```yaml
aliases:
date created: Wednesday, July 6th 2022, 2:16:32 pm
date modified: Friday, July 8th 2022, 4:07:17 pm
title: IoT Pentesting
```

tags: cybersecurity

IoT Pentesting

Summary

Recon

1. all physical input, output, debug ports and components
2. non-physical connection - bluetooth, wifi, zigbee, web/mobile app, network services

- how:
  - datasheet, FCC database
  - external and internal inspection

Attack Surface & Methods

1. hardware
   a. I/O ports, debug ports, storage medium
   b. debug info, modifying memory, dumping firmware/memory data, accessing shell, privilege escalation (including physically shorting pins)
2. firmware
   a. retrieved by
      i. direct download, dumping, RE to get download URL
      ii. if encrypted → look for firmware signing key (sent over HTTP, etc)
   b. hard-code keys, sensitive info, database systems
   c. vulnerability via fuzzing and RE

    d. see OWASP Firmware Security Testing Methodology
3. web/mobile app
4. network services
    a. ssh, ftp, http
    b. message protocols:
        i. MQTT
            1. authentication optional and in clear text
            2. check if there is sensitive information, subscribe/send to all topics, clientID misconfig
            3. fuzzing
        ii. CoAP
            1. check auth mechanism, DTLS may not be used
            2. fuzzing URIs
                1. Enumerate resources by GET, check PUT/POST/DELETE
5. wireless
    a. identify protocol, frequency/sample rate/modulation, channel and address
    b. sniffing and decode
    c. (bypass auth)
    d. replay-based attack/ modifying packet data

Goals

1. credentials, hard-coded keys and sensitive info
2. sending unauthenticated request
3. decrypting network traffic to eavesdrop
4. sabotage/ undermine service availability
5. root access to shell

Methodologies

Attack Surface Mapping

1. Look at:

a. embedded device
b. firmware, software and applications
c. radio communications

Hardware

Inspection

- External inspection
- Internal inspection
  - Datasheet, FCC database
  - components used in the device, CPU architecture type, communication protocols used, mobile application details, firmware upgrade process, input/output/debug ports, external media support on devices, etc

Protocols

- UART
  - Identity pins
  - Connect to Attify Badge (our UART reader device)
  - Identify Baud rate
  - Interact
- I2C, SPI for reading flash/ other storage medium
- JTAG for debugging and dumping contents in flash

The Interface	Purpose
Serial Interface	•Debug outputs •Shells
i2c and SPI Interfaces	•EEPROM data sniffing and injection •Memory Dumping •Debug output •Device management
JTAG Interface	•Firmware Dumping •Firmware Upgrades •Testing and Debugging the device

Methods

- directly connect to debug ports may provide shell
- dumping data
- Shorting NAND pin to gain u-boot root shell, bypassing system shell

Firmware

- See https://scriptingxss.gitbook.io/firmware-security-testing-methodology/
- be aware of file system and compression method
- getting binary
  - download, dumping, sniffing
    - Check encrypted?
    - If yes, check if XOR encrypted with hexdump
    - if no, extract file system type and contents
- analyzing file system contents and look for:
  - Hard-coded credentials.
  - Backdoor access.
  - Sensitive URLs.
  - Access tokens.
  - API and encryption keys.
  - Encryption algorithms.
  - Local pathnames.
  - Environment details.
  - Authentication and authorization mechanisms.
- binary exploitation
  - firmware diffing with kdiff to find vuln
  - code execution, backdooring
- auto scanner: bytesweep, fuzzing with AFL++

Mobile, Web and Network Services

- Methods:
  1. RE with MobSF

    2. nmap to look at services open
- Tools:
    1. jadx/ APKTool
    2. radare2/IDA Pro
- Look for:
    1. Hard-coded credentials or sensitive URL
    2. AES Key to decrypt traffic

Wireless

Radio

1. identify frequency, modulation, data rate, sample rate
2. decode, replay with different values in the command

Zigbee and BLE

1. identify channel used among 16 channels
2. intercept, replay, modify packets
3. Tool: KillerBee (not well supported), ZigDiddity, BTLEJuice (framework)

Study packet and observe pattern

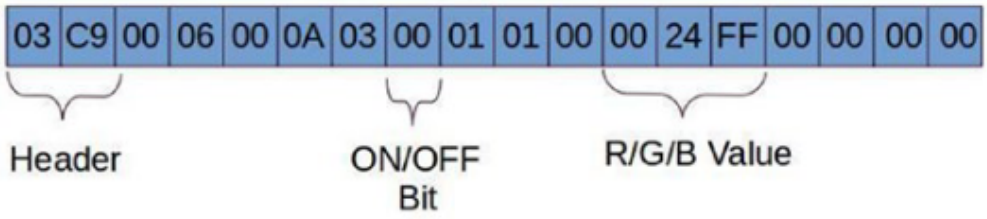

Figure 10-47. *BLE packet data structure for a light bulb showing the RGB and ON/OFF values*

Tools List with Price in HKD

1. Multimeter
2. Good screwdriver set
3. AttifyOS

4. BusPirate v3.6/ Attify Badge for UART, SPI, I2C and JTAG $168/ $345
    a. https://item.taobao.com/item.htm?spm=a230r.1.14.11.1e4d5a6fCZxgdE&id=529715134778&ns=1&abbucket=11#detail
    b. BusPirate is cheaper but slower in JTAG debugging
5. Wireless
    a. Radio: RTL-SDR exclude antenna(sniffing only) $158, LimeSDR $384
        i. https://item.taobao.com/item.htm?spm=a230r.1.14.38.72d13e78UeSmTy&id=622074779967&ns=1&abbucket=11#detail
        ii. https://www.crowdsupply.com/lime-micro/limesdr-mini#products
    b. Zigbee:
        i. Xbee, Xbee shield for Nano arduino $233
            1. https://item.taobao.com/item.htm?spm=a230r.1.14.8.55c5526bhTj7cR&id=586663892813&ns=1&abbucket=11#detail
            2. https://item.taobao.com/item.htm?id=16213111735&spm=2013.1.20141003.6.54026b7csLGcWQ&main_itemid=38885962640&go_item_id=16213111735&scm=1007.10011.99062.&pvid=23191bae-07b6-4dbe-9f94-b3205ba5ea43
        ii. Attack frameworks:
            1. Zigdiggity
                1. RaspBee module for Raspberry $318
            2. Killerbee (not well-supported)
                1. APIMote ~$1700
                2. Zigbee Packet Sniffer CC2531 but only support sniffing
        iii. Zigbee2MQTT HKD$248
            1. https://shop.electrolama.com/collections/usb-rf-sticks/products/zzh-multiprotocol-rf-stick?variant=40387937468577

    c. BLE:
        i. BLE dongle nRF52832 $70 - notice BLE 4.2/5.0
            1. https://item.taobao.com/item.htm?spm=a230r.1.14.3.5c2f20d0Cr2fLM&id=652694121526&ns=1&abbucket=11#detail

Common Vulnerabilities

OWASP Top 10 2018

1. Weak, Guessable, publicly available or Hardcoded Passwords
2. Insecure Network Services - ftp open
3. Insecure Ecosystem Interfaces - web, mobile
4. Lack of Secure Update Mechanism - malicious firmware
5. Use of Insecure or Outdated Components - heartbleed, meltdown...
6. Insufficient Privacy Protection
7. Insecure Data Transfer and Storage - no SSL...
8. Lack of Device Management
9. Insecure Default Settings
10. Lack of Physical Hardening

References:

1. The IoT Hacker's Handbook: www.ime.cas.cn/icac/learning/learning_3/201907/P020190724586712846107.pdf
2. Firmware Pentesting Methodologies: https://scriptingxss.gitbook.io/firmware-security-testing-methodology/
3. Resources for IoT Security: https://github.com/V33RU/IoTSecurity101
4. Hardware List: https://defcon-nn.ru/0×0B/Hardware%20toolkits%20for%20IoT%20security%20analysis.pdf
5. MQTT Security: https://payatu.com/blog/aseem/iot-security---part-10-introduction-to-mqtt-protocol-and-security