Name : Sandhya Singh

Intern ID : 233

Tool Name : Floss + Garbageman

History:

FLOSS was developed by FireEye's FLARE team (around 2015) to extract and deobfuscate strings from malware FLOSS Garbageman is a post-processing tool for FLOSS that extracts and cleans strings from binaries by removing noise and highlighting key IOCs. It speeds up malware analysis with automated filtering, IOC tagging, and multi-format reporting.

Description:

FLOSS Garbageman is a post-processing tool for FLOSS that extracts and cleans strings from binaries by removing noise and highlighting key IOCs. It speeds up malware analysis with automated filtering, IOC tagging, and multi-format reporting.

What Is This Tool About?

FLOSS Garbageman streamlines malware string analysis by automating FLOSS execution, filtering low-value noise, tagging key artifacts (e.g., URLs, IPs, registry keys), clustering related samples, and generating clean, analyst-ready reports in CSV, JSON, or HTML.

⭐ Key Characteristics / Features:

- Cross-platform support (Windows/Linux/macOS)

- Automatic FLOSS orchestration

- Regex & YARA-based keyword filtering

- IOC tagging & ranking (URLs, IPs, registry keys)

- HTML, JSON, and CSV report generation

- Batch mode for processing multiple binaries

- Cross-sample clustering for threat attribution

- Python API for automation

- Fast and lightweight static analysis

- Open-source and easily extensible

🔧 Types / Modules Available:

- String Extraction Engine

- Noise Filter & Cleaner

- IOC Tagger and Scorer

- Report Exporter (CSV/JSON/HTML)

- Batch/Corpus Analyzer

- Python Automation SDK

🎯 How Will This Tool Help?

- Extracts and cleans meaningful strings from malware samples

- Reduces false positives by removing irrelevant strings

- Speeds up incident response by automating IOC collection

- Clusters similar samples for threat hunting and attribution

- Generates ready-to-use investigation reports

🖼 Proof And Concept(PoC) Images:

The FLOSS + Garbageman PoC is illustrated through screenshots showing key features like CLI pipeline runs, “Before vs After” noise reduction, IOC-tagged HTML reports, and score-ranked suspicious strings. It also includes batch cluster views, Regex/YARA tagging, JSON exports, Jupyter/TI notebook integration, Docker one-liner execution, and a minimal Web UI, showcasing the tool’s workflow, flexibility, and reporting capabilities.

📝 15-Liner Summary:

1. Automates FLOSS string extraction

2. Removes noisy or irrelevant strings

3. Tags IOCs such as URLs, IPs, and commands

4. Generates analyst-friendly reports

5. Works on multiple OS platforms

6. Supports batch and automated pipelines

7. Clusters indicators across multiple samples

8. Provides scoring to highlight suspicious strings

9. Python API for advanced workflows

10. Easy integration into IR and TI pipelines

11. Lightweight and open-source

12. Cross-sample analysis for threat attribution

13. Supports various export formats

14. Fast execution with minimal resources

15. Ideal for mal- During malware reverse engineering

⏱️ Time to Use / Best Case Scenarios:

- Before in-depth dynamic analysis

- IOC collection for threat intelligence

- Incident response and forensic investigations

- Batch analysis of large malware corpora

🕵️ When to Use During Investigation:

- Ransomware or trojan investigations

- Campaign attribution and threat hunting

- Unpacking packed binaries

- Initial triage in incident response

- APT analysis and IOC harvesting

📚 Best Person to Use This Tool & Required Skills:

- Best User: Malware Analyst / Threat Hunter / Reverse Engineer

- Required Skills:

o Understanding of malware behaviors and binaries

o Familiarity with static analysis and FLOSS

o Basic knowledge of regex/YARA rules

o Ability to interpret IOCs and reports

❌ Flaws / Suggestions to Improve:

- Limited GUI (mainly CLI-driven)

- Dependent on FLOSS decoding capabilities

- No dynamic runtime behavior analysis

- Scoring requires manual tuning for campaigns

- Could benefit from AI-based IOC classification

✅ Good About the Tool:

- Speeds up manual string analysis
- Generates clean and actionable reports
- Cross-platform and open-source
- Easy to automate with Python scripts
- Lightweight and fast