Security in IoT Ecosystem

Module 2

IoT Pentesting methodology

Prof.: Joaquín Recas

Attify

- IoT Security and Exploitation Education
- Attify helps businesses secure their IoT devices, mobile applications and digital assets.
  - Offering security assessment, **penetration testing** and **training** to organizations all over the world
  - *(Attify is the leading global security provider)*

https://www.attify.com/

Reasons for IoT Security Vulnerabilities

- Lack of Security Awareness Among Developers:
  - Developers are often less knowledgeable about the possible security vulnerabilities in IoT devices
- Lack of a Macro Perspective:
  - It is easy for developers to forget about the fact that it is the interconnection of devices and various technologies that could lead to security issues.
- Supply-Chain-Based Security Issues:
  - It is common to have different components of devices being manufactured by different vendors, getting assembled by another vendor, and finally distributed by yet another one
- Usage of Insecure Frameworks and Third-Party Libraries:
  - Normally developers use existing libraries and packages and they introduce potentially vulnerable code samples into the product.

IoT Pentesting methodology

- **Steps** and techniques to be followed in order to effectively **pentest** a device:
  1. Attack Surface Mapping
  2. Firmware Security Analysis
  3. Attacking Web, Mobile and cloud assets
  4. Hardware Exploitation
  5. Radio based Exploitation

Attack Surface Mappin

- Finding as much **information** as possible about the **device.**
- The entire architecture can be divided into three categories:
  1. Embedded device.
  2. Firmware, software, and applications.
  3. Radio communications.

Embedded device

i. Serial ports exposed.
ii. Insecure authentication mechanism used in the serial ports.
iii. Ability to dump the firmware over JTAG or via Flash chips.
iv. External media-based attacks.
v. Power analysis and side channel-based attacks.

Firmware, Software, and Applications

i. Mobile application:
   a. Reverse engineering the mobile app.
   b. Runtime manipulation attacks.
ii. Web-based dashboard
   a. Client-side injection.
iii. Insecure network interfaces
   a. Outdated version with known vulnerabilities
iv. Firmware
   a. Ability to modify firmware.
   i. Hard-coded sensitive values in the firmware

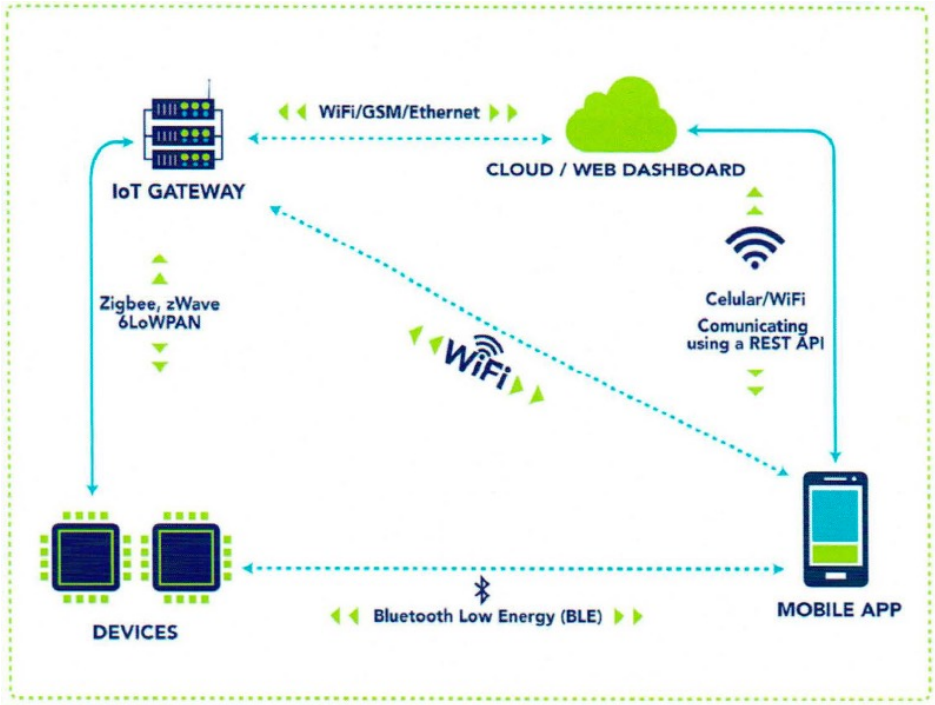
Radio Communications

i. Categories:
   a. Software Defined Radio (SDR).
   b. ZigBee exploitation.
   c. BLE exploitation

ii. Sets of vulnerabilities
   a. Man-in-the-middle & Replay-based attacks.
   b. Insecure Cyclic Redundancy Check (CRC) verification.
   c. Jamming-based & Denial of service (DoS) attacks.
   d. Lack of encryption.
   e. Ability to extract sensitive information from radio packets.
   f. Live radio communication interception and modification.

Attack Surface Mappin

- **High-level** architectural **diagram** of the entire solution



- Understand the different **components** involved, how they **communicate** and identify possible security issues

Ninja Recon Technique

- Basic information about the device:
  - In an **official pentest** engagement, the details would be **provided by the client**.
  - If it's an independent security research, you could look for these information and fill it up yourself

Ninja Recon Technique - Step 1	
What is the target product	
URLs (if available)	
Public domain info	
FCC ID	
Technologies Used	
Components in the product	
Mobile Application URLs	
Web Dashboard URLs	
Thick Clients	

Ninja Recon Technique

- Identifying Assets section:
  - **List down all the components** from our *Attack Surface Map*
  - Identify **user roles** which the product supports

Ninja Recon Technique - Step 2	
ASSET	**DESCRIPTION**
IP Camera	The IP Camera of the Smart Home solution acts as a constant security cam monitoring motions and clicking picture whenever an event occurs. It also notifies the user whenever a suspicious event has occured based on the parameters set by the user. The data of the IP Camera is stored for 24 hours on the SDCard, as well as sent to the application servers for the user to view.
LED bulbs	LED Bulbs of the Smart Home solution are controlled by the mobile application via Bluetooth Low Energy and transmit data over ZigBee to the central hub. The LED bulbs also has an automated color setting functionality which can be set by the user in the mobile application.
Mobile applications	The Smart Home solution comes with applications for both Android and iOS platform. It uses BLE to communicate with the local devices and sends data via a REST API to the remote endpoint. The mobile application is built using React Native and includes a number of 3rd party SDKs.
IoT hub	The IoT Smart Home Hub connects to the router/gateway to interface with the external world. It runs on an ARM SoC and provides support for UART, JTAG, SPI and additional communication protocols. It also uses MQTT for messaging over the port 1883. The IoT Hub also holds the configuration information of the entire smart home system and constantly logs information to be stored both locally and sent to the application server to process and provide intelligent insights.
Router	The router takes care of all the network communication happening between the devices and the remote endpoints. It can also selectively block certain devices or provide a pass-through mode for unrestricted access.
USER ROLES	**PERMISSION LEVELS**
Authenticated User	Able to change the storage location of the camera captures, sensitivity of motion sensors and upload frequency to the web server
Unauthenticated User(Guest)	-
Device Admin	Change the on/off timings of the IP camera, add additional users, modify the user to become an admin, update firmware

Ninja Recon Technique

- Prepare the **test cases** for each of the various **components** or section of components present in the product

Ninja Recon Technique - FINAL STEP				
SECTION	COMPONENT / TEST CASE	POSSIBLE VULNERABILITY	HOW TO TEST	IMPACT
MOBILE APPS	RE the Android and iOS application			
	Hardcoded and Sensitive information			
	SSL Pinning			
	Intercepting the communications API			
	Local Data Storage			
	Insecure authentication and authorization checks			
	Business and logic flaws			
	Side channel data leakage			
	Runtime manipulation attacks			
	Insecure network communication			
	Outdated 3rd party libraries and SDKs			
NETWORK	Secure channel used for communication			
	MITM based vulnerabilities			
	Fuzzing network communication protocols			
	Insecure services running on the target device			
	Ports open without authentication required			
RADIO	Radio communication over Insecure Channel			
	Encryption in radio packets			
	Key Storage for Encryption			
	Spoofing and Replay based attacks			

Firmware Security Analysis

- Firmware holds *a ton* of information about the device including:
  - Information on **how the device functions**
  - **Sensitive configurations**
  - API **keys** and more
- Some of the techniques of obtaining a firmware binary of an IoT device includes:
  - Intercepting the *OTA* update and dumping the firmware binary from the network capture
  - Dumping it from the device
  - Reversing the mobile application or trick client to find the remote location of the firmware
  - Downloading the firmware binary from the vendor's website or public forums

Embedded Device Exploitation

- Analysing the **hardware** device for security vulnerabilities will allow us to obtain sensitive information about the target IoT system as well as, in some cases gain, full access to the IoT device.
- Some of the ways in which we can attack the target are:
  1. **UART exploitation to gain root shell**
  2. **Reading and writing content from the SPI chip**
  3. JTAG debugging and exploitation
  4. Runtime manipulation via JTAG

Radio Exploitation

- Radio in IoT is the component which facilitates communication between any two components.
- Two of the most common radio communication protocols in IoT devices are:
  - Bluetooth Low Energy
  - WiFi
  - Zig Bee