Ethical Hacking
Computer Science, COSC 3840(701)/6840 (701/721), Fall 2025; 3 credits
Tu: 6:30PM - 9:00PM | Cudahy Hall RM 412

Name: Chirag Malkan
Email: chirag.malkan@marquette.edu

Midterm Project - Firmware Vulnerability Research and Analysis

Learning Objectives

By the end of this project, our goal is for you to be able to:

1. **Apply Ethical Hacking Methodology**
   - Break down firmware analysis into phases: Reconnaissance, Scanning/Enumeration, Gaining Access, Maintaining Access, and Covering Tracks.
   - Recognize how each phase translates when analyzing firmware or software images.
2. **Conduct Firmware Analysis Using Open-Source Tools**
   - Use opensource tools to extract, explore, and triage firmware images.
   - Identify critical files and services
3. **Develop and Document Custom Automation Scripts**
   - Write a reusable script (Python or Bash) that automates part of the firmware analysis pipeline.
   - Capture structured output (e.g., JSON/Markdown) to support reporting and reproducibility.
4. **Identify and Assess Security Vulnerabilities in Firmware**
   - Detect hardcoded credentials, weak configurations, and outdated software components.
   - Cross-reference extracted version strings with vulnerability databases (e.g., NVD, Exploit-DB) to identify known CVEs.
5. **Evaluate Security Risks and Recommend Mitigations**
   - Explain the security implications of findings
   - Propose realistic mitigations aligned with best practices in firmware security.
6. **Communicate Findings Effectively**
   - Prepare a professional technical report (5–7 pages) documenting methodology, findings, risks, and mitigations.
   - Deliver a clear, team-based presentation highlighting methodology, automation, and key vulnerabilities.

Contents

Client Engagement

Penetration Testing Engagement – Firmware Security Assessment
Client: ThanosTech LLC
Motto: *“Where we are inevitable.”*

Client Scope

ThanosTech LLC requests a firmware security assessment of the router firmware images supplied. These firmware images are the core modules that control the spacecraft’s communication relay.

The objective is to uncover potential weaknesses, apply structured security analysis techniques, and create automation for repeatable testing. The **ultimate goal is to improve the product's security posture**. Final deliverables must include a professional report, a demonstration script, and a team presentation. This engagement is designed to help your team grow as penetration testers by practicing real-world skills.

Project Overview

Teams of 2–3 (MAX) assess router firmware (Linux: TP-Link TL-WR841N Vx sample; optional second image; plus one bare-metal blob).

Environment and Tools

- Script should be executable one Ubuntu 22.04 LTS VM
- Python 3.10
- Binwalk and all supporting tools (file, grep, hexdump, strings) installed. If something special, should be in your requirement.txt file)
  - **binwalk** = main extractor.
  - **file + strings + grep** = quick triage inside extracted files.
  - **hexdump** = fallback when no filesystem exists (bare-metal blobs).
  - Other tools are **HIGHLY ENCOURAGED!**

Firmware Binaries Provided

There are three binaries available.

1. Linux firmware images:
   - Name: D-Link DCS-8000LH firmware – IP Cam
   - Name: TP-Link TL-WR841N firmware – Router ; will also be used for demo.
2. Bare metal firmware images:
   - Name: Project FW Image bare-metal
   - Provided as elfs instead of bin

Deliverables

1. **Automation Script**
   a. **Requirement**
      - Automate at least one non-trivial step: extraction + triage, keyword sweep, version harvesting, or report JSON
      - Execute on three firmware images (
      - Language of your choice

   b. **Deliverables**
      - Github Repo link with the firmware (ensure its viewable to be graded)
      - Format of GITHUB REPO:

```
team-XX-fw/
├── fw_triage.py (this is your automation script)
├── README.md
├── findings.json/md
├── extracted/ or _*.extracted/ (filesystem)
├── screenshots/
├── report/final.pdf
└── slides/deck.pdf
```

      - Script + README + output executed on BOTH firmware images.

2. **Report**
   a. **Requirement**
      - Report 5–7 pages.
      - Slides + 5 min talk.
        - intro, methodology, findings, script, recommendations, conclusion.
        - demo of script running
        - top-3 risks with fixes.

   b. **Deliverables**
      - PDF version of word document with page numbers

Report Writing Format

Structure

- Title page: course, project title, team members, date.
- Table of contents.
- Sections flow logically.

Required Sections

1. **Introduction**
    - Who the “client” is (ThaonsTech LLC).
    - Scope: firmware assessment, no live exploitation.
2. **Methodology**
    - Tools used (binwalk, file, strings, grep, hexdump).
    - Workflow: recon / vuln discovery → extraction → scanning → vuln discovery (If there) → scripting.
    - Identify:
        - Kernel image type.
        - Root filesystem format
        - Startup scripts or configuration files.
        - Identify vector table, interrupt handlers, or magic constants
        - Type of architecture code is compiled for ARM, MIPS, or another architecture
    - Document directory structure and key binaries.
    - Make it reproducible (commands, parameters).
3. **Findings**
    - At least 1 vulnerabilities.
    - Evidence: file paths, snippets, screenshots.
    - Security impact + mitigation.
4. **Automation Script**
    - Purpose of the script.
    - How to run it (with sample output). HINT: README MD
    - Value: what it saves time on.
5. **Recommendations**
    - Vendor-focused fixes (update BusyBox, disable telnet, enforce SSH) based on the Findings in Section # 3.
6. **Conclusion**
    - Summarize key risks.
    - Stress importance of firmware hardening.

3. Style Requirements

- Evidence-driven: every claim supported by screenshot, log snippet, or script output.
- Clear, concise, professional tone (avoid “we think,” “maybe”).
- Citations: CVEs, vendor advisories, blogs, NVD or other sources used

4. Length and Format

- 5–7 pages, PDF.
- Figures/screenshots labeled and referenced in text.
- Appendices allowed for extra logs.

5. Presentation Cross-link

- Work on Report → slides → live presentation.
- Slides highlight top 2–3 findings; detailed evidence stays in the report.

Schedule/Milestones [hint: steps]

The following are suggested deliverables for the project.

1. **Recon & Research**
    - Identify the firmware vendor, product type, and version.
    - Setup work environment
    - Research similar devices for known vulnerabilities.
    - Research different things to look for in static firmware security analysis.
    - Research opensource tools that could help you in your objective.
    - Deliverable: 1–2 page background brief with references / screenshots
2. **Scanning & Extraction**
    - Leverage tools researched.
    - Extract filesystem, ID key directories and note files of interest.
    - Deliverable: Directory tree snapshot and notes in your repo.
3. **Vulnerability Discovery**
    - Look for hardcoded credentials, insecure configs, outdated binaries, or sensitive files (keys/certs).
    - Provide evidence and explain why each finding matters.
    - Deliverable: List of plausible vulnerabilities & where did you look? Share impact and suggested mitigations.
4. **Automation Script**
    - Build one reusable script (Python or language of your choice) to automate process items #2, #3 above (e.g., keyword scanning, version harvesting, or Binwalk wrapper). You do not need to re-create other tools, can call other tools from your script. Ensure your script can work on multiple binary files.
    - Needs to work on Ubuntu X.Y VM. Assume Python 3.10 version is available. If other tools needed by script, have an requirements.txt file that lists the dependencies.
    - Deliverable: GitHub Repo containing Script + README + sample output checked into your repo.
5. **Final Report & Presentation**
    - Report (5–7 pages).
    - Presentation (5 min - 7 mins): walkthrough, script demo, and top findings.

Grading Rubric (100 pts)

- Research & Recon — 15 pts
- Scanning & Extraction — 15 pts
- Vulnerability Discovery — 15 pts
- Automation Script — 30 pts
- Report & Presentation — 25 pts

Evaluation Criteria

Category	**Points**	**Attempted but Incorrect**	**Weak**	**Good**	**Score**
1. Recon & Research	15	5	10	15	____
2. Scanning & Extraction	15	5	10	15	____
3. Vulnerability Discovery	15	10	20	25	____
4. Automation Script	30	10	15	20	____
5. Report & Presentation	25	10	20	25	____
Total	100				____

FYI: Not attempted will be 0 points.