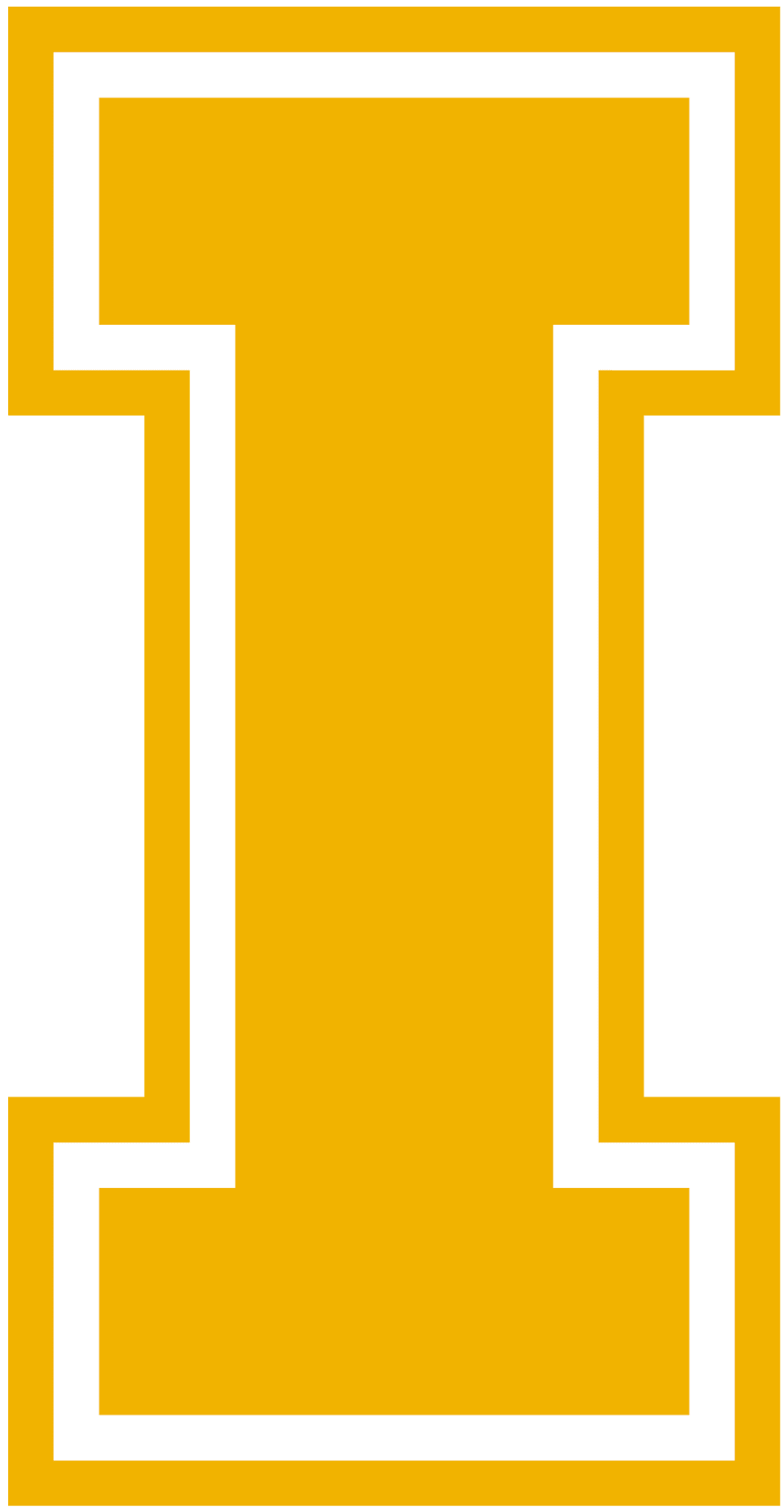
**University of Idaho Cyber Security Tutorials**

**Final Report**

University of Idaho, Department of Computer Science

****

**Mentor:**

Ananth Jillepalli - Washington State University

**Your Team’s Name & Logo**

Darasimi Ogunbinu-Peters

Isabella Sunderman

CptS 432 Cybersecurity Capstone Project

Fall 2025

**TABLE OF CONTENTS -**

[I. Introduction 3](#_Toc208653990)

[II. Project Requirements Specification 4](#_Toc208653991)

[II.1. Project Stakeholders 4](#_Toc208653992)

[II.2. Use Cases 4](#_Toc208653993)

[II.3. Functional Requirements 5](#_Toc208653994)

[II.4. Non-Functional Requirement 5](#_Toc208653995)

[III. Software Design - From Solution Approach 6](#_Toc208653996)

[III.1. Workflow Design 6](#_Toc208653997)

[III.1.1. Overview 6](#_Toc208653998)

[III.1.2. Subsystem Decomposition 6](#_Toc208653999)

[III.2. Data design 7](#_Toc208654000)

[III.3. Reproducibility design 8](#_Toc208654001)

[IV. Test Case Specifications and Results 9](#_Toc208654002)

[IV.1. Testing Overview 9](#_Toc208654003)

[IV.2. Environment Requirements 10](#_Toc208654004)

[IX.1. Test Results 11](#_Toc208654005)

[X. Projects and Tools used 11](#_Toc208654006)

[XI. Description of Final Prototype 11](#_Toc208654007)

[XII. Social Responsibility and Broader Impacts 11](#_Toc208654008)

[XIII. Product Delivery Status 11](#_Toc208654009)

[XIV. Conclusions and Future Work 11](#_Toc208654010)

[XIV.1. Limitations and Recommendations 11](#_Toc208654011)

[XIV.2. Future Work 11](#_Toc208654012)

[XV. Acknowledgements 12](#_Toc208654013)

[XVI. Glossary 12](#_Toc208654014)

[XVII. References 12](#_Toc208654015)

[XVIII. Appendix A – Team Information 12](#_Toc208654016)

[Team Members & Bios 12](#_Toc208654017)

[XIX. Appendix B - Example Testing Strategy Reporting 12](#_Toc208654018)

[XX. Appendix C - Project Management 13](#_Toc208654019)

# Introduction

**Introduction**

Cybersecurity education is increasingly recognized as a critical need; however, many academic programs remain heavily focused on theory rather than practical applications. A gap is created between what students learn in classrooms and the hands-on skills required in industry. The CERES (Cybersecurity Education RESources) project was established in 2015 to address this gap by creating open-source tutorials and walkthroughs that help students gain practical experience in core cybersecurity domains.

Our project continues this effort by developing tutorials in the area of Wireless and Distributed Systems security. Specifically, we aim to design step-by-step, reproducible guides that teach students how to analyze, test, and defend against vulnerabilities in wireless networks and distributed environments. Past CERES teams have developed tutorials on topics including desktop penetration testing, web application testing, network vulnerability scanning, and firewall management. Building on this foundation, our team will expand coverage into new areas that align with industry needs, with a focus on WPA2/3 wireless penetration testing and mobile application security testing.

The motivation behind this work is twofold: first, to help students and professionals practice cybersecurity concepts in a safe and structured lab environment, and second, to create resources that make advanced security concepts more accessible to a broader audience. By bridging the gap between theory and practice, these tutorials contribute to workforce readiness and support the growing demand for applied cybersecurity skills.

The goals of our project are:

1. To produce at least two high-quality, reproducible tutorials that walk learners through a wireless or distributed systems security scenario.
2. To ensure the tutorials are documented in a way that allows replication by future students and professionals.
3. To host the tutorials in an open-source repository where they can be easily shared, improved, and expanded.

Our client is Dr. Daniel Conte de Leon, Associate Professor in the Department of Computer Science at the University of Idaho (contact: [dcontedeleon@uidaho.edu](mailto:dcontedeleon@uidaho.edu)), who oversees CERES project collaborations. He has been instrumental in shaping the long-term vision of CERES and ensuring that each iteration of student projects contributes meaningful educational resources.

Our mentor is Dr. Ananth Jillepalli, Scholarly Assistant Professor in the School of Electrical Engineering and Computer Science at Washington State University (contact: [ananth.jillepalli@wsu.edu](mailto:ananth.jillepalli@wsu.edu)). Dr. Jillepalli provides weekly coaching, guidance on technical direction, and feedback to ensure that our tutorials meet academic and client expectations.

Together, this collaboration between WSU and the University of Idaho provides both academic oversight and real-world applicability, ensuring that the deliverables are practical, replicable, and valuable to the cybersecurity community.

# Project Requirements Specification

Building on the project background, motivation, and goals described in the introduction, we now define the concrete requirements that guide our work. These requirements ensure that our tutorials not only address the gap between academic theory and industry practice but also remain reproducible, accessible, and aligned with the broader mission of the CERES project.

## Project Stakeholders

**Client: Dr. Daniel Conte de Leon (University of Idaho, Department of Computer Science)** Needs tutorials that bridge the gap between academic coursework and industry-required hands-on skills. He expects deliverables to be reproducible and directly helpful in teaching cybersecurity.

**Mentor: Dr. Ananth Jillepalli (Washington State University, EECS)** Needs consistent updates and demonstrations that show progress. Ensures the tutorials meet academic rigor and are suitable for distribution through CERES.

**Students (WSU and University of Idaho, current and future users)** Need easy-to-follow, step-by-step tutorials with screenshots, commands, and explanations that let them practice cybersecurity without requiring extensive prior experience.

**Cybersecurity Education Community (CERES project)** Needs open-source, reusable, and scalable resources that expand the CERES library of cybersecurity tutorials for classroom, workshop, and self-study use.

## Use Cases

A learner discovers a tutorial in the CERES repository.

The learner follows the documented steps, using screenshots and commands to reproduce the task.

The learner performs all activities in a safe, non-malicious, authorized environment.

The learner successfully gains hands-on knowledge that reinforces their theoretical coursework and contributes to their cybersecurity skill set.

## Functional Requirements

Tutorials must include all tools needed for completion (e.g., Aircrack-ng, Wireshark, Hashcat, hcxtools, SQLMap, Nmap).

Each tutorial must be reproducible across different machines (e.g., Kali Linux VM, RADICL lab environment).

All commands and workflows must be documented in a step-by-step manner, with no steps omitted.

Tutorials must incorporate screenshots and annotations at key points.

Each tutorial must include an “Ethics and Scope” section to emphasize the responsible use of resources.

Deliverables must be hosted in GitHub for open access and version control.

## Non-Functional Requirement

**Reproducibility:** Tutorials should yield consistent results on different but compatible environments.

**Comprehension:** Instructions must be clear and accessible to learners with only basic Linux skills.

**Practicality:** Tutorials should strike a balance between depth and feasibility, focusing on tasks that are achievable in lab environments.

**Open-Source Nature:** Distributed under a permissive educational license (CC-BY-NC-ND) to allow sharing and reuse.

**Responsible Transmission of Knowledge:** All content must emphasize ethical boundaries, legal constraints, and intended educational use.

# Software Design - From Solution Approach

This section describes the final design of our project, which is structured around producing reproducible cybersecurity tutorials for the CERES community. The system is not a traditional software application but a framework of tutorials, tools, and supporting resources organized in a GitHub repository. Our design emphasizes reproducibility, modularity, and educational usability.

## Workflow Design

Revise and include Section II from your Solution Approach report here. Provide the block diagram of your architecture and give a brief description of it.

### Overview

At a high level, our project workflow follows these steps:

1. **Research and Scoping:** Identify relevant topics in wireless penetration testing (WPA2/3) and mobile application security (static and dynamic analysis).
2. **Experimentation and Data Collection:** Perform controlled lab experiments, using sample packet captures for WPA2/3 and sample APKs for mobile app analysis.
3. **Tutorial Development:** Translate technical steps into tutorials with screenshots, annotated commands, and ethics disclaimers.
4. **Review and Iteration:** Share drafts with mentor and client for feedback, update based on academic and technical requirements.
5. **Delivery:** Publish tutorials in the open-source GitHub repository, organized into Notes, Drafts, Experiments, and Final Tutorial directories.

This workflow allows both team members to develop tutorials in parallel while maintaining a shared structure and quality standard.

.

### Subsystem Decomposition

Our project decomposes naturally into two tutorial subsystems:

* **Subsystem A: WPA2/3 Wireless Penetration Testing (Darasimi)**
  + Focus on WPA2 handshake capture, analysis, and password cracking using tools such as Aircrack-ng, Wireshark, hcxtools, and Hashcat.
  + Incorporates a Wi-Fi lab setup plan (hardware shortlist, sample PCAPs, live-capture workflow).
  + Includes reproducible steps using public sample captures before live testing.
* **Subsystem B: Mobile Application Security Testing (Isabella)**
  + Focus on static and dynamic analysis of Android applications and iOS applications.
  + Tools include MobSF (Mobile Security Framework) and Frida (for runtime instrumentation).
  + Tutorials will walk learners through inspecting APK and IPA files for insecure configurations and monitoring runtime behavior in an android emulator.

Both subsystems feed into the **FinalTutorial** directory as polished, step-by-step, reproducible guides.

## Data design

Our project does not involve designing custom databases, but does require handling structured experiment data and captures:

* **For WPA2/3 tutorials:**
  + **PCAP files**: Contain WPA2 handshakes for Wireshark and Aircrack-ng analysis.
  + **CSV exports** from Airodump-ng: Include network metadata (BSSID, channel, encryption type, signal strength).
  + **Wordlists**: Approved password lists for cracking demonstrations (stored securely and referenced in documentation).
* **For Mobile App tutorials:**
  + **APK files**: Test applications used for static analysis and reverse engineering.
  + **Decompiled source files**: Decompiled code and configuration files analyzed in static tutorials.
  + **Runtime logs**: Console output or intercepted traffic during dynamic analysis.

All data is stored in Experiments/ and linked to specific tutorials, ensuring reproducibility.

## Reproducibility design

The user interface for our project is not a graphical application but a structured repository and tutorial design. To maximize reproducibility, we have made the following design decisions:

* **GitHub Repository Structure:**
  + Notes/: Raw research logs and meeting notes.
  + Drafts/: Work-in-progress tutorials and requirements documents.
  + Experiments/: Lab setup instructions, PCAPs, APKs, and test outputs.
  + Final Tutorial/: Polished tutorials ready for delivery.
* **Tutorial Format:**
  + Markdown files with consistent section headers (Overview, Lab Setup, Steps, Ethics & Scope).
  + Embedded screenshots with labeled placeholders until final testing is complete.
  + Code snippets and commands are formatted in fenced blocks for easy copy/paste.
  + Ethics disclaimer included in every tutorial.
* **Use of the Washington State University Cybersecurity Club lab:**
  + Tutorials will include notes about how to replicate experiments with loaned hardware (e.g., Wi-Fi adapters, Android devices/emulators).
  + Students outside the university can still reproduce results using sample files and emulator setups provided in the repo.

This design ensures that tutorials remain accessible, reproducible, and maintainable over time for learners to gain knowledge of every growing discipline.

# Test Case Specifications and Results

## Testing Overview

The primary objective of testing in our project is to ensure that each tutorial is reproducible, accurate, and educationally valuable for students with limited prior experience. Because our “software system” is structured as tutorials rather than a single application, testing focuses on verifying that each documented workflow executes successfully in a controlled environment.

Our testing approach includes:

* **Prototype validation:** Confirming that all commands and steps in the tutorials can be run without errors in a fresh environment.
* **Environment verification:** Ensuring tools install correctly in Kali Linux and sample data (e.g., PCAPs, APKs) can be analyzed.
* **Reproducibility testing:** Running tutorials multiple times on clean VMs to verify consistent outcomes.
* **Educational validation:** Informal peer testing, where students follow the tutorials and provide feedback on clarity and usability.

We are not implementing Continuous Integration (CI) pipelines for this project, as it is not code-driven in the traditional sense. However, we use GitHub version control to track changes and updates to tutorial files. Future work could include automated reproducibility tests (e.g., scripted environment setup and file verification).

Categories of testing:

* **Unit Testing (tutorial-level):** Each command and workflow step is executed independently to verify correctness.
* **Integration Testing:** Steps are run in sequence (e.g., capturing → analyzing → cracking a WPA2 handshake).
* **System Testing:** Entire tutorial run-throughs confirm that a learner can start from the setup stage and reach the final outcome.
* **Usability Testing:** Informal peer testing ensures instructions are understandable and free of ambiguity.
* **Non-Functional Testing:** Reproducibility and portability confirmed by executing tutorials in multiple environments (personal VMs and, later, Cybersecurity Club labs).

## Environmental Requirements

Specify both the necessary and desired properties of the test environment. The specification should contain the physical characteristics of the facilities, including the hardware, the communications and system software, the mode of usage (for example, stand-alone), and any other software or supplies needed to support the test. Identify special test tools needed.

**WPA2/3 Tutorial (Darasimi):**

* **Hardware:** USB Wi-Fi adapter with monitor and injection support (e.g., Alfa AWUS036NHA).
* **Software:** Kali Linux VM (2025.2), VirtualBox/VMware, Aircrack-ng, Wireshark, Hashcat, hcxtools.
* **Data:** Sample WPA2 handshake PCAPs (public) prior to live captures.
* **Mode of usage:** Stand-alone VM or Cybersecurity Club lab with authorized APs.
* **Special tools:** Wordlists for password cracking (approved and scoped ethically).

**Mobile App Security Tutorial (Isabella):**

1. **Hardware:** Standard laptop with virtualization support.
2. **Software:** Kali Linux or Ubuntu VM, Android Studio Emulator, MobSF (Mobile Security Framework), APKTool, JADX, Frida.
3. **Data:** Sample APKs (intentionally vulnerable test apps).
4. **Mode of usage:** Emulator-based analysis (no live production apps).
5. **Special tools:** Debugging frameworks and mobile runtime instrumentation utilities.

## Test Results

Include your prototype test results from your “Test Case Specifications and Results” document, as updated with your current project status.

# Projects and Tools used

# Description of Final Prototype

# Social Responsibility and Broader Impacts

Social responsibility: Identify informed judgments that you made for this project based on legal and ethical principles. Discuss if those judgments are in line with your responsibilities as a computing/software / cyber engineer. If not, what prompted you to make judgments that are contradictory to your professional responsibilities?

Broader impacts: highlight how your work aligns with the WSU EECS’s goals of benefiting society and broadening participation in STEM. Describe not only how your project addressed a real-world problem (briefly state the problem) but also emphasized accessibility, inclusivity, and/or educational outreach. If you engaged with local communities through user testing and feedback sessions, mention that. If you collaborated with multiple teams from various departments or groups, mention that. If you documented your project and released it freely to support open-source missions, mention that. If your project outcomes contribute to technological innovation and provide a foundation for future student-led research and community-based applications, mention that.

# Product Delivery Status

# Conclusions and Future Work

## Limitations and Recommendations

Limitations to this project include resource access; although the tools and tutorials themselves are open-source, performing the tasks requires specific hardware.

## Future Work

Include a conclusion and discuss the future work. Future work should include a discussion on how to extend this project. For those of you who are self or EECS-sponsored, discuss commercialization possibilities.

# Acknowledgements

Thank the people who have contributed to your project. Also, thanks to your sponsors.

# Glossary

# References

# Appendix A – Team Information

# Team Members & Bios

**Darasimi Ogunbinu-Peters** is a cybersecurity student at Washington State University pursuing a Bachelor of Science in Cybersecurity. His technical interests include wireless security, penetration testing, and digital forensics. For this project, Darasimi was responsible for designing and implementing the WPA2/3 Wireless Penetration Testing tutorial. His contributions include researching WPA2/3 handshake vulnerabilities, setting up lab experiments with Aircrack-ng, Wireshark, and Hashcat, and documenting reproducible workflows that emphasize ethical use and reproducibility.

**Isabella Sunderman** is a cybersecurity student at Washington State University pursuing a Bachelor of Science in Cybersecurity. She is interested in machine learning, network architecture, and system administration. For the project, Isabella was responsible for designing the Mobile Application Security Tutorial on both static and dynamic scanning. Contributions include researching Mobile app security tools and resources, setting up Virtual Machines, and setting learning objectives and outcomes of tutorials.

# Appendix B - Example Testing Strategy Reporting

1. Identify the requirements being tested.
2. Either link to available online test results/or take screenshots of the various system testing results
3. User testing can be demonstrated via survey results, quotes, and a discussion of the feedback received

Test reporting was conducted through informal feedback between team members on tutorials, focusing on the categories of readability and reproducibility.

# Appendix C - Project Management

Weekly Meetings with the instructor/mentor to discuss the previous week's progress and set goals for the current week, task progress, and non-immediate roadblocks. These meetings are designed to touch base. The project mentor also serves in the role of sponsor, clarifying project deliverables and maintaining a close connection to the project and previous contributions.

Weekly Team-only meetings to discuss more administrative, non-technical project tasks. These meetings provide an open space for communication about the mentor meeting and to update the project board, individual sprint deliverables, and allocate work amongst team members. The open forum of discussion also addressed non-immediate questions and communication between the team and mentor.

The team utilized GitHub, Discord, Zoom, and email for communication and project organization. The team's project GitHub repository also includes a GitHub Project where the team creates issues in the repository and tracks them on GitHub. This allowed team members to see what each other were working on without pulling up the commit log or interfering with each other's branches.