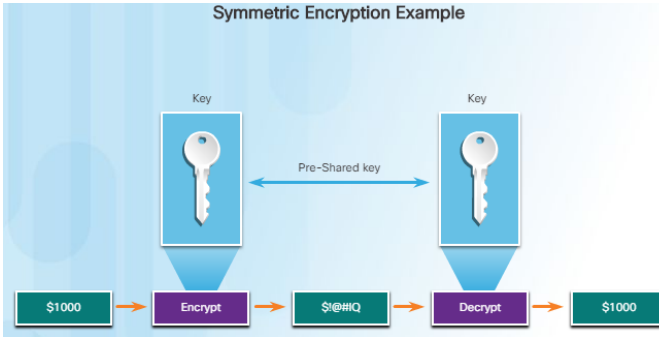
- Devices can communicate directly if within range

Cluster Tree



- Higher layer protocols like RPL may create their own topology that do not follow 802.15.4 topologies

- **This allows to operate independently in Home-Automation.**

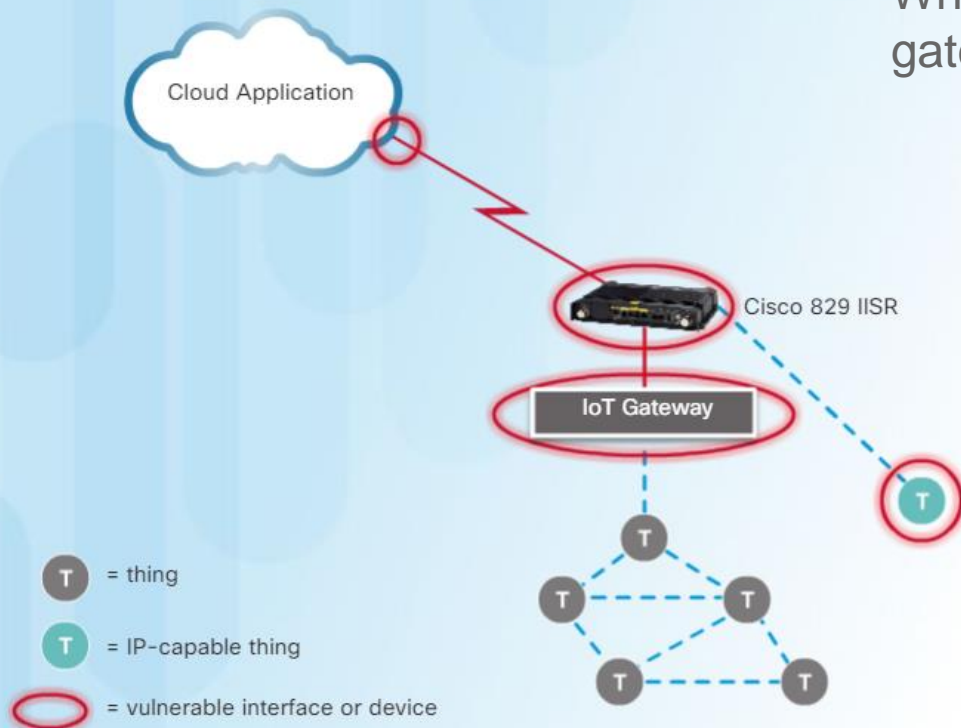


- Because 802.15.4 operates at the OSI physical and data link layers security of the frames is important.
- Four basic security services performed at the data link layer:
 - Access control – Prevents unauthorized devices from joining the network.
 - Message integrity – Protects against alteration of data while it is in transit by using an encrypted cryptographic key (message authentication code).
 - Message confidentiality – Prevents threat actors from reading the transmitted data. Message data payloads are encrypted to protect the confidentiality of the message.
- Replay protection - Legitimate messages can be captured and sent out on the network at a later time from a previous M2M session. If these messages are replayed frequently, network performance can be degraded to the extent that legitimate data cannot reach the gateway.
- 802.15.4 uses symmetric key cyphers for encryption. Symmetric keys are less secure than asymmetric, or public key, cryptography.

IoT Communication Layer Attack Surfaces

Why do some IoT devices use a gateways?

- many IoT devices do not support a full TCP/IP stack
- These devices rely on gateways, which provide IP transmission services, to send sensor data for processing.



IP Vulnerabilities

Common IP Vulnerabilities

These are some of the more common IP-related attacks:

- **DoS attacks** - Threat actors attempt to prevent legitimate users from accessing information or services.
- **DDoS attacks** – This attack is similar to a DoS attack, but features a simultaneous, coordinated attack from multiple source machines.
- **ICMP attacks** - Threat actors use Internet Control Message Protocol (ICMP) echo packets (pings) to discover subnets and hosts on a protected network, to generate DoS flood attacks, and to alter host routing tables.
- **Address spoofing attacks** - The threat actor puts the source IP address in a packet to masquerade as a different source, tricking the destination into believing the packet came from a legitimate source.
- **Man-in-the-middle attack (MITM)** - Threat actors position themselves between a source and destination to transparently monitor, capture, and control the communication. They could simply eavesdrop by inspecting captured packets or alter packets and forward them to their original destination.
- **Session hijacking** - Threat actors gain access to the physical network, and then use an MITM attack to sniff a valid token for access to a web server.

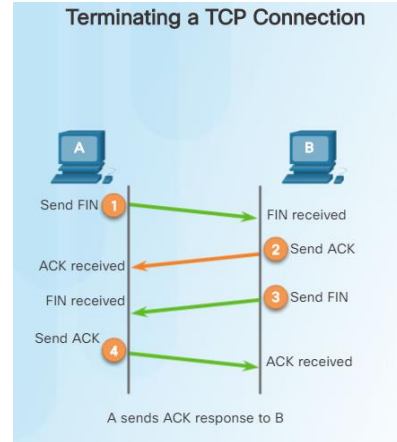
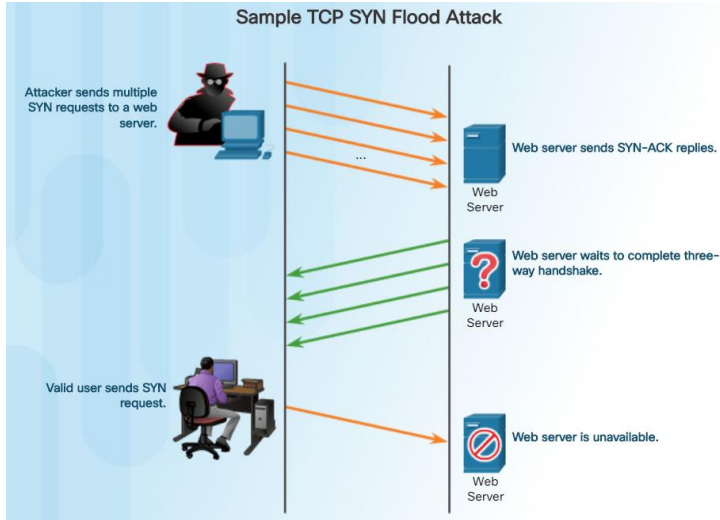
IP Vulnerabilities

Common IP Vulnerabilities

IP address spoofing attacks occur when a threat actor creates packets with false source IP address information. These packets help the threat actor to either hide the identity of the sender or to pose as another legitimate user. Spoofing is usually incorporated into another attack such as a Smurf attack. Spoofing attacks can be conducted as follows:

- **Non-blind spoofing** - The threat actor can see the traffic that is being sent between the host and the target. Non-blind spoofing is used by the threat actor to inspect the reply packet from the target victim. Reasons for non-blind spoofing include determining the state of a firewall, TCP sequence-number prediction, or hijacking an authorized session.
- **Blind spoofing** - The threat actor cannot see the traffic that is being sent between the host and the target. Blind spoofing is used in DoS attacks.

TCP Vulnerabilities



A TCP reset attack can be used to terminate TCP communications between two hosts by sending a **spoofed TCP RST packet**. A TCP connection is torn down when it receives an RST bit.

Port Addressing



To: you@example.com
 From: me@example.com
 Subject: Email



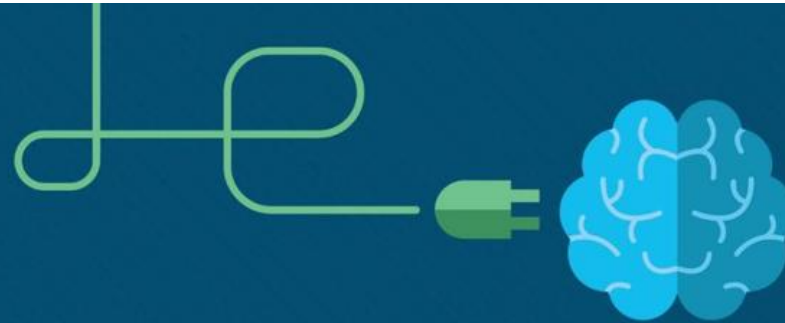
Different Applications	Electronic Mail	HTML Page	Internet Chat
Protocols	POP3	HTTP	IM
Transport	Application Port Data	Application Port Data	Application Port Data
Port Numbers	110	80	531

Network applications use TCP or UDP ports.
 Threat actors conduct port scans of target devices to discover which application services are active.



Chapter 5: IoT Application Layer Attack Surface

IoT Security 1.0 v2.0



Chapter 5 - Sections & Objectives



- 5.1 Perform vulnerability assessment activities of IoT applications and protocols.
 - Perform vulnerability assessment activities of the IoT local applications.
 - Perform vulnerability assessment of IoT remote applications.
 - Perform vulnerability assessment of IoT application messaging protocols.

- 5.2 Recommend measures to mitigate threats to IoT applications.
 - Recommend measures to mitigate threats to IoT messaging protocols.

OWASP Application Vulnerabilities



- According to the Open Web Applications Security Project (OWASP), the most widely exposed vulnerabilities are these:**Username enumeration** – The threat actor is able to find valid usernames through the authentication application.

- Weak passwords** – Threat actors use default passwords that have not been changed or are able to set account passwords that they choose. Most widely exposed.

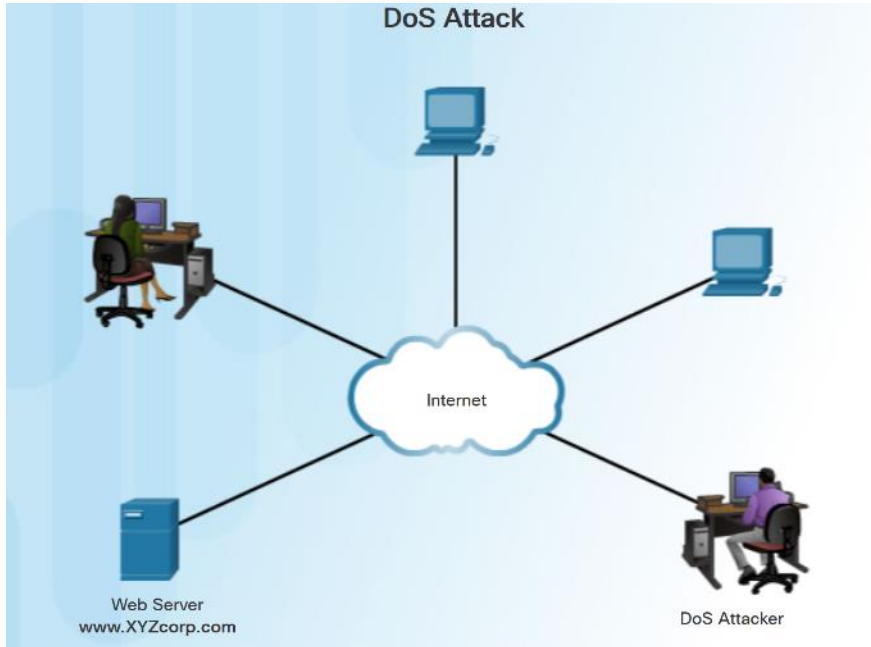
- Account lockout** – The threat actor finds a way to attempt to authenticate many times after multiple failed attempts.

- Lack of multi-factor authentication** – It is easier for a threat actor to gain access when only one form of authentication is required.

- Insecure 3rd party components** – As vulnerabilities are discovered, they often become patched. When components such as Secure Shell (SSH), BusyBox, or web servers are not kept up to date, the threat actor might expose these vulnerabilities and gain access.

IoT Local Application Vulnerabilities

Local Applications

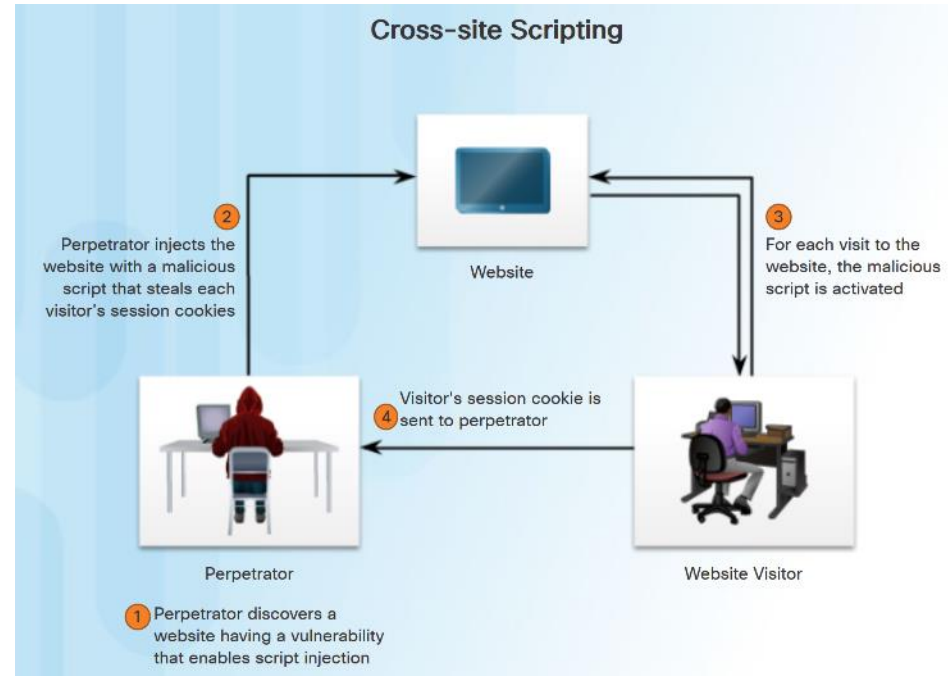


- Some of the most popular **local exploits**:
 - **Firmware Replacement**
 - **Cloning**, to creating a duplicate device
 - **Denial of Service (DoS)**
 - **Extraction of Security Parameters**
- Some of the most popular **remote exploits**:
 - **Man-In-the-Middle Attack (MITM)** – The threat actor gets between devices in the system and intercepts data.
 - **Eavesdropping Attack** – When devices are being installed, the threat actor can intercept data such as security keys that are used by constrained devices.
 - **SQL Injection (SQLi)** – The threat actor uses a flaw in the Structured Query Language (SQL) application that allows access to modify data or gain administrative privileges.
 - **Routing Attack** – A threat actor could place a rogue routing device on the network.

IoT Web and Cloud Applications Vulnerabilities

Web Frontend Vulnerabilities

- Web frontend vulnerabilities apply to the apps, APIs, and services.
- The three most common web frontend vulnerabilities:
 - **Cross-Site Scripting** - In a XSS attack, the threat actor **injects code**, most often JavaScript, mitigate this attack by escaping;validating;santizing
 - **SQL Injection** - SQLi is where the threat actor injects **malicious commands/code** into fields that will be used to query the SQL database for unauthorized records.
 - **Broken Authentication** - A threat actor can hijack a session to assume the identity of a user when session tokens are left unexpired.



Web Frontend Vulnerabilities



- There are three best practices to prevent XSS in applications:
- **Escaping** – This censors the data that the web page receives. It ensures the data that the application receives is secure before it is rendered for the end user.
- **Validating Input** – Whitelisting can be used to allow only known good characters into the application. This is especially effective at preventing XSS in forms because the user cannot add special characters in the fields.
- **Sanitizing** – Remove potentially harmful markup from any input. This is especially useful when a site allows HTML markup.



Chapter 6: Vulnerability and Risk Assessment in an IoT System

IoT Security 1.0



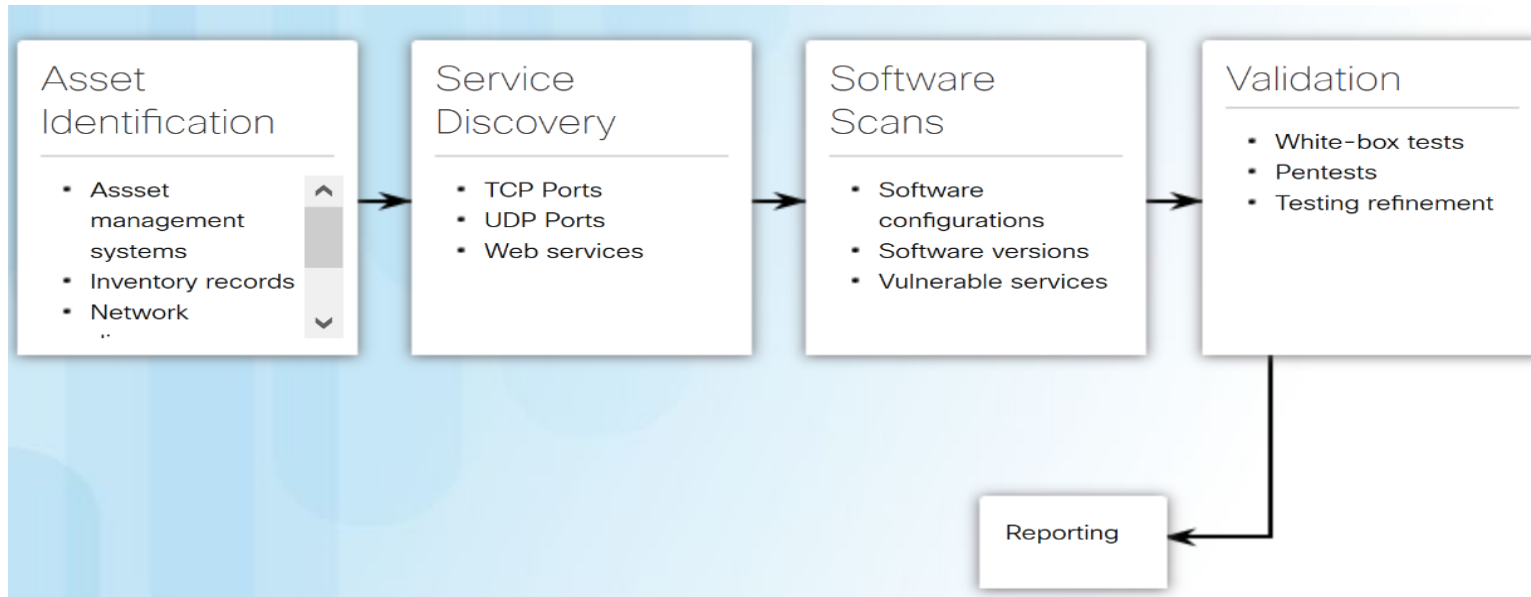
Chapter 6 - Sections & Objectives



- 6.1 Explain how vulnerabilities are assessed in IoT Systems.
 - Explain how security vulnerabilities are assessed.
 - Explain how tools and services are used to assess vulnerabilities in IoT systems.
- 6.2 Evaluate security in an IoT system risk using assessment.
 - Explain approaches to security risk assessment.
 - Evaluate risks in an IoT system using assessment tools.
 - Use the STRDE and DREAD models as a part of a risk assessment process.
 - Explain approaches to risk management.
- 6.3 Explain innovations in IoT Security
 - Explain the role of blockchain in IoT systems.
 - Explain how blockchain works.

Vulnerability Assessment

Vulnerability Assessment



Finally, it is important to report the vulnerabilities that were discovered and the potential levels of exposure to threats. Later, **in risk assessment**, the vulnerabilities **are translated into risks** so that they may be prioritized and addressed accordingly.

Vulnerability Testing Types and Tools

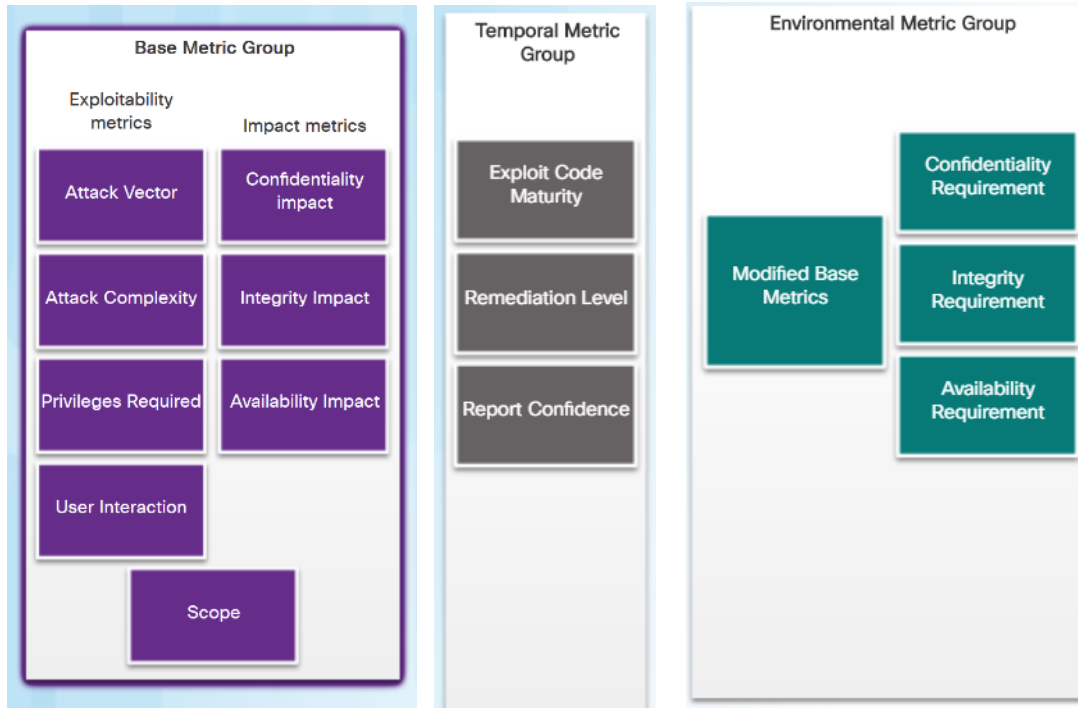
Penetration Testing

- **Port Mapping Tools**
Nmap, Netcat, or SolarWinds Port Scanner
- **Password Vulnerability Tools**
 - Brute force** - This attack is a very time consuming, inefficient, automated means of trying every possible combination of letters, numbers, and symbols to challenge logins.
 - Defeat a brute force attack?**
 - limited number of authentication failures before a specified user account is locked out
 - Dictionary attack** - This attack uses lists of words that could be used as passwords.
 - Password sniffing and cracking**
- **Web Application Vulnerability Tools**
 - OWASP
 - OpenVAS
 - Burp Suite

Risk Assessment Concepts and Approaches

CVSS Base Metric Group

Common Vulnerability Scoring System CVSS

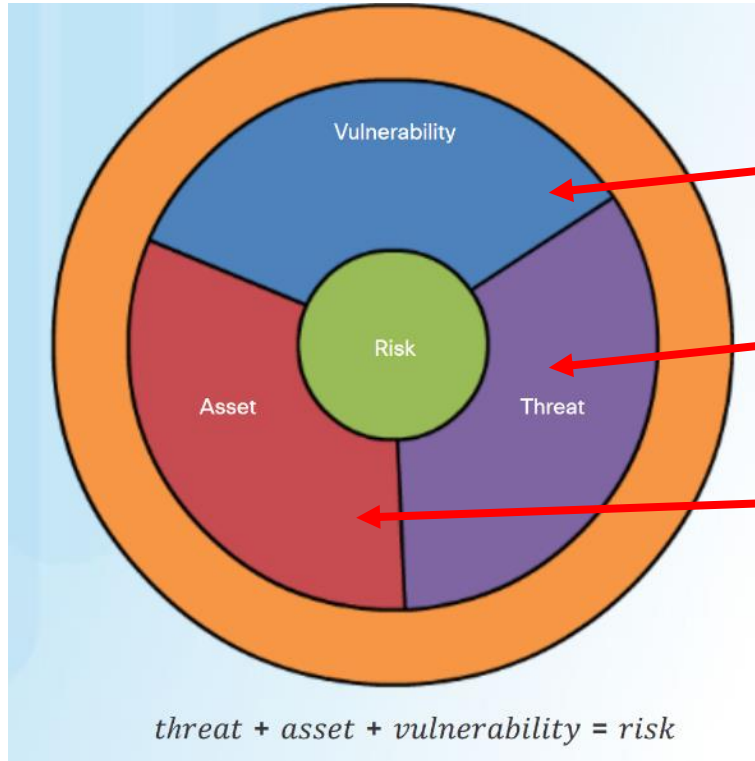


The metrics inside the Environmental metric group are optionally set by end users.

The metrics inside the Base metric group and the Temporal metric group are set by vendors.

Risk Assessment Concepts and Approaches

IoT Risk Assessment



What are weaknesses in the current security policy?

What is the likelihood that different attacks will occur?

How would the organization be affected by successful attacks?

Data Flow Diagrams Components









DFDs use 4 symbols to represent these devices.
This course uses Yourdon and Coad symbols.

- **External entity** - users, contractors, and partners outside of the control of the system that send or receive data

- **Process** - data output from sensing, actuating, traffic forwarding, analysis, and control systems

- **Data store** - data at rest in local, fog, cloud, or data center storage

- **Data flow** - single headed arrows that indicate uni-directional data flow; double headed arrows that indicate bi-directional data flow

Type	Description	Gane and Sarson	Yourdon and Coad
External Entity	Users, contractors, and partners outside of the control of the system that send or receive data		
Process	Data output from sensing, actuating, traffic forwarding, analysis, control systems		
Data Store	Data at rest in local, fog, cloud, or data center storage		
Data Flow	Single headed arrows indicated uni-directional data flow; Double headed arrows indicate bi-directional data flow		

The Promise of Blockchain

Applying traditional security methods to an IoT system is challenging due to its **decentralized topology and the limited resources of IoT devices**.

Trusted IoT Alliance, a consortium of 17 companies to help establish a standard protocol for a blockchain-based IoT security solution.

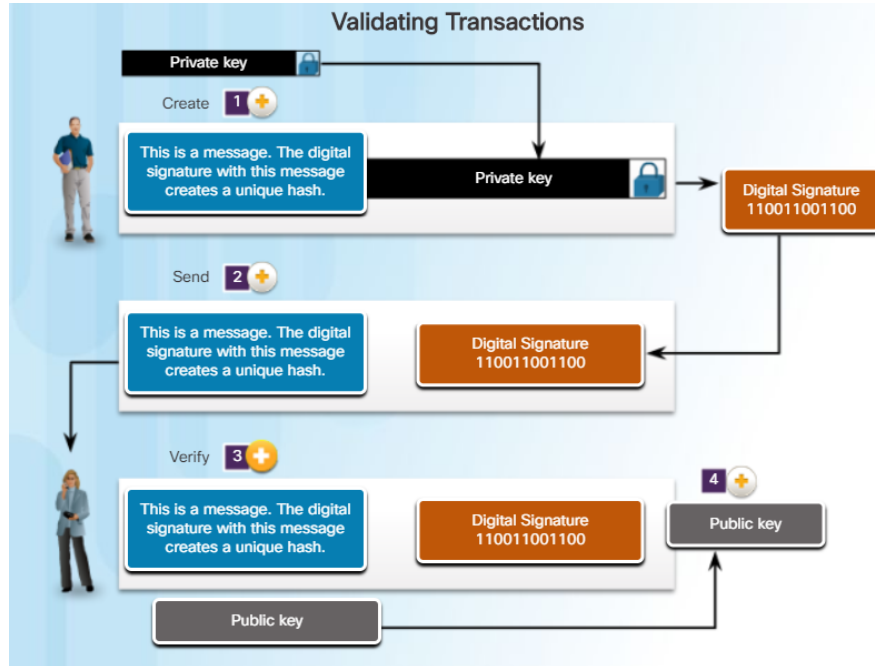
Technologies used by a blockchain

- digital signatures for authentication,
- **a decentralized ledger to track the transaction process,**
- **an algorithm for reaching consensus to verify a transaction,**
- **and each block has a hex hash of the previous block thus forming a blockchain**



Introduction to Blockchain

Reaching Consensus



Validating transactions in a block uses a process that makes it time-consuming for someone to produce, but easy for others to verify.

This is known as Proof of Work (PoW). The PoW is use an hash algorithm.



Good luck for the Final!

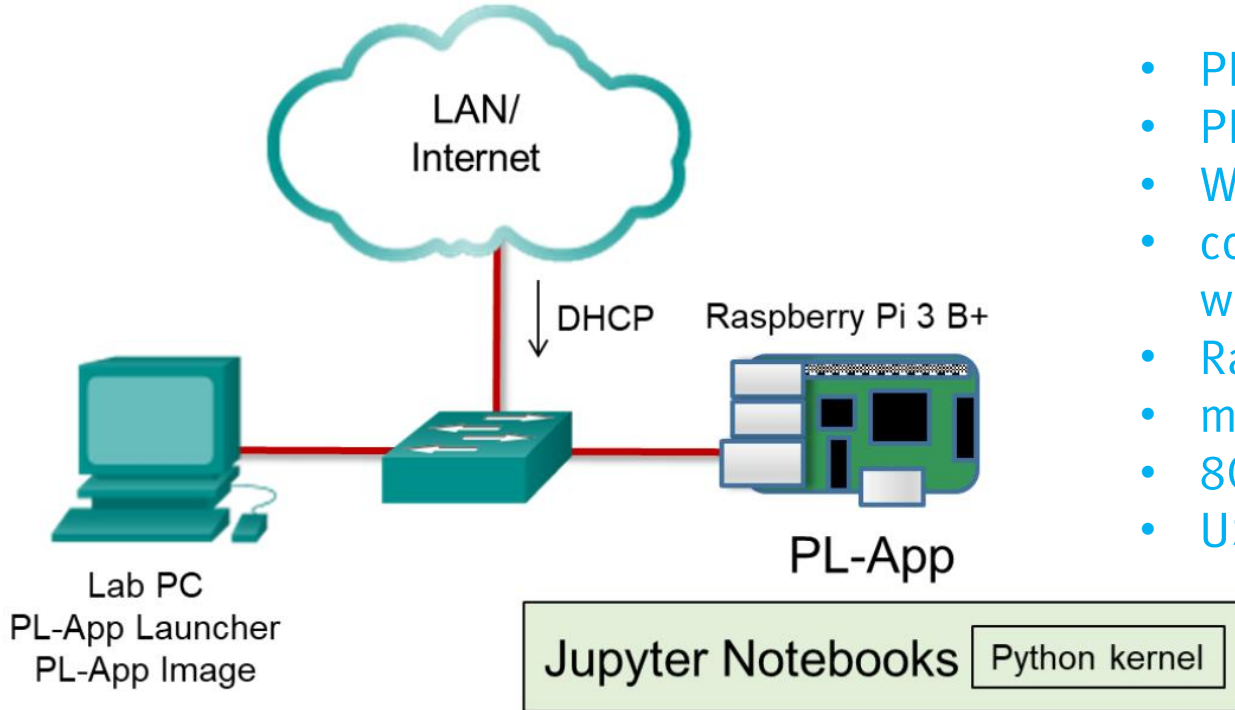
IoT Security Instructor Training Academy-Day 2019

Praktischer Teil



Three different lab topologies in the IoT security course!

Topologie 1

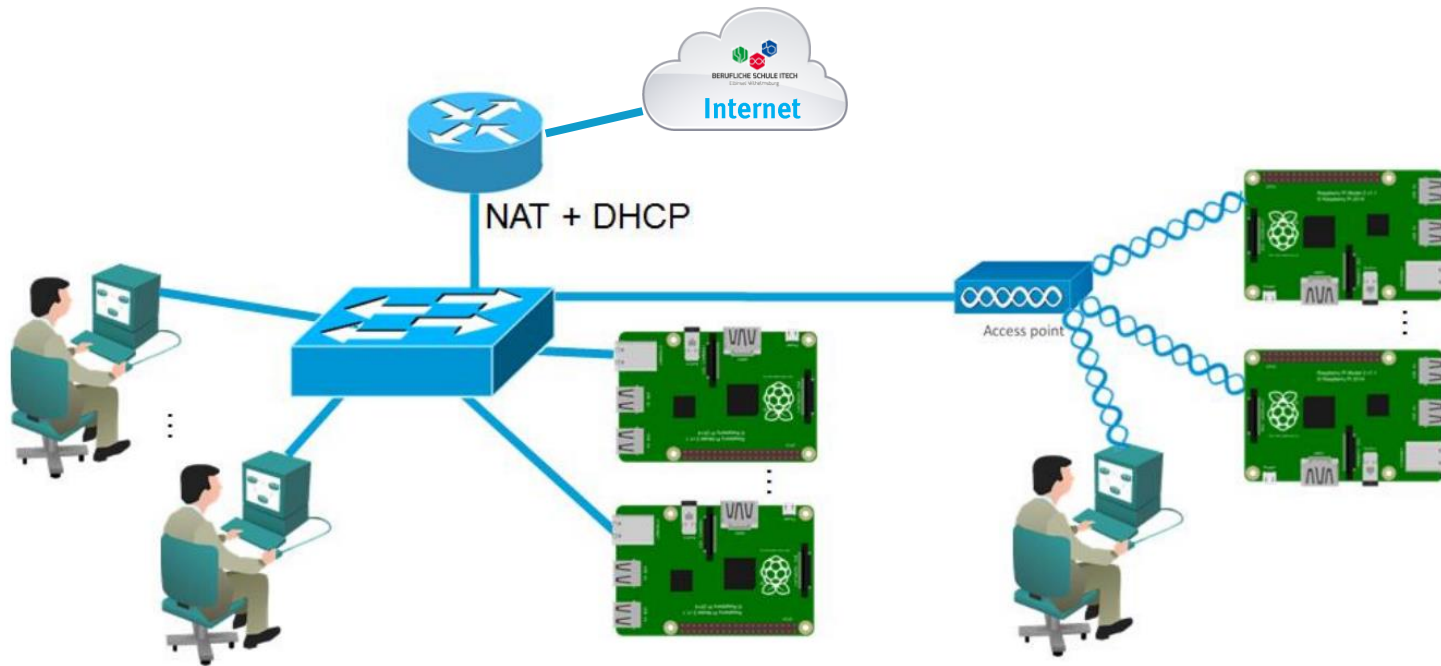


Required Resources:

- PL-App Launcher
- PL-App Image file
- Wired Ethernet or Wi-Fi
- connection local-area network with DHCP
- Raspberry Pi
- modern web browser
- 8GB μ SD
- USB to μ SD card reader

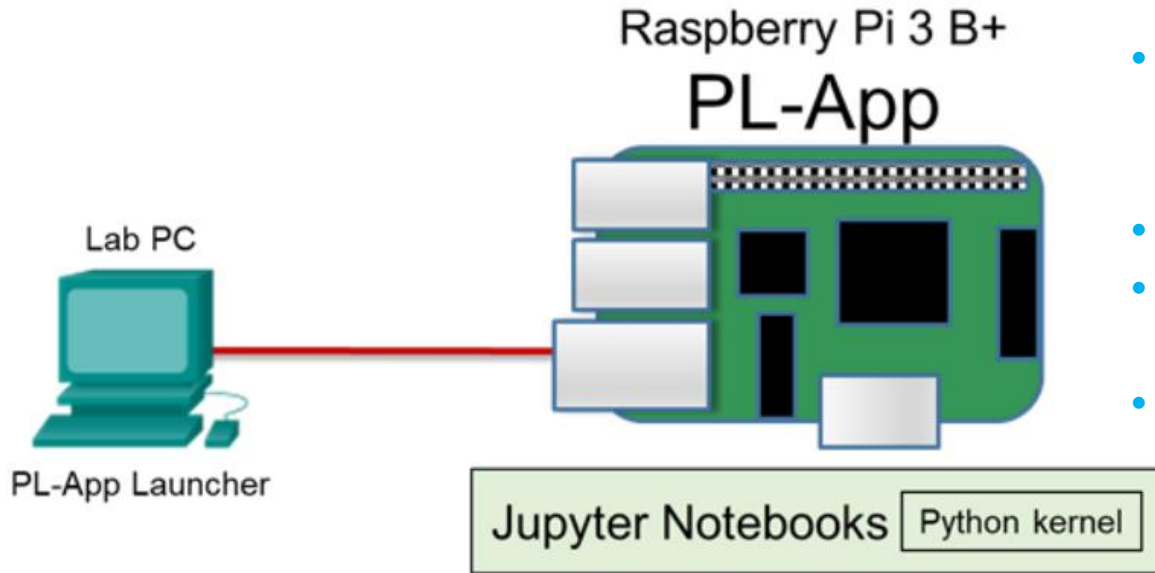
Two different lab topologies in the IoT security course!

Topology 1 classroom topology (our topology)



Two different lab topologies in the IoT security course!

Topologie 2



Required Resources:

- Host computer with at least 4 GB of RAM and 15 GB of free disk space
- Oracle VirtualBox / VMware
- IoT Security Kali Linux OVA and Metasploitable OVA files
- an Ethernet patch cable

Our workshop Lab with the Classroom topology

Preparation of raspberry, is done. Install PL-App-Launcher!

Classrom network!

The screenshot shows the 'Setup a New Device' tab of the Cisco PL-App Launcher. It contains five numbered steps:

1. Insert the SD card reader into the USB port and select it from the dropdown menu: [Dropdown menu] [Refresh]
2. Select the PL-App image file that you have downloaded from NetAcad.com: [Find Image: Browse]
3. Create a unique Device Name and Password for the PL-App device:
[i] Device Name: *Include your name or initials (e.g. jj-pi1984)*
[!] Device Password: *Password to access PL-App on your device*
4. Optional settings (connecting the device to an existing Wireless LAN):
WiFi SSID: _____
WiFi Password: _____
5. [Update Config Only] [Write Disk Image]

Version 1.5.11

The screenshot shows the 'Available Devices' tab of the Cisco PL-App Launcher. It displays a table of devices with their names, IP addresses, and connection status.

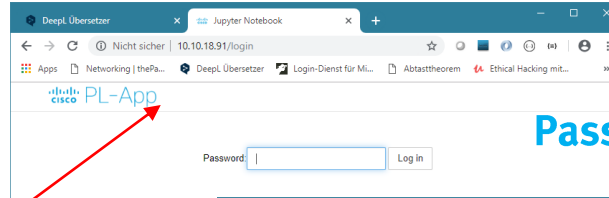
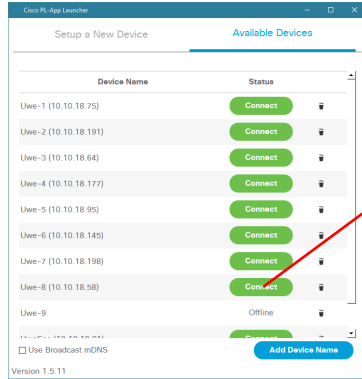
Device Name	Status
Uwe-1 (10.10.18.75)	Connect
Uwe-2 (10.10.18.191)	Connect
Uwe-3 (10.10.18.64)	Connect
Uwe-4 (10.10.18.177)	Connect
Uwe-5 (10.10.18.95)	Connect
Uwe-6 (10.10.18.145)	Connect
Uwe-7 (10.10.18.198)	Connect
Uwe-8 (10.10.18.58)	Connect
Uwe-9	Offline
Uwe-10 (10.10.18.21)	Connect

[Add Device Name]

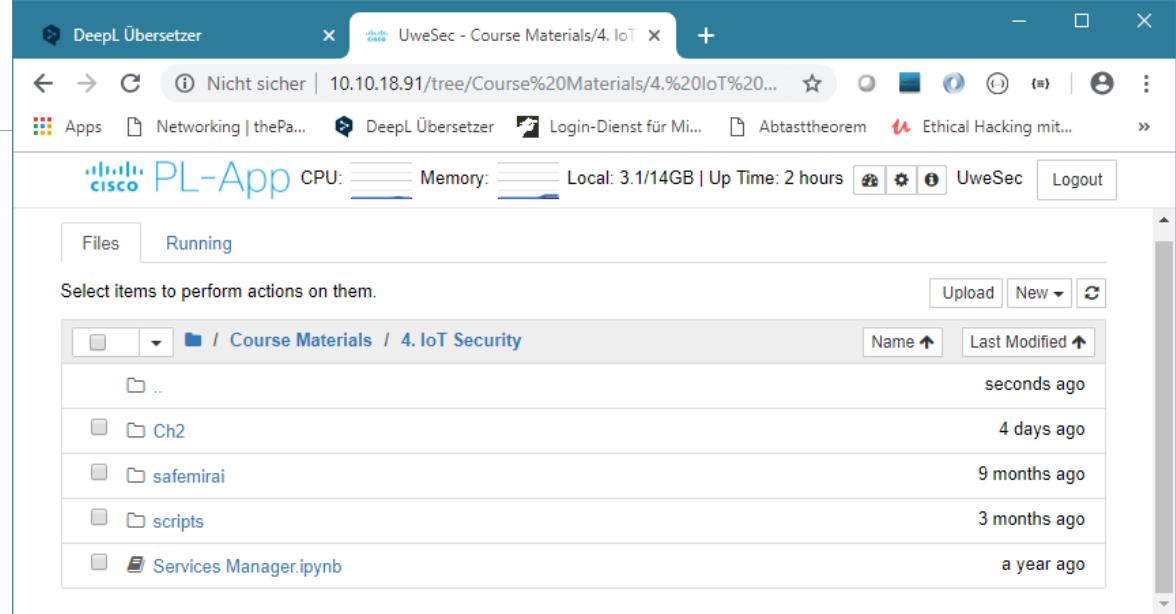
Version 1.5.11

Our workshop Lab with the Classroom topology

Try to establish the web connection to your Raspberry!



Password is „secure“ use your Raspi-Name!

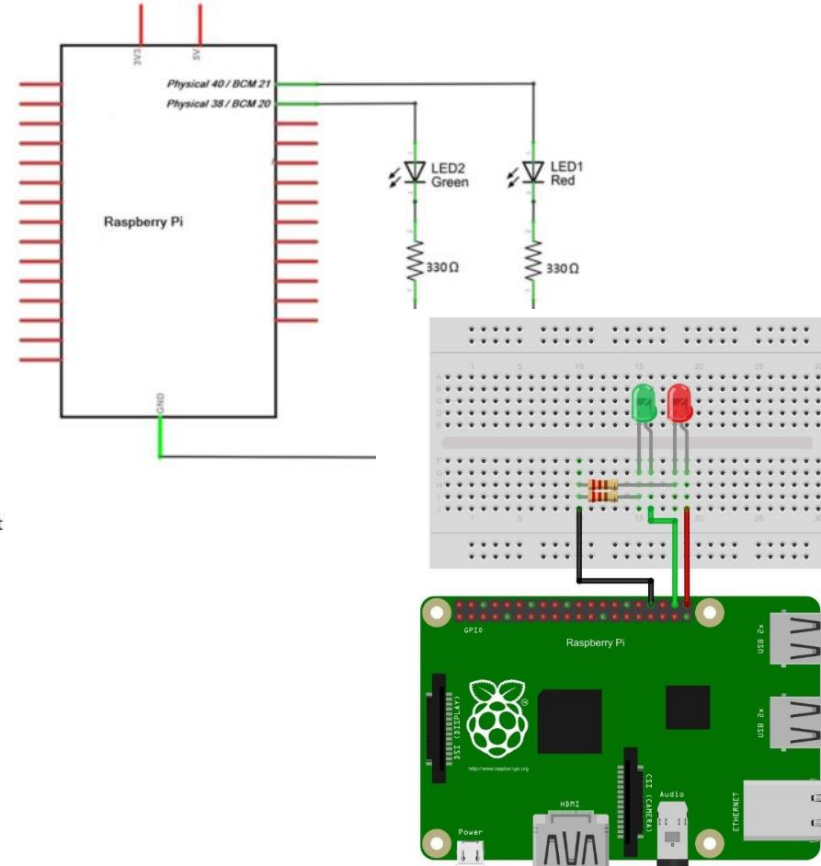
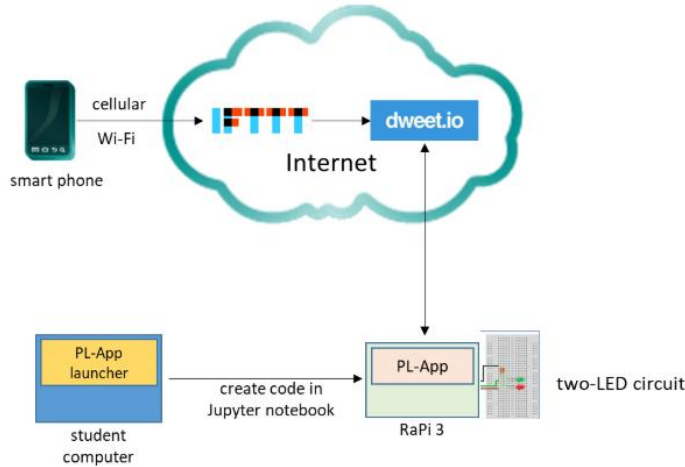


Our workshop Lab with the Classroom topology

Open Jupyter-Notebook Lab-2.2.1.4

Lab - Create an IoT Sensor-Actuator System

Topology



Our workshop Lab with the Classroom topology

Open Jupyter-Notebook Lab-2.2.1.4

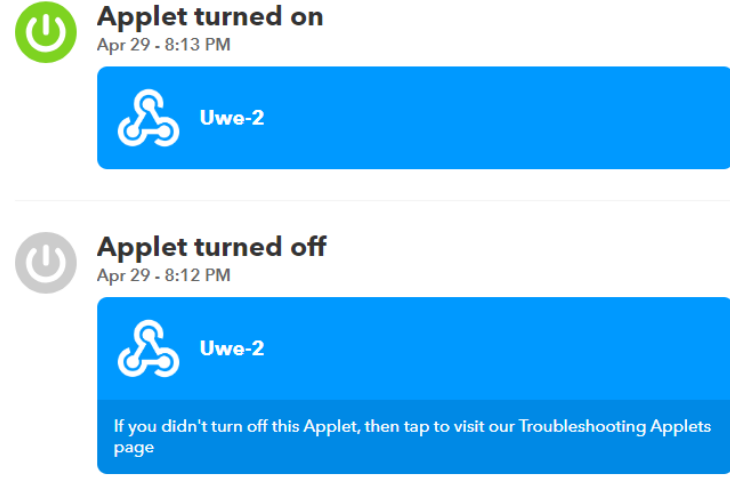
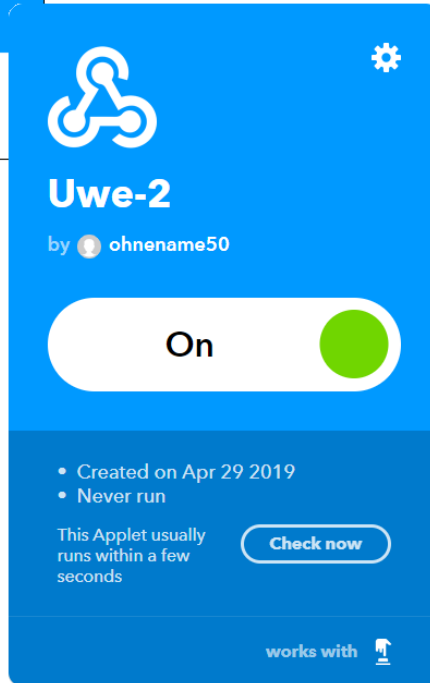
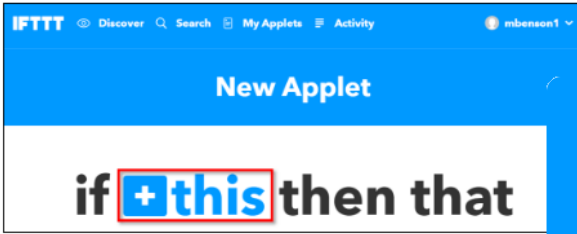
```
In [*]: 1 # Code Cell 3.
2 #Test the circuit by changing the state of the GPIO pins
3 #This code will cycle twice.
4 for i in range(2):
5     print("Green ON Red Off")
6     GPIO.output(GreenLEDPin, True) # True = set 3.3V on the pin
7     GPIO.output(RedLEDPin, False) # False = set 0V on the pin
8     time.sleep(1) #wait one second.
9     print("Green OFF Red ON")
10    #-----
11    GPIO.output(GreenLEDPin,False) # Add values: turn the LED off
12    GPIO.output(RedLEDPin,True) # Add values: turn the LED on
13    #-----
14    time.sleep(1) #wait one second.
15 GPIO.output(RedLEDPin, False) #turn off red LED after loop

Green ON Red Off
Green OFF Red ON
Green ON Red Off
```

Our workshop Lab with the Classroom topology

Open Jupyter-Notebook Lab-2.2.1.4

Part 2: Create an IFTTT app. Step 1: Register a free user account at IFTTT



Our workshop Lab with the Classroom topology

Open Jupyter-Notebook Lab-2.2.1.4



```
In [67]: 1 #Code cell 5
2
3 #-----
4 #myThing = "your_thing_name" #Add value: add your dweet thing name
5 myThing = "Uwe_Sec_2" #Add value: add your dweet thing name
6 #-----
7
8 old_dweet = dweepy.dweet_for(myThing,{"dweet": "1"}) #this function sends data to dweet.io
9
10 old_created = old_dweet['created'] #get the time stamp of the first dweet
11
12 print(old_created)
13
2019-04-29T18:29:22.603Z
```

```
In [*]: 1 #Code cell 6.
2 counter = 0
3 lit1 = "g"
4 while True:
5     new_dweet = dweepy.get_latest_dweet_for(myThing) #get latest dweet
6     #print("New :", new_dweet)
7     new_created = new_dweet[0]["created"] #put the created value of the lastest dweet into a variable
8     #print ("Old :", old_created)
9     if new_created != old_created: #check to see if the the old dweet is different from the new dweet
10        counter += 1
11        print(str(counter) + " New dweet detected!",end='\n')
12        old_created = new_created
13        if lit1 == "g":
14            print("Activate red LED")
15            GPIO.output(GreenLEDPin, False) # False = set 0V on the pin
16            GPIO.output(RedLEDPin, True) # True = set 3.3V on the pin
17            lit1 = "r"
18        elif lit1 == "r":
19            print("Activate green LED")
20            GPIO.output(GreenLEDPin, True)
21            GPIO.output(RedLEDPin, False)
22            lit1 = "g"
23        time.sleep(1)
24
25 1 New dweet detected!
26 Activate red LED
27 2 New dweet detected!
28 Activate green LED
```


Our workshop Lab with the Classroom topology

Open Jupyter-Notebook Lab-2.2.1.4



*Ethernet 10

File Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe

tcp.stream eq 1308

No.	Time	Source	Destination	Protocol	Length	Info
6828	471.862611	UweSec.local	dweet.io	TCP	74	39444 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=656442341 TSecr=0 WS=128
6833	472.078603	dweet.io	UweSec.local	TCP	74	443 → 39444 [SYN, ACK] Seq=0 Ack=1 Win=26847 Len=0 MSS=1452 SACK_PERM=1 TSval=864236886 TSecr=...
6834	472.078603	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=656442557 TSecr=864236886
6835	472.173426	UweSec.local	dweet.io	TLSv1.2	289	Client Hello
6838	472.388722	dweet.io	UweSec.local	TCP	66	443 → 39444 [ACK] Seq=1 Ack=224 Win=28160 Len=0 TSval=864236963 TSecr=656442652
6839	472.388723	UweSec.local	dweet.io	TLSv1.2	1506	Server Hello
6840	472.388723	dweet.io	UweSec.local	TCP	1506	443 → 39444 [ACK] Seq=1441 Ack=224 Win=28160 Len=1440 TSval=864236963 TSecr=656442652 [TCP se...
6841	472.388724	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=224 Ack=1441 Win=32128 Len=0 TSval=656442867 TSecr=864236963
6842	472.388875	UweSec.local	dweet.io	TCP	1282	443 → 39444 [PSH, ACK] Seq=2881 Ack=224 Win=28160 Len=1216 TSval=864236963 TSecr=656442652 [T...
6843	472.388877	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=224 Ack=2881 Win=35072 Len=0 TSval=656442868 TSecr=864236963
6844	472.389038	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=224 Ack=4097 Win=37888 Len=0 TSval=656442868 TSecr=864236963
6845	472.390445	dweet.io	UweSec.local	TLSv1.2	1241	Certificate, Server Key Exchange, Server Hello Done
6846	472.390582	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=224 Ack=5272 Win=40832 Len=0 TSval=656442869 TSecr=864236964
6847	472.397085	UweSec.local	dweet.io	TLSv1.2	192	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
6849	472.612706	dweet.io	UweSec.local	TLSv1.2	117	Change Cipher Spec, Encrypted Handshake Message
6850	472.614059	UweSec.local	dweet.io	TLSv1.2	264	Application Data
6853	472.869916	dweet.io	UweSec.local	TCP	66	443 → 39444 [ACK] Seq=5323 Ack=548 Win=29184 Len=0 TSval=864237084 TSecr=656443093
6856	473.556325	dweet.io	UweSec.local	TLSv1.2	224	[TCP Previous segment not captured], Application Data
6857	473.556325	UweSec.local	dweet.io	TCP	78	[TCP Window Update] 39444 → 443 [ACK] Seq=548 Ack=5323 Win=43648 Len=0 TSval=656444035 TSecr=...
6858	473.556326	dweet.io	UweSec.local	TLSv1.2	100	Application Data
6859	473.556471	UweSec.local	dweet.io	TCP	78	[TCP Window Update] 39444 → 443 [ACK] Seq=548 Ack=5323 Win=46592 Len=0 TSval=656444035 TSecr=...
6862	473.828499	dweet.io	UweSec.local	TCP	306	[TCP Retransmission] 443 → 39444 [PSH, ACK] Seq=5323 Ack=548 Win=29184 Len=240 TSval=86423732...
6863	473.828499	UweSec.local	dweet.io	TCP	66	39444 → 443 [ACK] Seq=548 Ack=5755 Win=49408 Len=0 TSval=656444307 TSecr=864237324

Length: 4821
Certificates Length: 4818
> Certificates (4818 bytes)

- Transport Layer Security
 - TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 333
 - Handshake Protocol: Server Key Exchange
 - Handshake Type: Server Key Exchange (12)
 - Length: 329
 - EC Diffie-Hellman Server Params
 - TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 4
 - Handshake Protocol: Server Hello Done
 - Handshake Type: Server Hello Done (14)
 - Length: 0

Frame (1241 bytes) Reassembled TCP (4830 bytes)

Internet Protocol Version 4 (IP), 20 Bytes

uwe.starke@hs-wismar.de IoT - Security Acadday 2019

Paket: 6871 | Angezeigt: 20 (0.4%) | Verworfen: 0 (0.0%) | Profil: Default

Our workshop Lab with the Classroom topology



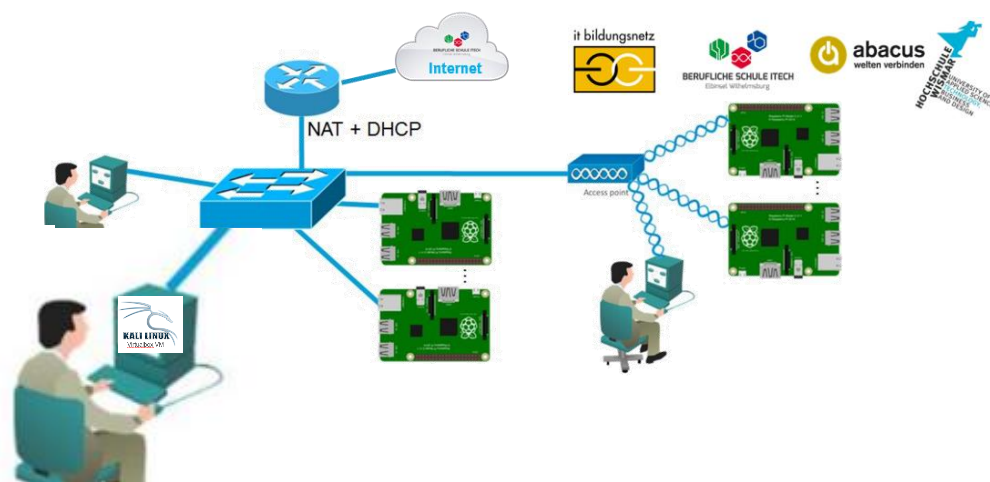
Problems :

- **Cloud-communication between IFTTT and <https://dweet.io/dweet/for/things>**
 - **do not use App, refresh the Webpage <https://dweet.io/dweet/for/things>**
- **Raspberry to dweet.io behind NAT / Firewall**
 - **ask the IT**

Lab - Port Scanning an IoT Device

Topology Classroom-Demo

Kali-Linux VM is preconfigured



```

root@kali: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
root@kali:~# nmap --top-ports 10 192.168.7.0/24

```

```

root@kali: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
3389/tcp closed ms-wbt-server

Nmap done: 256 IP addresses (11 hosts up) scanned in 14.42 seconds
root@kali:~#

```

```

root@kali: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
Nmap scan report for UweSec.fritz.box (192.168.7.98)
Host is up (0.00063s latency).

PORT      STATE SERVICE
21/tcp    closed ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    closed smtp
80/tcp    open  http
110/tcp   closed pop3
139/tcp   closed netbios-ssn
443/tcp   closed https
445/tcp   closed microsoft-ds
3389/tcp  closed ms-wbt-server
MAC Address: B8:27:EB:83:92:44 (Raspberry Pi Foundation)

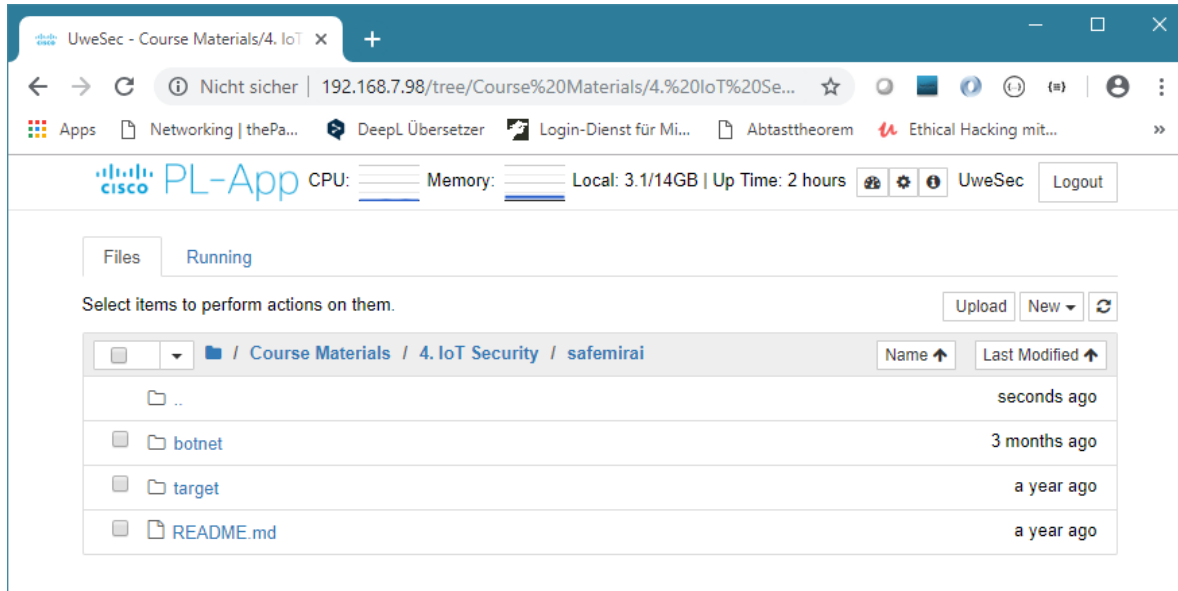
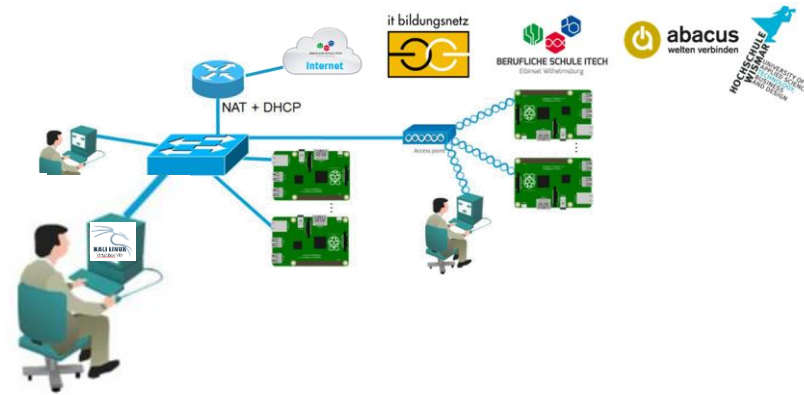
Time      Source      Destination      Protocol  Length  Info
Nmap scan report for 192.168.7.101
221 6 0 251      MDNS      77 Standard qu

```

Lab - Port Scanning an IoT Device

Topology Classroom







What possibilities?



UweSec - Course Materials/4. IoT x

Nicht sicher | 192.168.7.98/tree/Course%20Materials/4.%20IoT%20Se...

Apps Networking | thePa... DeepL Übersetzer Login-Dienst für Mi... Abtasttheorem Ethical Hacking mit...

 PL-App CPU:  Memory:  Local: 3.1/14GB | Up Time: 2 hours    UweSec Logout

Files Running

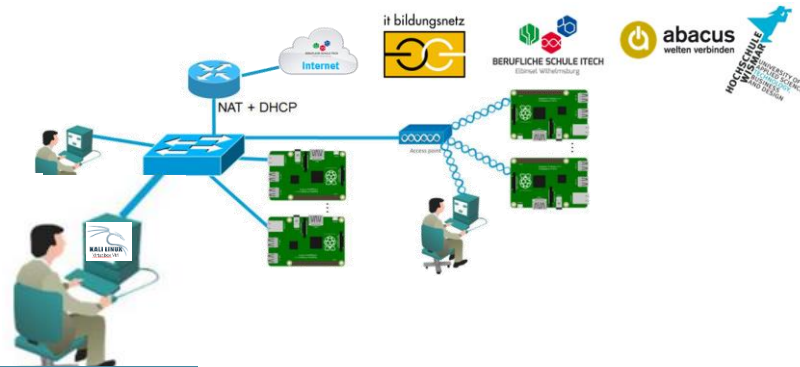
Select items to perform actions on them. Upload New ↕ ↻

<input type="checkbox"/>		Name ↑	Last Modified ↑
<input type="checkbox"/>	..		seconds ago
<input type="checkbox"/>	botnet		3 months ago
<input type="checkbox"/>	target		a year ago
<input type="checkbox"/>	README.md		a year ago

Lab - Port Scanning an IoT Device

Topology Classroom

What possibilities?



```
UweSec - Course Materials/4. IoT x README.md x +
Nicht sicher | 192.168.7.98/view/Course%20Materials/4.%20IoT%20Se...
Apps Networking | thePa... DeepL Übersetzer Login-Dienst für Mi... Abtasttheorem Ethical Hacking mit...

# safemirai

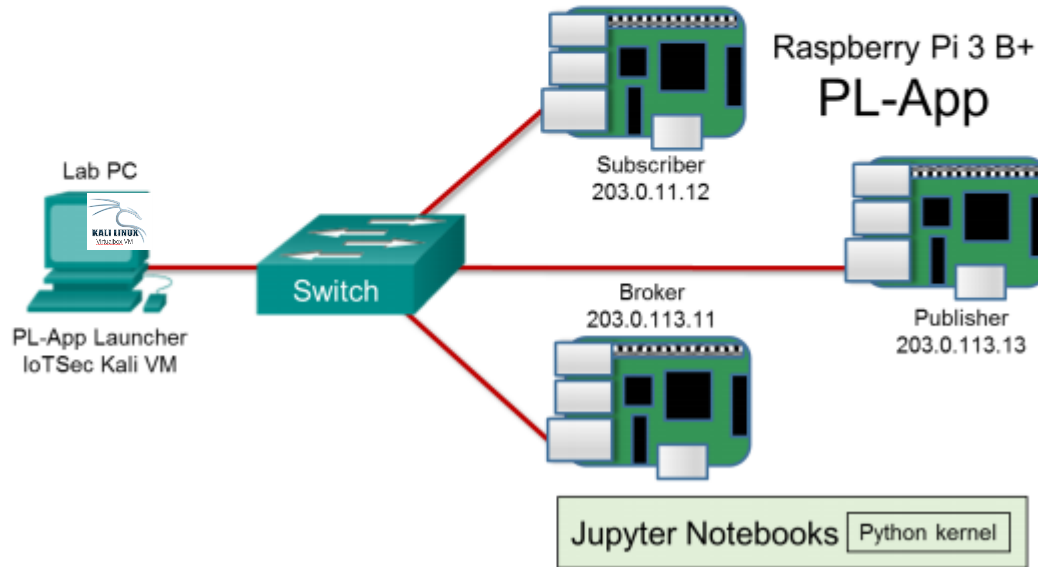
## Equipment
* 7 pi's

## Setup
1. Use PL-App Launcher to burn images on the SD cards, and name them instructor, target, pi1, pi2, ..., pi5.
1. Connect all pi's to the same subnet using Ethernet and DHCP assigned IPs.
1. On pi1 to pi5, run the botnet/node_setup.sh script.
1. Change the password of admin to 'admin': passwd admin
1. On target, start a web server in the target folder: python -m http.server

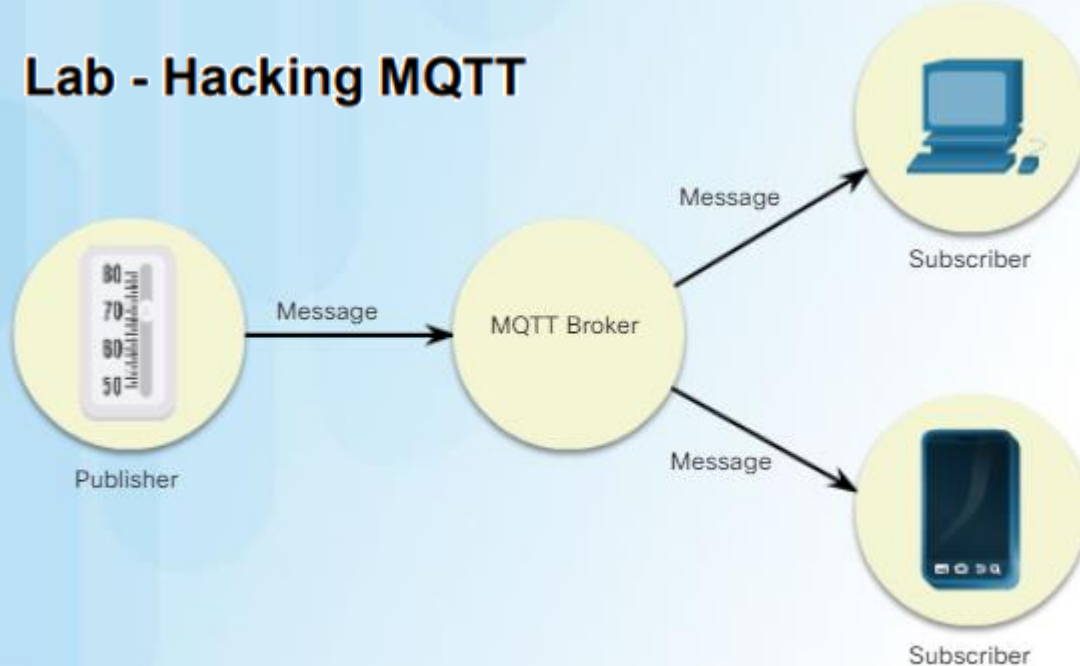
## Run
1. On instructor, run: python master.py 2323
1. Go to http://\<instructor-ip>:1888/master.html.
1. Click on Scan, and look at python console scanning for IPs in the same subnet.
1. Wait til pi1 to pi5 are infected and added to the list.
1. Go to http://\<target-ip>:8000 to see that the target is reachable.
1. Enter the IP and port of the web server on the target, 1000000 for count, and click Attack.
1. Go to http://\<target-ip>:8000 again and see that it times out or takes a while.
```

Lab - Hacking MQTT

Topology



Lab - Hacking MQTT



MQTT uses a type client-server model called **publish-subscribe**.

Clients connect to the server called a broker.

Client **either publishes a topic or subscribes** to a topic.

A topic is any specific type of message, like humidity, temperature, or light.

Lab - Hacking MQTT

Steps in the Lab:

- Run a local broker server.
- Subscribe to a topic and publish MQTT messages.
- Sniffing Data using Kali.
- MITM attack using Kali on TCP.
- View the capture using Wireshark.
- Publish rogue data.
- Adding Username/Password Authentication.
- Replay attack and publish rogue data.
- Adding TLS Protection.
- MITM attack using Kali.

