CHANDIGARH UNIVERSITY, GHARUAN

Department of Computer Science and Engineering

INTERNSHIP REPORT

Submitted in Partial Fulfilment of
the Requirements for the Degree of
Bachelor of Engineering (B.E.) in Computer Science and Engineering

Internship Program (20 Sep-25 Dec):

Cybersecurity Internship Program – Shentinelix Sphere Pvt Ltd

Task 9

Python Basics and Introduction to PoC Scripting

Internship Report Submitted By:
Vikash Upadhyay

Table of Contents

Executive Summary

This report summarizes my practical learning and accomplishments while studying Python scripting and proof-of-concept (PoC) exploit development. The program focused on developing core Python skills (variables, control flow, functions, data structures, file I/O, modules) and applying those skills to security tasks: adapting PoC code, automating reconnaissance, and safely testing exploit concepts in controlled lab environments. Emphasis was placed on methodical analysis, safe experimentation, and responsible disclosure practices. Deliverables include scripts for automation, PoC translation exercises (conceptual conversion of Metasploit/Ruby code to Python), and documentation of investigative methodology.

Introduction

Automation and scripting are cornerstone skills for modern security practitioners. Python is widely used because it's readable, well-supported, and has libraries for networking, parsing, and automation. Exploit development (writing PoC code) is a complementary skill: analyzing a vulnerability, understanding exploitation techniques, and crafting small programs that demonstrate the vulnerability in a controlled setting. This internship united both subjects: build a solid programming foundation, then safely apply it to read, modify and adapt PoC code—always within legal, lab-only scopes.

- Learn Python fundamentals sufficient to write small penetration-testing utilities.
- Understand how to read PoC and Metasploit module source code, identify the exploitable endpoint and required parameters.
- Translate/convert PoC concepts (Ruby/Metasploit) into Python safely, focusing on proof-of-concept structure rather than weaponized payloads.
- Practice responsible testing inside lab environments (TryHackMe AttackBox) and document methodologies and findings.
- Produce repeatable scripts for benign tasks (recon automation, parsing logs, testing input vectors) and templates for safe PoC writeups.

Overview of the Company

Shentinelix Sphere Pvt. Ltd. is a cybersecurity training and solutions provider that specializes in delivering practical exposure in information security, penetration testing, and cyber defense. The company focuses on empowering students and professionals with real-world skills through structured labs, Capture-the-Flag (CTF) style challenges, and guided mentorship.

During the internship, I was assigned to the Cybersecurity Research and Training Department The organization strongly emphasizes hands-on learning by offering lab environments, curated challenges, and continuous support from experienced security practitioners, ensuring that interns develop both technical competence and a strong research methodology.

Description of Duties

- Completed Python fundamentals exercises: variables, conditionals, loops, functions, lists/dicts/tuples, file read/write, and module imports.
- Built small automation scripts: parse log files, batch HTTP checks, basic port-reachability scans (safe wrappers around nmap), and CSV report generation.
- Studied Metasploit modules and PoC materials to learn exploit anatomy: discover target input, authentication flow, vulnerable parameter, and expected effect.
- Practiced converting PoC logic from Ruby to Python (conceptual & structural translation), implementing secure input handling and logging for reproducibility.
- Wrote lab PoC wrappers that demonstrate *concepts* (e.g., sending crafted POSTs and verifying controlled response differences) without deploying destructive payloads.
- Documented each PoC: objective, assumptions, test plan, safe verification steps, and remediation advice.

Accomplishments

- **Python competency:** wrote multiple working scripts demonstrating I/O, HTTP interactions, and simple automation.
- **PoC translation practice:** converted Metasploit/Ruby module logic into Python pseudocode and safe test harnesses (no harmful payloads).
- **Artifact analysis:** extracted and parsed an SQLite lab DB, enumerated user records, and demonstrated how to validate password storage methods.
- **Exploit methodology documentation:** created step-by-step templates to analyze a CVE and craft a responsible PoC (identify vector, reproduce, verify, report).
- **Responsible practice:** followed ROE in every lab, created sanitized reports for findings and remediation recommendations.

Skills Learned

Python Fundamentals (practical)

- **Variables & Types:** strings, ints, floats, booleans, None.
- **Control Flow:** if/elif/else, ternary expressions.
- **Loops:** for, while, comprehension syntax (list/dict comprehensions).
- **Functions & Modules:** def functions, parameters/defaults, return values, module import patterns.
- **Data Structures:** lists, tuples, dictionaries, sets — choosing the right container for tasks.

- **File I/O:** safe reading/writing with context managers (with open(...) as f:).
- **Exception Handling:** try/except/finally for robust scripts.
- **Third-party libs:** requests for HTTP, sqlite3 for DB parsing, argparse for CLI.

Scripting for Security Tasks

- **Automating reconnaissance:** script to fetch robots.txt, sitemap, and common endpoints (polite, throttled, and only in-scope).

- **Parsing artifacts:** use sqlite3 to read flat DBs and modest reporting (no cracking/dumping sensitive production data).

- **Safe HTTP interactions:** POSTing form data, authenticating sessions, and validating response codes and content patterns.

- **Logging & CLI:** structured logging and command-line options to make scripts reproducible.

PoC / Exploit Development (methodology & safe practice)

- **Vulnerability analysis workflow:** read PoC/CVE → identify target interface → prepare minimal test case (non-destructive) → reproduce results → document.
- **PoC structure:** initialization (params), authentication (if needed), exploitation attempt (crafted request), verification (non-invasive check), cleanup/logging.
- **Translating PoC code:** map roles rather than copy code — convert logic, keep parameterization, avoid payloads that cause persistent harm.
- **Encoding & transport:** learned when URL-encoding or Base64 is needed to safely carry special characters in URIs or form fields (example shown conceptually below).
- **Ethics & disclosure:** responsible disclosure steps and safe testing rules reinforced.

Challenges Faced

- Converting PoC logic across languages (Ruby → Python) sometimes required understanding library differences (session handling, cookie formats).
- Managing encoding and character escaping reliably when constructing URIs.
- Avoiding temptation to run destructive payloads — forced a disciplined approach to "proof" vs "exploit".
- Ensuring reproducibility across lab environments (different Python versions, missing modules).

Conclusion

This combined study of Python and PoC development delivered solid foundations: reliable scripting skills plus a disciplined approach to understanding and safely demonstrating vulnerabilities. Next steps I plan to pursue:

- Build more reusable security automation tools (credential checkers, recon pipelines) with proper rate-limits.
- Learn secure Python patterns (avoid eval, sanitize inputs) and more advanced networking modules.
- Study binary exploitation and safer sandboxed exploit development frameworks.
- Contribute to a sanitized portfolio of lab-only PoC writeups and scripts.

Acknowledgments

I would like to thank the team at **Shentinelix Sphere Pvt. Ltd**. for providing access to lab environments, learning resources, and a supportive environment that encouraged practical learning and skill development. Special thanks to the **TryHackMe** platform and the authors of the Research Methodology room for providing structured, hands-on lessons that significantly enhanced my research and problem-solving skills. Finally, I extend my gratitude to my peers and family for their continuous support and encouragement throughout the internship period.