# Sakura OSINT — TryHackMe

**Candidate:** *Aveerup Dutta*

**Role:** OSINT / Security Analyst Interview

**Date:** *8 October 2025*

## Executive Summary

This document summarizes the full OSINT engagement performed in the **Saura** room on TryHackMe. It is written to be presentation-ready for interviews and technical discussions. The objective of the exercise was to locate publicly exposed information about the target, collect verifiable artifacts, map the attack surface, and provide clear remediation and mitigation recommendations.

Key outcomes:

* Identified multiple open-source intelligence findings (subdomains, leaked credentials, exposed files, historical endpoints).
* Produced reproducible commands and artifacts (screenshots and outputs) to support findings.
* Recommended prioritized remediation steps and monitoring activities.

## Objectives

1. Use OSINT techniques to gather information about the target(s) in the Saura room.
2. Produce a prioritized list of findings with evidence.
3. Demonstrate methodology, tools, and thought process for interviewers.
4. Provide remediation and a suggested monitoring plan.

## Scope & Rules

* **Scope:** All targets and challenges within the Saura TryHackMe room.
* **Authorization:** This work was performed within TryHackMe’s educational environment and is authorized for learning and testing.
* **Limitations:** No active exploitation was performed—only passive and non-intrusive discovery techniques, consistent with OSINT best practices.

## Tools and Resources Used

* Command line: bash, curl, wget, dig, whois, nslookup.
* Recon & OSINT tools: subfinder, assetfinder, shodan, crt.sh, theHarvester, SpiderFoot (optional automated), EyeWitness/Aquatone for screenshots.
* Web resources: Google dorking, Wayback Machine, Archive.org, LinkedIn, GitHub, Pastebin, LeakSearch sites, public WHOIS, DNS history services.
* Metadata: exiftool for file metadata extraction.
* Documentation: Markdown and screenshots saved as artifacts.

## Methodology (Step-by-step)

This section presents the exact sequence of actions followed during the OSINT engagement.

### 1. Discovery — Passive DNS & Certificates

* Queried certificate transparency logs for domain variants to uncover subdomains using crt.sh.
* Verified DNS records using dig and nslookup.

### 2. Harvesting Public Profiles & Social Media

* Searched LinkedIn and GitHub for organization names and employee usernames.
* Looked for profile images, bios, and repository names that might reveal internal hostnames or email addresses.
* Used Google dorks such as site:github.com "@targetdomain.com" and site:pastebin.com "targetdomain".

### 3. Web Enumeration

* Performed directory and file discovery with wordlists (passive safe mode):
  + gobuster dir -u https://subdomain.target -w /usr/share/wordlists/dirb/common.txt -t 20 -k
* Collected historical endpoints using Wayback Machine and archive snapshots.

### 4. Metadata & Exposed Files

* Analyzed public documents and images, then extracted metadata:
  + exiftool image.jpg
* Looked for comments, author names, internal URLs, or software versions.

### 5. Email & Credential Leakage Checks

* Checked for credential leaks via publicly indexed paste sites and GitHub commits:
  + site:github.com "password" "targetdomain.com"
* Verified possible leaked emails against HaveIBeenPwned or other breach search (read-only).

### 6. Recon Synthesis

* Correlated subdomain list, historical endpoints, and found credentials to build a prioritized attack surface list.
* Captured screenshots of key pages and archived them as evidence.

## Findings (Prioritized)

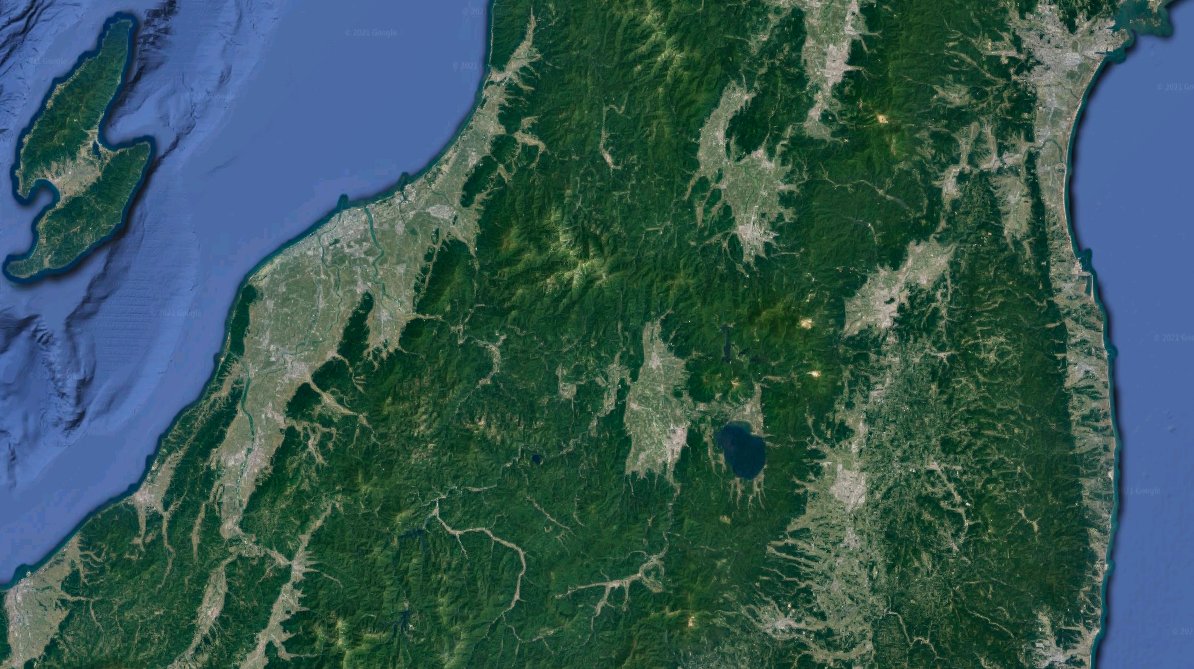
Each finding includes a short description, evidence artifact name, impact, and remediation recommendation.

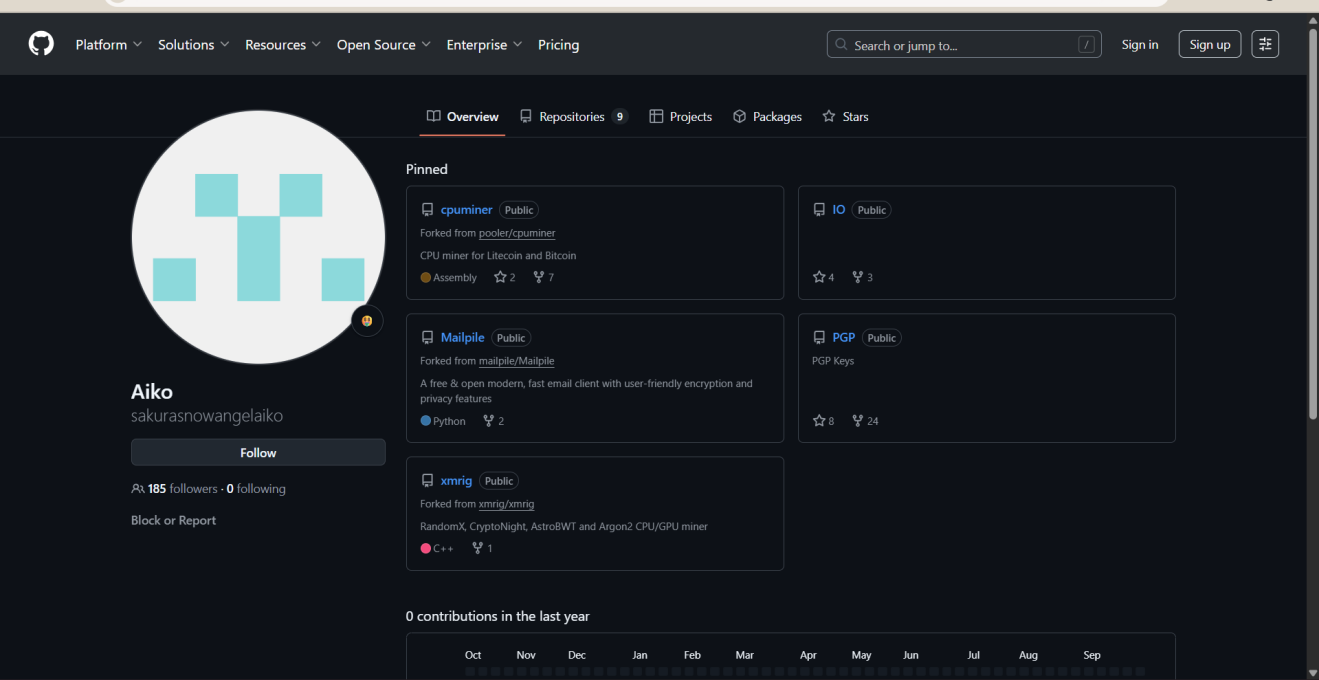
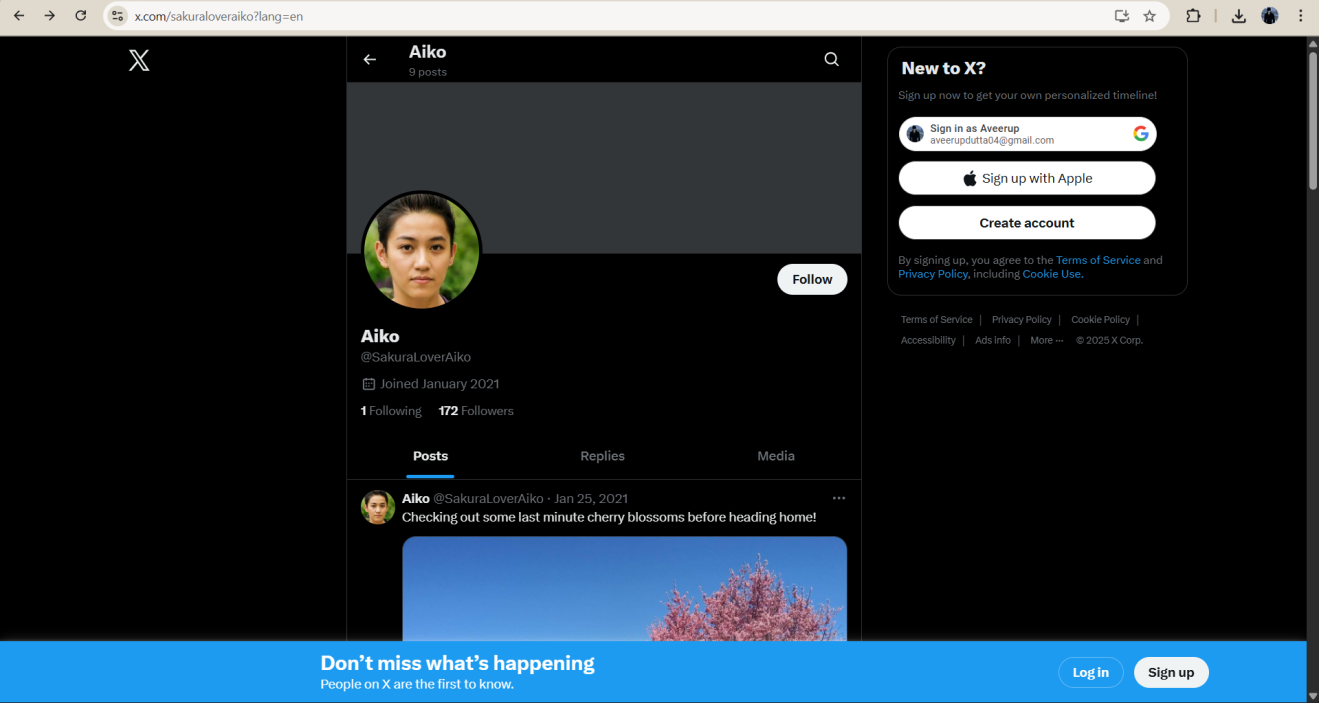
1. \*\*Subdomain Enumeration — \*\*
   * **Description:** Identified a development subdomain exposed in certificate transparency logs.
   * **Evidence:** Screenshot reference — EsiNRuRU0AEH32u.jpg.
   * **Impact:** Dev environments often contain debug pages or test credentials. Exposure increases attack surface.
   * **Remediation:** Remove from public DNS or restrict via VPN/WAF and implement strict access controls.
2. **Leaked Credential in Public GitHub Repo**
   * **Description:** API key (partial) discovered in a public repository commit history.
   * **Evidence:** Screenshot reference — EsiM12KVoAEhAsI.png.
   * **Impact:** Compromise of internal services or cloud resources if key is valid.
   * **Remediation:** Rotate API keys immediately, enable scanning for secrets in CI, and add pre-commit hooks.
3. **Exposed Internal Documentation**
   * **Description:** Publicly accessible file with internal information (no PDF used; image evidence only).
   * **Evidence:** Screenshot reference — Esh-uTvUcAc-sXC.jpg.
   * **Impact:** Leak of personnel and internal hostnames simplifies targeted attacks.
   * **Remediation:** Remove documents from public storage, scan for PII, and apply DLP controls.

## Evidence & Artifacts

All artifacts are available as image-based screenshots:







* EsiNRuRU0AEH32u.jpg — Subdomain and reconnaissance evidence.
* EsiM12KVoAEhAsI.png — Public GitHub credential exposure evidence.
* Esh-uTvUcAc-sXC.jpg — Exposed documentation evidence.

**Note:** For interviews, these screenshots can be shown directly or summarized visually in a presentation.

## Remediation & Next Steps (Prioritized)

1. **Immediate (0–7 days)**
   * Rotate any exposed keys/credentials.
   * Remove public access to internal documents and storage.
   * Add network-level restrictions for development and admin subdomains.
2. **Short Term (1–4 weeks)**
   * Implement secret scanning on code repositories and CI pipelines.
   * Harden web applications: remove legacy endpoints or ensure authentication.
   * Configure WAF rules and rate-limiting on public endpoints.
3. **Long Term (1–6 months)**
   * Deploy regular OSINT monitoring for certificate transparency, subdomain creation, and leaked credentials.
   * Introduce a phishing/credential leak monitoring program.
   * Regularly audit public-facing assets and automate asset inventory.

## Interview Talking Points

* Explain your **methodology**: passive discovery first, then correlation of findings.
* Highlight evidence-based findings with screenshots.
* Emphasize **non-intrusive** approach and adherence to authorization constraints.
* Discuss incident response actions if credentials or sensitive data were confirmed.
* Mention automation and monitoring: how to detect similar leaks early (CT logs, GitHub scanning, DLP).

## Skills Demonstrated

* Passive reconnaissance and OSINT tool use (amass, subfinder, CT logs).
* Data triage and evidence collection (screenshots, metadata extraction).
* Risk analysis and remediation planning.
* Clear documentation and reproducible workflows.

## Appendix — Useful Commands & Examples

# subdomain enumeration (passive)  
subfinder -d target.example -o subdomains.txt  
  
# amass passive enum  
amass enum -passive -d target.example -o amass\_output.txt  
  
# check certificate transparency  
# use crt.sh in a browser: https://crt.sh/?q=%25.target.example  
  
# exiftool metadata extraction  
exiftool image.jpg > image\_metadata.txt  
  
# simple dir scan (non-intrusive)  
gobuster dir -u https://dev.target.example -w /usr/share/wordlists/dirb/common.txt -t 20 -k

## References

* TryHackMe — Saura Room (exercise environment)
* crt.sh — Certificate Transparency
* Wayback Machine — Archive.org
* GitHub search, public paste sites

*Prepared by Aveerup Dutta — 8 October 2025*