Project Title: Hardware Interface Attack Toolkit

Short Project Description:

A hardware interface attack toolkit for discovering and exploiting vulnerabilities in physical devices. The system focuses on identifying insecure hardware interfaces (e.g., UART, JTAG, SPI, etc), extracting firmware/RAM data, and generating actionable reports for hardening.

Component	Role
Physical Interface Discovery Module	Detects UART, JTAG, SPI ports physically
Logic Analyzer/Probing Module	Taps into electrical signals
Debugging Exploitation Toolkit	Exploits insecure debug ports
Memory Dumping Engine	Dumps flash memory or RAM content
Reporting Engine	Summarizes successful physical attacks

Component Details:

1. **Physical Interface Discovery Module**:
    - Uses multimeter probing, oscilloscope, or automatic pin detection tools (like Shikra).
2. **Logic Analyzer/Probing Module**:
    - Sniffs traffic on identified pins:
        - UART: serial output
        - SPI: flash chip communication
        - JTAG: debugging data
        - Etc
3. **Debugging Exploitation Toolkit**:
    - Exploits unlocked debug ports:
        - JTAG access to RAM
        - Bypass authentication routines
        - Direct CPU control
4. **Memory Dumping Engine**:
    - Dumps:
        - Firmware from external NOR/NAND Flash
        - Memory snapshots via JTAG.
5. **Reporting Engine**:
    - Reports:
        - Found physical attack paths
        - Dumped data and vulnerabilities

Overall System Flow:

- Input: Physical access to the device
- Output: Full report on physical access vulnerabilities
- Focus: **Hardware-layer exploitation**.

Internal Functioning of Each Module:

1. Physical Interface Discovery Module

- **Discovery techniques**:
  - **Visual Inspection**:
    - Look for test pads, unused headers, unpopulated connectors, etc.
  - **Multimeter Testing**:
    - Identify Ground (GND) easily: multimeter continuity check.
    - Measure voltage levels on pins (typically 3.3V or 5V).
  - **Automatic Discovery Tools**:
    - Use tools like **Shikra** or **JTAGulator**:
      - Cycle through pin combinations automatically.
      - Identify UART RX/TX or JTAG TAP pins.
- **Pin Function Detection**:
  - Use oscilloscopes or logic analyzers to recognize typical UART traffic (ASCII recognizable patterns at idle voltage ~3.3V).

---

2. Logic Analyzer/Probing Module

- **Signal Capture**:
  - Use a **Logic Analyzer** (e.g., Saleae) or **Oscilloscope** to:
    - Tap into UART/SPI/JTAG communication.
    - Record bitstreams.
- **Analysis**:
  - Decode captured signals:
    - UART: 8-N-1 serial frames.
    - SPI: Master-Slave protocol data exchanges (MISO, MOSI, CLK).

---

3. Debugging Exploitation Toolkit

- **Exploitation**:
  - **UART**:
    - Get shell access via serial console (admin/root access without credentials if exposed).
  - **JTAG**:
    - Pause/resume CPU.
    - Dump or modify RAM contents.

        - Bypass boot authentication checks by altering memory directly.
- **Toolkits**:
    - OpenOCD + Raspberry Pi
    - UrJTAG for scripting JTAG commands
    - Etc

---

4. Memory Dumping Engine

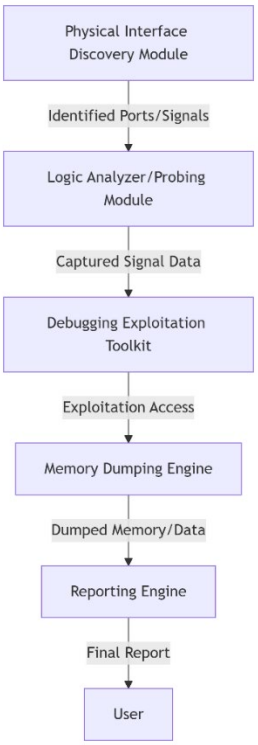
- **Methods**:
    - **SPI flash dumping**:
        - Clip onto chip physically using SOIC8 clip + BusPirate/Flashrom to extract contents.
    - **JTAG RAM dumping**:
        - Direct CPU memory extraction commands.
- **Result**:
    - Extract entire firmware images, keys, bootloaders, etc.

---

5. Reporting Engine

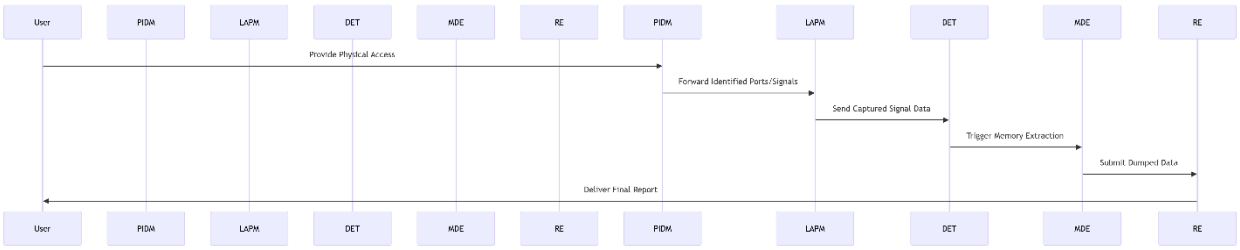
- **Report elements**:
    - Physical access points found.
    - Memory regions extracted.
    - Shell access possibilities.
    - Recommendations for hardening:
        - Disable UART consoles
        - Lock JTAG access
        - Encrypt flash contents
        - Etc

Component Diagram



- **Physical Interface Discovery Module**: Detects hardware interfaces (UART, JTAG, SPI) via visual inspection, multimeters, or automated tools.
- **Logic Analyzer/Probing Module**: Captures and decodes electrical signals (e.g., UART/SPI/JTAG traffic).
- **Debugging Exploitation Toolkit**: Exploits insecure debug ports (e.g., JTAG/UART) to gain control or bypass authentication.
- **Memory Dumping Engine**: Extracts firmware or RAM content via SPI clips or JTAG commands.
- **Reporting Engine**: Generates a report detailing vulnerabilities, dumped data, and hardening recommendations.

Sequence Diagram



1. **User** provides physical access to the target device.
2. **Physical Interface Discovery Module** identifies ports/signals and forwards them to the **Logic Analyzer/Probing Module**.
3. **Logic Analyzer/Probing Module** captures signals and sends data to the **Debugging Exploitation Toolkit**.
4. **Debugging Exploitation Toolkit** exploits debug ports and triggers the **Memory Dumping Engine** to extract data.
5. **Memory Dumping Engine** submits dumped data (firmware, RAM, etc) to the **Reporting Engine**.
6. **Reporting Engine** compiles findings into a final report for the **User**.

Detailed Project Description: Hardware Interface Attack Toolkit

A hardware interface attack toolkit for discovering and exploiting vulnerabilities in physical devices. This toolkit focuses on identifying insecure hardware interfaces (e.g., UART, JTAG, SPI), extracting firmware/RAM data, and generating actionable reports for hardening.

---

1. System Components and Roles

1.1 Physical Interface Discovery Module

Purpose: Identify exposed hardware interfaces (UART, JTAG, SPI) on a target device.
Implementation Details (e.g.):

- **Tools**:
    - **Multimeter**:
        - Identify ground (GND) via continuity testing.
        - Measure voltage levels (3.3V or 5V) to detect active pins.
    - **Automated Tools**:
        - **JTAGulator** or **Shikra**: Automate pin detection for UART/JTAG (e.g., brute-force pin combinations).
        - **Binwalk**: Analyze firmware for debug interfaces.
    - **Etc**
- **Steps**:
    1. Visually inspect the device for test pads, headers, or unpopulated connectors.
    2. Use a multimeter to map voltage levels and identify GND.
    3. Run JTAGulator to auto-detect UART/JTAG pins.

1.2 Logic Analyzer/Probing Module

Purpose: Capture and decode electrical signals from hardware interfaces.
Implementation Details (e.g.):

- **Tools**:
    - **Saleae Logic Analyzer**:

        - Configure for UART (8-N-1, baud rate 115200) or SPI (MISO, MOSI, CLK).
        - Example setup:
          ```
          # Saleae API script to capture UART
          from saleae import automation
          with automation.Manager.connect() as manager:
              device = manager.get_devices()[0]
              device.set_capture_pins([0, 1])  # RX/TX pins
              device.start_capture()
          ```
    - **PulseView**: Open-source logic analyzer software for decoding protocols.
    - **Etc**.
- **Analysis**:
    - Use **sigrok-cli** to decode captured signals:
      ```
      sigrok-cli -i capture.sr -P uart:rx=0:baudrate=115200
      ```

1.3 Debugging Exploitation Toolkit

Purpose: Exploit insecure debug ports to gain control or extract data.

Implementation Details (e.g.):

- **Tools**:
    - **UART Exploitation**:
        - Use **screen** or **minicom** to access serial consoles:
          ```
          screen /dev/ttyUSB0 115200
          ```
    - **JTAG Exploitation**:
        - **OpenOCD + Raspberry Pi**:
          ```
          openocd -f interface/raspberrypi-swd.cfg -f target/stm32f1x.cfg
          ```
        - **UrJTAG**: Script JTAG commands to dump memory or bypass authentication.
    - **SPI Exploitation**:
        - **Flashrom** with SOIC8 clip:
          ```
          flashrom -p linux_spi:dev=/dev/spidev0.0 -r firmware.bin
          ```
    - **Etc**

1.4 Memory Dumping Engine

Purpose: Extract firmware or RAM content via physical or debug interfaces.
Implementation Details (e.g.):

- **Methods**:
  - **SPI Flash Dumping**:
    - Use **Bus Pirate** or **CH341A programmer** to read flash chips.
  - **JTAG RAM Dumping**:
    - Use OpenOCD to dump memory regions:
      ```
      dump_image ram_dump.bin 0x20000000 0x10000  # Dump 64KB RAM
      ```
- **Output**:
  - Firmware images (`firmware.bin`), RAM snapshots (`ram_dump.bin`).

1.5 Reporting Engine

Purpose: Generate reports detailing vulnerabilities and recommendations.
Implementation Details (e.g.):

- **Tools**:
  - **Python + Pandas**: Aggregate findings into CSV/JSON.
  - **LaTeX** or **Markdown** for structured reports.
- **Report Contents**:
  - Identified interfaces (e.g., "UART console exposed on pins 3-4").
  - Extracted data (e.g., "Firmware AES key at 0x0800FF00").
  - Recommendations (e.g., "Disable JTAG in production firmware").

---

2. System Integration and Component Interaction

1. **Discovery**:
   - **Physical Interface Discovery Module** identifies UART/JTAG pins → feeds to **Logic Analyzer**.
2. **Signal Capture**:

    - **Logic Analyzer** records traffic → decodes data (e.g., serial console output).
3. **Exploitation**:
    - **Debugging Toolkit** exploits interfaces (e.g., UART shell access) → triggers **Memory Dumping Engine**.
4. **Data Extraction**:
    - **Memory Dumping Engine** extracts firmware/RAM → sends to **Reporting Engine**.
5. **Reporting**:
    - **Reporting Engine** compiles findings into a PDF/HTML report.

---

3. Implementation Steps (e.g.)

3.1 Hardware Setup

- **Required Tools**:
    - Multimeter, JTAGulator, Saleae Logic Analyzer, SOIC8 clip.
    - Raspberry Pi (for OpenOCD/JTAG).
- **Safety**: Use anti-static mats and grounded tools to avoid damaging hardware.

3.2 Pin Detection

- **JTAGulator Command**:
```
jtagulator -u -v 3.3  # Auto-detect UART pins at 3.3V
```

3.3 UART Exploitation

- **Access Serial Console**:
```
minicom -b 115200 -D /dev/ttyUSB0
```
    - If unauthenticated shell access is granted, run commands like `cat /etc/passwd`.

3.4 Firmware Extraction

- **SPI Flash Dumping**:

```
flashrom -p buspirate_spi:dev=/dev/ttyUSB0 -r firmware.bin
```

3.5 Report Generation

- **Markdown Template**:
```
# Hardware Assessment Report
## Vulnerabilities
- **UART Exposure**: Unauthenticated root shell on pins 3-4.
- **JTAG Enabled**: Full CPU control via TDI/TDO pins.
## Recommendations
- Disable UART in production firmware.
- Encrypt firmware partitions.
```

---

4. Evaluation Criteria

1. **Interface Detection Rate**: Percentage of exposed interfaces identified.
2. **Exploit Success**: Ability to extract firmware or gain shell access.
3. **Report Accuracy**: Correctness of vulnerability descriptions and fixes.

---

5. Ethical and Safety Considerations

- **Ethics**: Use only on devices you own or have explicit permission to test.
- **Safety**: Avoid short circuits; use non-conductive probes for live devices.

---

6. Tools and Resources (e.g.)

- **Discovery**: JTAGulator, Shikra, multimeter, etc.
- **Analysis**: Saleae Logic Analyzer, PulseView, etc.
- **Exploitation**: OpenOCD, UrJTAG, Flashrom, etc.
- **Reporting**: Pandas, LaTeX, etc.
