# ****Internet of Things Security****

**Lecture 3: Review of Attacks in IoT**  
**Instructor:** Mehmoona Jabeen  
**Email:** Mehmoona.jabeen@au.ed.pk  
**Department of Cyber Security, Air University**  
**Course:** CYS, Spring 2025

## ****Lecture Outline****

* Taxonomy of Attacks
* Physical Attacks
* Software Attacks
* Network Attacks
* Attacks in WiFi
* ESP32 for WiFi Pentesting
* IoT Security Foundation

## ****IoT Security Goals****

* The traditional and common security goals include **Confidentiality**, **Integrity**, and **Availability (CIA)**.
* Apart from the CIA triad, other requirements have become important:
  + **Privacy**
  + **Lightweight solutions**
  + **Authenticity**
  + **Standardized policies**

### ****Breakdown:****

* **Authenticity:** user, device, context
* **Confidentiality:** storing data and keys securely
* **Integrity:** data, logs, and firmware integrity
* **Availability:** fault tolerance and scalability
* **Privacy:** non-link-ability, location, data and device

## ****Taxonomy of Attacks****

|  |  |  |
| --- | --- | --- |
| **Physical Attacks** | **Software Attacks** | **Network Attacks** |
| Node Tampering | Malware | Traffic Analysis Attack |
| RF Jamming | Loggers Attack | RFID Spoofing |
| Malicious Node Injection | Data Tempering | Routing Information Attack |
| Sensor/Actuator Attack |  | Selective Forwarding |
| Physical Damage (PDoS) |  | Sinkhole Attack |
| Sleep Denial Attack |  | Sybil Attack |
| Side Channel Attacks |  | Man in the Middle Attack |
| Energy Harvesting Attack |  | Replay Attack |
| Power Analysis Attack |  | DDoS |
| Timing Attack |  |  |
| EM Side Channel Attack |  |  |
| Reverse Engineering |  |  |

## ****Physical Attacks****

### ****Node Tempering****

* Attacker physically alters the compromised node to obtain login credentials, encryption keys, and sensitive info.
* Can occur in:
  + Development / manufacturing / packaging phases
  + Pre-deployment phase
  + Deployment phase

### ****Smart Meter Tampering****

(Title only - no content in original slides)

### ****RF Jamming****

* Instead of sending RF signals, attacker transmits **noise signals** to launch **DoS attacks** on RFID tags.
* Known as **RF interfacing/jamming**.
* Primary goal: **hinder communication**.

### ****Node Injection****

* Malicious node is dropped among legal nodes (fake node injection).
* Attacker gains control of data flow.
* Many physical devices are vulnerable.

### ****Sensing/Actuating Attack****

* False/spoofed commands to actuators disrupt operations.
* Example:
  + Alter sensor reading
  + Cause pump to over/under deliver water

### ****Permanent Denial of Service (PDoS)****

* Physically damaging the node to stop requests.
* Known as **phlashing**: bricks a device or corrupts firmware.
* Device must be repaired or replaced.
* Example: Hacker "Janit0r" bricked 2 million IoT devices in 2017.

#### ****Cont. PDoS****

1. **Compromising a Device**
   * Bricker Bot uses **Telnet brute force** with common credentials.
2. **Corrupting a Device**
   * Executes Linux commands to:
     + Corrupt storage
     + Disrupt connectivity
     + Wipe files

### ****Reverse Engineering****

* Attacker disassembles a device to discover and exploit vulnerabilities.

### ****Side Channel Attacks****

* Target cryptosystems by analyzing:
  + **Power**
  + **Timing/delay**
  + **Electromagnetic emissions**

#### Types:

* **Power Analysis Attack**
* **Timing Attack**
* **Electromagnetic Side-channel Attack**
* **Fault Attack**

### ****Energy Harvesting/Depleting Attack****

* IoT devices use energy from ambient sources.
* Attacker:
  + Blocks energy source
  + Engages device in heavy tasks to drain battery

## ****Software Attacks****

### ****Malware****

* IoT **botnets**: infected devices used for **DDoS attacks**.

### ****Data Tampering****

* Deliberately altering data in transit or at rest.
* Dangers:
  + **Lack of detection**
  + **Small tampering = big consequences**

### ****Traffic Analysis Attack****

* Attacker gains network info without direct access.
* Risk of **data leakage**.

### ****MITM (Man-in-the-Middle) Attack****

* Eavesdrops/monitors communication between IoT devices.
* Violates data privacy.
* Can take control of smart actuators (e.g., industrial robot disruption).

### ****Firmware Attacks****

* **Reverse Engineering**: Extract/analyze firmware for info
* **Firmware Modification**: Inject malicious code
* **Obtaining Authorization**: Gain unauthorized access
* **Installing Unauthorized Firmware**: Malicious/stolen firmware
* **Unauthorized Devices**: Fake device receives authentic firmware

**Basic Attacks on WiFi**

* **Deauthentication attack**
* **WPA handshake brute-force attack**
* **PMKID capture and brute-force attack**
* **WPS PIN attack**
* **KRACK attack**

**Authentication in WiFi**

**Deauthentication Attack**

* Deauthentication attack targets wireless networks by sending deauthentication frames to devices on the network, causing them to disconnect from the network.
* The deauthentication frames are sent using a spoofed MAC address, making it difficult for the network to identify the attacker.
* The purpose is to disrupt normal WiFi network operations, causing devices to repeatedly disconnect and reconnect, resulting in a Denial of Service (DoS) condition.
* Can cause network instability or crashes.
* Tools like **Aircrack-ng** can be used to carry out such attacks.
* Particularly effective against networks with **weak or outdated security protocols**, such as **WEP** or **WPA**.

**WPA Handshake Brute-Force Attack**

* When a client connects to a WPA/WPA2 network, a **four-way handshake** is performed using the **pre-shared key (PSK)** or **pairwise master key (PMK)**.
* This handshake can be **captured using a wireless packet capture tool**.
* After capturing, attackers **guess the PSK** using brute-force—trying different character combinations.
* Specialized software performs brute-force attempts until the correct key is found.

**PMKID Capture and Brute-Force Attack**

* A **relatively new attack** targeting networks using **WPA3-PSK encryption**.
* Involves capturing the **Pairwise Master Key Identifier (PMKID)** used during the client’s authentication process.
* PMKID is generated using the network's **SSID** and **PSK**.
* Attackers capture the PMKID using a packet capture tool and **brute-force the PSK** until it's found.

**WPS PIN Attack**

* Targets networks using the **WPS feature** for device connection.
* Instead of entering the PSK, users input an **eight-digit PIN**.
* Attack involves **brute-forcing the PIN** using tools like **Reaver** or **Bully**, which send PIN guesses to the access point until the correct one is found.

**KRACK Attack (Key Reinstallation Attack)**

* Targets **WPA/WPA2** networks.
* Exploits a protocol vulnerability to **intercept and decrypt network traffic**.
* Intercepts the **four-way handshake** between the client and access point.
* Forces the client to **reuse an already-used encryption key**, allowing traffic decryption.

**ESP32**

**Introduction**

* **ESP32** is a powerful **Wi-Fi and Bluetooth enabled** microcontroller chip by **Espressif Systems**.
* Successor to the **ESP8266** with:
  + More processing power
  + More memory
  + Added features like Bluetooth and **dual-core processing**
* Features:
  + **Dual-core processor** up to 240 MHz
  + Built-in **Wi-Fi** and **Bluetooth**
  + Multiple protocols and GPIO support

**ESP32 Basic Architecture**

* **Dual-core processor**: Two Tensilica LX6 cores up to 240 MHz
* **Memory**:
  + Up to 520KB SRAM
  + Up to 4MB Flash
  + 8KB RTC memory
* **Connectivity**:
  + Wi-Fi, Bluetooth Classic, BLE
  + Wi-Fi Direct, WPS
* **Peripheral Interfaces**:
  + UART, SPI, I2C, PWM, ADC
  + SD card, Ethernet, CAN bus
* **Security**:
  + Secure boot
  + Flash encryption
  + Hardware-accelerated encryption
* **Power Management**:
  + Multiple sleep modes for low power usage

**ESP32 for WiFi Attacks**

* **Low power**, **lightweight**, **battery-operated** device ideal for Wi-Fi attacks.
* **Universal Wi-Fi penetration tool** developed:
  + Uses **ESP-IDF public API**
  + Bypasses closed-source libraries that block forged frames

**Available WiFi Attacks Using ESP**

1. **ESP8266 Deauther** (by Stefan Kremser aka “spacehuhn”)
   * Performs **deauthentication attacks** via ESP8266
2. **ESP32 Deauther Tool**
   * Open-source project by various developers (e.g. GANESH ICMC/USP)
   * Supports:
     + Deauthentication attacks
     + Beacon frame injection
     + Probe request attacks
   * Portable and easy-to-use

**ESP32 Wi-Fi Penetration Tool Components**

* **Wi-Fi Controller**
  + Manages Wi-Fi interfaces and configurations
  + Enables promiscuous mode
* **Frame Analyzer**
  + Parses captured frames (e.g. PMKIDs)
  + Filters by BSSID and helps analysis
* **WSL Bypasser**
  + Overrides blocking functions in Wi-Fi Stack Libraries
* **Webserver**
  + UI for tool control and attack configuration
  + Built on **ESP-IDF HTTP server**
* **PCAP & HCCAPX Serializers**
  + Convert frames to formats for:
    - **Wireshark** (PCAP)
    - **Hashcat** (HCCAPX)
* **Main Component**
  + Manages attack types and variations
  + Handles timeouts, configs, and shared operations

**IoT Attacks Mitigation & Research**

**Research Challenges**

* Need for **lightweight and robust trust management**
* Ensure **physical security**, **risk management**, and **intrusion detection** at all IoT layers
* Develop a **standardized security framework**
* Improve **security protocols** as per application
* Consider **K-anonymity** for identity/location privacy
* Use **ML/DL-based real-time analysis** at IoT nodes
* **Dataset limitations** lead to ML inaccuracies
* Design **lightweight cryptographic algorithms**
* Combine encryption with **autonomic approaches**

**IoT Security Foundation**

* Established to address **IoT security challenges**
* Proposes a **hub-based architecture** to:
  + Reduce system complexity
  + Demonstrate secure system examples
  + Manage system hygiene and resilience
  + Provide centralized management for enterprises

**Hub-Based Security Architecture**

**Hub Functions**

**Network Management & Security**

* Local IoT Network (separation for security)
* Staging & Live system separation
* Gateways and firewalls

**Connecting Devices Securely**

* **Authentication & Authorization**
* **Secure Boot**
* **Roots of Trust**

**Lifecycle Management**

* **Monitoring, Auditing, Discovery**
* **Update and Patch Management**
* **Device Identity and Authorization**
* **End-of-life Device Management**