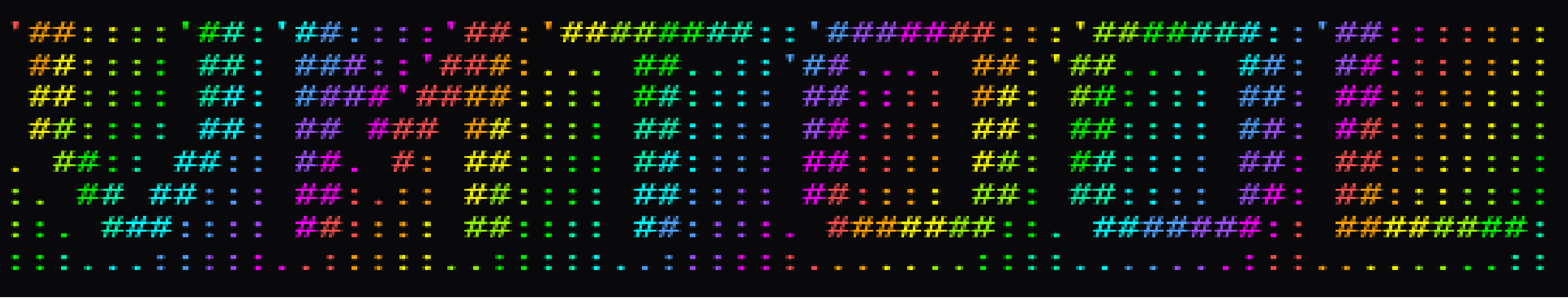


E0-256: Computer Systems Security

VM-Image-Diffing Project

Akash Maji
24212
Computer Science And Automation
Indian Institute of Science, Bengaluru

What Is VMTool?

- Web-based VM disk/memory diffing platform built with Python/Flask UI (frontend)+ C++/pybind11 (backend) for VM analysis, snapshot validation, and VM state comparisons.
- Integrates libguestfs to mount and inspect diverse disk formats (qcow2, vdi, vmdk, raw) without booting guests

Why VMTool?

- VM image diffing is a powerful technique in security forensics and system analysis
- for analyzing malware behavior, investigating security incidents, verifying software integrity, and understanding system changes during software installation or configuration updates

High-Level Architecture [Monorepo structure]

- backend/ (C++ core): C++/pybind11 core exposing libguestfs-powered disk inspection APIs to Python
- frontend/server/ (Flask app): Flask UI with authentication and HTML/JSON endpoints that call the `vmtool` backend
- frontend/vmt/ (CLI): Python CLI package that wraps the same scripts/ops for terminal automation.
- volatility3/ (memory analysis) : Volatility3 toolkit integration for memory dumps, plugins, and report export.

Functionality:

- C++ VMTool.cpp exposes filesystem introspection APIs via **pybind11** to Python server.py
- Flask server imports **vmtool** for browser-driven operations
- Separate CLI (**`vmt`**) auto-discovers scripts for advanced workflows and automation, sharing the same **vmtool** core

Backend & Automation Implementation

Build toolchain:

CMake + GCC/Clang (C++17) + build toolchain, *libguestfs* + *volatility3* dependencies (refer README.md for details)

Outputs shared library (*vmtool*) installed into Python env (install the built using **pip**/**make install**)

Automation:

Scripts in `frontend/vmtool_scripts/` provide command-specific orchestration

Some essential libguestfs commands:

- **guestfs_create()** – Initializes a new libguestfs session for disk operations.
- **guestfs_add_drive_ro()** - Attaches VM disk images in read-only mode for safe analysis.
- **guestfs_launch()** – Boots the appliance to enable filesystem inspection.
- **guestfs_inspect_os()** – Detects installed guest OS roots automatically.
- **guestfs_ls()** – Lists files recursively or within a specific directory.
- **guestfs_statns()** – Fetches extended file metadata like size, permissions, ownership, and timestamps.
- **guestfs_exists()** – Quickly verifies file existence in the guest filesystem.

Additional Support

Support for **qemu-img** conversions and launching QEMU/VirtualBox/VMware instances with standardized flags (CPU, RAM etc.)

Key VMTool Functions

VMTool.cpp inside backend/ is the main file containing all the code that enables leveraging of libguestfs

File Operations

- **get_guestfs_version()** – Validate libguestfs environment
- **list_files_with_metadata()** – Files + size, permissions, timestamps
- **write_files_with_metadata()** – Export metadata to readable file
- **get_files_with_metadata_json()** – JSON output for automation
- **get_file_contents_in_disk()** – Read file content (full/partial)
- **check_file_exists_in_disk()** – Verify existence + stat info

Disk & Block Operations

- **get_disk_meta_data()** – Disk stats + ownership breakdown
- **list_blocks_difference_in_disks()** – Block-level diffing
- **get_block_data_in_disk()** – Inspect specific disk block
- **BlockCompareWorker::operator()** – Parallel block comparison

Memory Snapshot Operations

Linux.pslist Lists processes using linked task structures

Linux.psscan Scans raw memory for process structures (even unlinked)

Linux.lsmod Lists loaded kernel modules

Linux.bash Extracts bash history from memory

Linux.sockstat Enumerates active network sockets

Linux.lsof "List Open Files" from processes

Web Application & UX Implementation

- Flask server (`frontend/server/app.py`) delivers authenticated dashboards, DataTables views, dark/light themes, and flash notifications
- Endpoints cover files listing, metadata, file content viewing, file comparisons, and exports; same functionality exposed via JSON/PDF for reporting

API endpoints:

The python flask server is hosted at http://localhost:8000/

Disk Analysis (libguestfs)

- `/list-files`, `/files-json`, `/meta`
- `/file-contents`, `/file-contents-format`, `/check-exists`
- `/file-compare`, `/files-diff`, `/directory-diff`
- `/compare`, `/block-data`, `/block-contents-compare`

Memory Snapshot Analysis (volatility3)

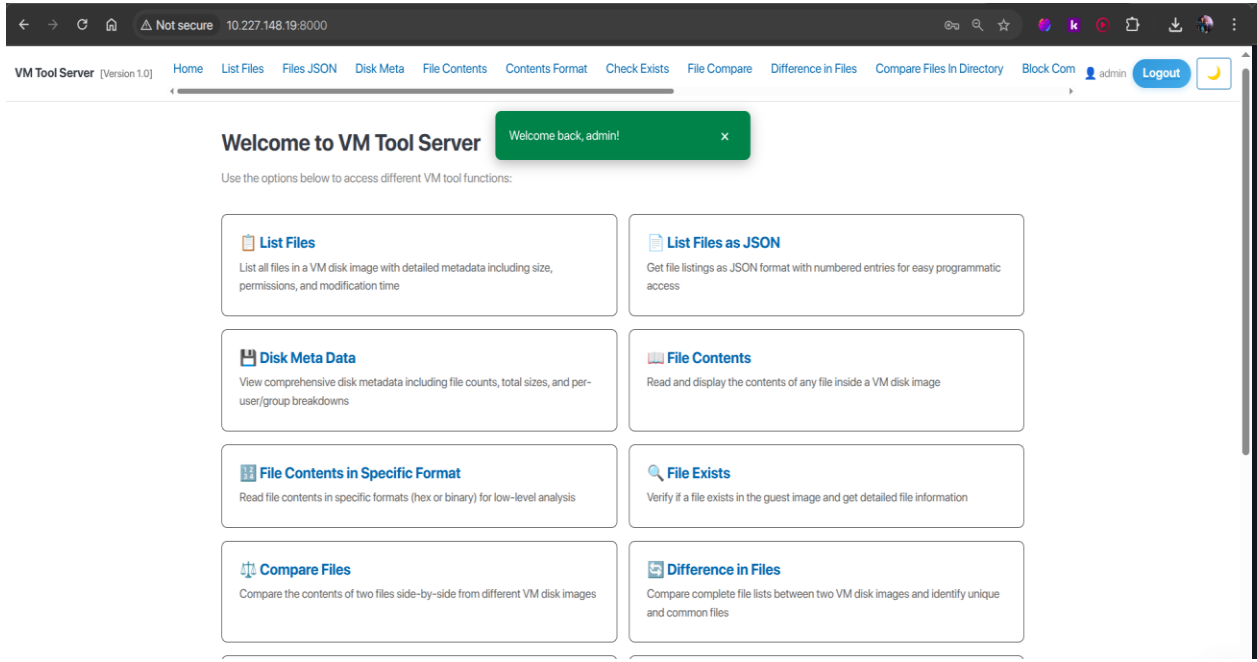
`/volatility/dump`, `/volatility/analyze`,

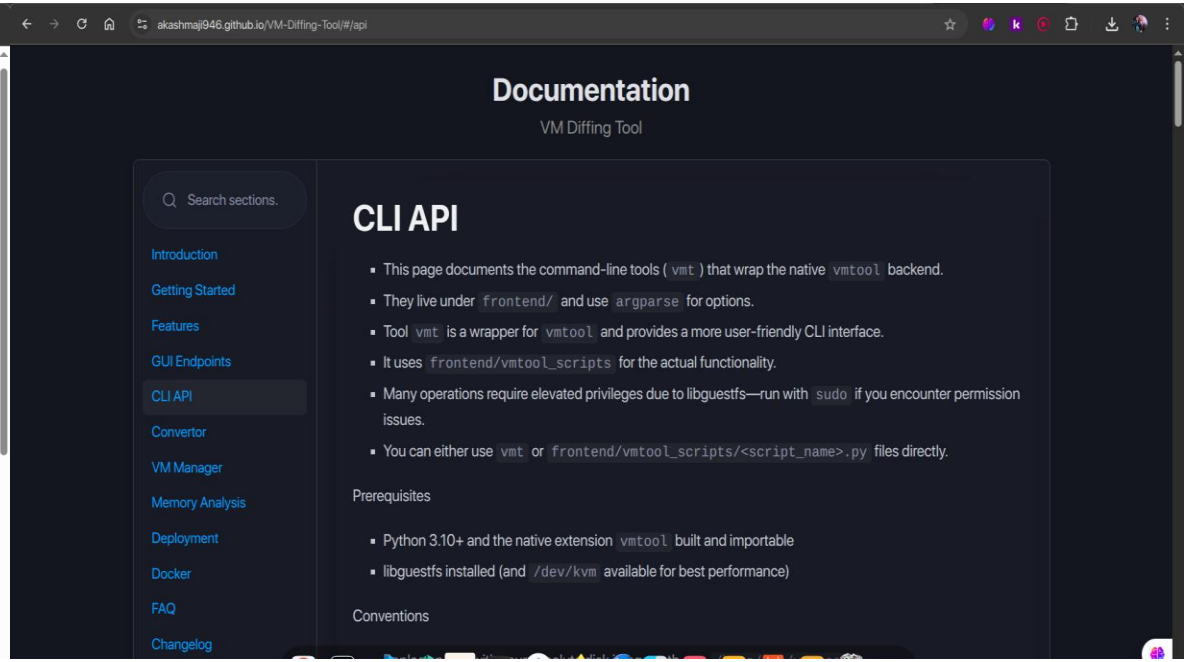
`/volatility/compare`, `/volatility/compare/diff`

Additional Features:

- Flask-Login sessions, email verification using Gmail OAuth + tokenized links
- Deployment flexibility: Docker compose stack (recommended), manual server start, or CLI-only usage

Thank You!





GitHub Link:
https://github.com/akashmaji946/VM-Diffing-Tool
Documentation:
https://akashmaji946.github.io/VM-Diffing-Tool/

Akash Maji
24212
Computer Science And Automation
Indian Institute of Science, Bengaluru