

ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028



A T M E

College of Engineering

DEPARTMENT OF CSE- CYBER SECURITY



LECTURE NOTES

COURSE: INTRODUCTION TO CYBER FORENSICS

COURSE CODE: BCY714A

SEMESTER: VII

COURSE COORDINATOR: Mr. SHASHANK PATIL

(ACADEMIC YEAR 2026-27)

INSTITUTIONAL VISION AND MISSION

Vision

Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

DEPARTMENT VISION AND MISSION

Vision

To nurture competent and ethical cyber security professionals capable of contributing to secure and resilient digital ecosystems.

Mission

M1: To impart strong foundational knowledge in cyber security through effective teaching–learning practices and a continuously evolving curriculum that addresses emerging cyber threats and technologies.

M2: To promote research, innovation, and industry collaboration in emerging areas of cyber security to address real-world digital security challenges.

M3: To instill professional ethics, teamwork, leadership qualities, and a sense of social responsibility in addressing real-world cyber challenges.

M4: To foster a culture of lifelong learning among students by encouraging higher studies, professional development, and active engagement with the global cyber security community.

Program Educational Objectives

PEO1 –Excel as cyber security professionals or entrepreneurs by applying strong fundamentals of cyber security engineering.

PEO2 – Pursue higher studies, research, and professional certifications while engaging in lifelong learning to keep pace with evolving cyber technologies and threats.

PEO3 – Exhibit leadership, teamwork, professional ethics, and social responsibility while addressing cyber security challenges in multidisciplinary environments.

Program Specific Outcomes

PSO1 – Apply principles of computer science, programming, algorithms, operating systems, and computer networks to design and develop efficient and reliable computing solutions.

PSO2 – Identify, analyze, and mitigate cyber threats and vulnerabilities using techniques related to network security, ethical hacking, penetration testing, and digital forensics.

PSO3 – Apply cyber security policies, cyber laws, risk management practices, and incident response strategies to ensure the protection and resilience of digital systems.

FOUNDATIONS OF CYBER FORENSICS

Comprehensive Course Notes

Reference Textbook:

J. Doherty — Wireless and Mobile Device Security
Jones & Bartlett Learning, 2nd Edition, December 2021

Introduction to Cyber Forensics

1.1 What is Cyber Forensics?

Cyber forensics (also called digital forensics) is the application of scientific methods and techniques to identify, collect, preserve, analyze, and present digital evidence from electronic devices and networks. It forms a critical branch of forensic science that deals with the legal process of recovering data from computers, mobile devices, networks, and cloud systems.

Term	Definition
Cyber Forensics	Application of scientific methods to identify, collect, preserve, analyze, and present digital evidence from electronic devices.
Digital Evidence	Any data stored or transmitted in binary form that can be relied upon in court — files, logs, emails, metadata, database records.
Chain of Custody	A documented process tracking evidence handling from discovery through court presentation, ensuring integrity and authenticity.
Forensic Image	A bit-for-bit exact duplicate of a storage device, including deleted files and unallocated space, created without altering the original.

1.2 Core Principles of Cyber Forensics

The following five principles govern all forensic investigations and must be maintained throughout every phase of the process:

- Preservation — Evidence must be preserved in its original, unaltered state. Write-blockers are used to prevent any modification to source media during acquisition.
- Integrity — Hash values (MD5, SHA-1, SHA-256) are calculated before and after acquisition to verify that data has not been modified at any stage.
- Authenticity — Evidence must be provably genuine. Metadata, timestamps, and chain of custody documentation support authenticity claims in court.
- Non-repudiation — Forensic findings must be defensible and independently reproducible using the same methods and tools by another qualified examiner.
- Documentation — Every action taken during the investigation must be meticulously recorded: what was found, when, by whom, and using which tools and procedures.

KEY

Hash verification is the cornerstone of digital evidence integrity. Always compute SHA-256 of the source media BEFORE and AFTER imaging, and record both values in the chain of custody documentation.

1.3 The Forensic Investigation Process

The standard cyber forensic investigation follows six sequential phases. Skipping any phase risks evidence contamination or inadmissibility in court.

Phase 1: Identification

Define the scope of the investigation. Identify all relevant devices, storage media, cloud accounts, and network segments. Obtain legal authority (warrant, consent, or corporate policy) before proceeding.

Phase 2: Preservation

Secure and protect the digital scene. Document the state of all devices (on/off, network status). Create forensic images using write-blockers. Maintain strict chain of custody from this point forward.

Phase 3: Collection

Acquire digital evidence using validated forensic tools. Collect volatile evidence first (RAM, network connections) before non-volatile data. Tools: dd, dcfldd, FTK Imager, EnCase, Magnet Acquire.

Phase 4: Examination

Extract relevant data from forensic images and memory dumps. Apply specialized techniques to recover deleted files, parse file system structures, and locate forensic artifacts within the acquired data.

Phase 5: Analysis

Interpret and correlate the extracted data. Reconstruct timelines of events. Identify suspects, establish motives, and connect evidence to the elements of the offence or incident under investigation.

Phase 6: Presentation

Compile a forensic report suitable for the intended audience (technical, legal, executive). Present findings in clear, objective, defensible language. Expert witness testimony may be required in legal proceedings.

1.4 Types of Digital Evidence

Volatile Evidence (lost when power is removed)

- Contents of RAM — running processes, open files, encryption keys, plaintext passwords
- Active network connections and socket states
- System clock and time zone settings
- Clipboard contents, logged-in user sessions
- Temporary file system mounts and kernel state

Non-Volatile Evidence (persists on storage media)

- Files: documents, images, databases, executables, email archives
- Deleted files recoverable from unallocated disk space
- Log files, browser history, cookies, cached web content
- Registry entries (Windows), configuration files (Linux)
- System metadata: timestamps (MACB), file attributes, ownership

Network-Based Evidence

- Packet captures (PCAP files) from network taps or span ports
- Firewall logs, IDS/IPS alerts, DHCP and DNS server logs
- Email server logs, VPN authentication logs, proxy access logs

1.5 Legal Framework & Admissibility

Term	Definition
Daubert Standard	US legal test requiring expert testimony to be based on sufficient facts, reliable methodology, and correct application to the case facts.
Best Evidence Rule	Requires the original or an accurate, verified copy (with hash) of a document to be presented as evidence in court proceedings.
FRE Rule 901	Federal Rules of Evidence: authentication requirement — proponent must produce evidence sufficient to support a finding that the item is what it is claimed to be.
Fourth Amendment	US Constitutional protection against unreasonable search and seizure — digital investigations generally require a lawful warrant.
GDPR / Data Privacy	Investigators must comply with applicable data protection laws when handling personal data, even during criminal investigations.

1.6 Evidence Acquisition Techniques

Bit-for-Bit (Physical) Imaging

Creates a sector-by-sector exact copy of the entire drive, including deleted files, file slack space, and unallocated space. This is the preferred method for criminal investigations. Always use hardware or software write-blockers.

Logical Imaging

Copies only the accessible file system — files and folders visible to the operating system. Faster than physical imaging but misses deleted data and unallocated space artifacts. Suitable for live, time-sensitive collections.

Live Acquisition (Volatile Data)

Must be performed on running systems before shutdown. Order of volatility: CPU registers and cache → RAM → Network state → Running processes → Disk. Use validated tools: WinPmem, DumpIt, LiME (Linux).

Cloud Evidence Acquisition

Requires legal authority (subpoenas, preservation letters, court orders). Follow the cloud service provider's law enforcement request procedures. Use provider APIs or legal process portals. Evidence may span multiple jurisdictions.

1.7 Anti-Forensics Techniques

Investigators must be aware of deliberate techniques used to hinder forensic investigation:

- Data Destruction: Secure file wiping (DoD 5220.22-M), degaussing, physical destruction of media
- Data Concealment: Steganography (hiding data in images), NTFS Alternate Data Streams, encrypted containers
- Trail Obfuscation: Log deletion/tampering, timestamp modification (timestamping), use of Tor/VPNs
- Counter-Forensics: Anti-forensic software (Evidence Eliminator), volatile-only operations (live CD/USB systems)

WARN	Timestamping: Attackers frequently modify MACB timestamps to confuse investigators. Always compare \$STANDARD_INFORMATION vs \$FILE_NAME timestamps — discrepancies indicate tampering.
-------------	---

1.8 Forensic Report Writing

A forensic report must be objective, accurate, complete, and written at the appropriate level for its audience. It typically contains:

- Executive Summary — Non-technical overview for management and legal counsel
 - Scope and Objectives — What was investigated, the legal mandate, and what was excluded
 - Evidence Inventory — All items received, with hash values, serial numbers, and chain of custody
 - Methodology — Tools used, procedures followed, any exceptions or deviations
 - Technical Findings — Detailed analysis results with supporting evidence and exhibits
 - Timeline of Events — Chronological reconstruction based on digital evidence
 - Conclusions — Objective findings; never speculate beyond what the evidence shows
-

-
- Examiner Credentials — Signature, certifications, and date of report

Windows Forensics

2.1 Windows OS Architecture for Forensics

Windows is the most prevalent desktop operating system in enterprise environments, making Windows forensics an essential skill. Key forensic components include the NTFS file system, Windows Registry, Event Logs, and memory structures managed by the Windows kernel.

- NTFS File System — Primary file system since Windows NT; stores \$MFT, \$LogFile, \$UsnJrnl for forensic analysis
- Windows Registry — Centralized hierarchical database storing OS, application, and user configuration
- Event Logs — Structured records in EVTX (XML-based binary) format capturing security and system activity
- Prefetch / Shimcache / Amcache — Program execution tracking mechanisms
- Volume Shadow Copies (VSS) — Automated snapshots of previous file system states

2.2 Windows Registry — Structure & Forensic Value

The Windows Registry is a hierarchical database that stores low-level settings for the OS and installed applications. From a forensic perspective, it contains a wealth of evidence about user activity, installed software, connected devices, and system configuration.

Term	Definition
HKEY_LOCAL_MACHINE (HKLM)	System-wide configuration for hardware, OS, installed software, and security settings. Stored in hive files: SYSTEM, SOFTWARE, SAM, SECURITY.
HKEY_CURRENT_USER (HKCU)	User-specific settings loaded from NTUSER.DAT. Most valuable for user activity analysis — contains RecentDocs, UserAssist, RunMRU.
HKEY_USERS (HKU)	Contains profiles for all users on the system. Useful for investigating multiple user accounts and distinguishing activity.
HKEY_CLASSES_ROOT (HKCR)	File associations and COM object registrations — helps trace file type usage and application execution patterns.

Registry Hive Files — Physical Locations

Hive	File Path	Forensic Value
SYSTEM	C:\Windows\System32\config\SYSTEM	Hardware config, services, boot settings, USB device history
SOFTWARE	C:\Windows\System32\config\SOFTWARE	Installed applications, OS version, uninstall entries

SAM	C:\Windows\System32\config\SAM	Local user accounts and NTLM password hashes
SECURITY	C:\Windows\System32\config\SECURITY	Security policy, audit settings, cached domain credentials
NTUSER.DAT	C:\Users\<username>\NTUSER.DAT	User activity, RecentDocs, UserAssist, RunMRU, typed URLs
UsrClass.dat	C:\Users\<username>\AppData\Local\Microsoft\Windows	Shell bag data (folder browsing), MUICache entries

2.3 Key Registry Forensic Artifacts

User Activity Artifacts

- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs — Recently opened files per extension
- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU — Commands typed in the Run dialog
- HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\TypedPaths — URLs typed in Windows Explorer address bar

Program Execution Evidence

- UserAssist: HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist — Programs launched, run count, last run time (data ROT-13 encoded)
- ShimCache: HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache — Executable path + last modified time of every file seen by Windows

USB & External Device Tracking

- HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR — Device class, friendly name, serial number, first/last connect timestamps
- HKLM\SYSTEM\CurrentControlSet\Enum\USB — VID (Vendor ID) and PID (Product ID) for connected USB devices
- HKLM\SOFTWARE\Microsoft\Windows Portable Devices\Devices — Friendly names assigned to portable devices

2.4 Windows Event Logs

Windows Event Logs are the primary source of security and activity evidence on Windows systems. Since Windows Vista, logs are stored in EVTX (binary XML) format in C:\Windows\System32\winevt\Logs\

Critical Security Event IDs

Event ID	Event Name	Forensic Significance
4624	Successful Logon	Logon Type: 2=Interactive, 3=Network, 10=RemoteInteractive (RDP). Records username, source IP, logon time.
4625	Failed Logon	Repeated failures indicate brute-force attack. Captures username attempted and source workstation.
4634/4647	Account Logoff	Session duration when correlated with 4624. Type 4634 = forced; 4647 = user-initiated.
4648	Explicit Credential Logon	Logon using alternate credentials — common lateral movement indicator (RunAs, Pass-the-Hash).
4688	Process Creation	Records new process, parent PID, command-line arguments. Requires audit policy. Detect malware execution.
4720	User Account Created	New local account creation — persistence mechanism. Check for unexpected or hidden accounts.
4776	NTLM Authentication	NTLM authentication attempt to domain controller. Detects pass-the-hash and legacy authentication.
7045	Service Installed	New Windows service installed — primary persistence and privilege escalation vector for malware.

2.5 Windows Memory Forensics

Memory (RAM) forensics is critical because it captures evidence that exists only while the system is running — decrypted data, malware that never touches disk, active network connections, and process artifacts.

Memory Acquisition Tools

- WinPmem — Open-source, lightweight; writes to raw .mem or AFF4 format
- DumpIt / Magnet RAM Capture — Single-executable tools; minimal footprint on target system
- FTK Imager — GUI tool capable of live memory acquisition alongside disk imaging
- hiberfil.sys — Windows hibernation file; contains compressed RAM image; accessible offline
- VM Snapshots (.vmem) — VMware/Hyper-V suspend files contain full physical memory

Volatility Framework — Key Windows Plugins

Plugin	Purpose & Output
windows.pslist	List all running processes from EPROCESS linked list — shows PID, PPID, start time, exit time
windows.pstree	Process hierarchy view — reveals orphaned processes (parent process killed) indicating injection
windows.netscan	Active and recently closed network connections — shows PID, protocol, local/remote IP and port

<code>windows.cmdline</code>	Command-line arguments for each process — reveals attacker commands passed to cmd.exe/PowerShell
<code>windows.malfind</code>	Detects code injection using VAD flags (RWX pages) and PE header signatures in memory regions
<code>windows.dlllist</code>	DLLs loaded in each process — detect DLL injection (unexpected DLLs, paths outside System32)
<code>windows.hashdump</code>	Extract NTLM password hashes from LSASS memory — requires SYSTEM privileges on acquisition
<code>windows.dumpfiles</code>	Dump mapped files (DLLs, executables) from memory to disk for further analysis

2.6 NTFS File System Forensic Artifacts

Master File Table (\$MFT)

The \$MFT contains one record for every file and directory on the volume. Each record holds two sets of timestamps: \$STANDARD_INFORMATION (SI) — modifiable by users, and \$FILE_NAME (FN) — maintained by the kernel. Discrepancies between SI and FN timestamps indicate timestomping.

- MACB Timestamps: Modified, Accessed, Changed (\$MFT metadata), Born (created)
- Location: \$MFT is a hidden system file at the root of every NTFS volume
- Tools: MFTECmd (Eric Zimmerman), Autopsy, istat (Sleuth Kit)

Journal Files

- \$LogFile — Transaction journal for NTFS metadata changes; enables recovery of recently modified or deleted MFT entries
- \$UsnJrnl (Update Sequence Number Journal) — Records all file system changes (creates, renames, deletes, security changes) with timestamps and process information

Program Execution Artifacts

- Prefetch Files: C:\Windows\Prefetch\ — Records execution count, last 8 run times, and files accessed during execution
- ShimCache: SYSTEM hive AppCompatCache — Tracks executable paths and modification timestamps for every file seen by Windows
- Amcache.hve: C:\Windows\appcompat\Programs\Amcache.hve — Records SHA1 hashes, first execution time, publisher, and install paths
- Jump Lists: Recent\AutomaticDestinations — Tracks recently opened files and pinned items per application

KEY

The Volatility plugin `windows.malfind` is the primary tool for detecting process injection. It looks for memory regions marked RWX (Read-Write-Execute) that contain a PE header — a strong indicator of reflective DLL injection or shellcode.

Linux Forensics

3.1 Linux in the Forensic Context

Linux powers the majority of web servers, cloud infrastructure, IoT devices, Android platforms, and enterprise back-end systems. Linux forensics requires familiarity with the file system structure, log management systems, user account artifacts, and process/network evidence — all of which differ significantly from Windows.

- Primary forensic distributions: SIFT Workstation (SANS), Kali Linux, CAINE Linux
- Init systems: Systemd (modern) vs SysVinit (legacy) — affects log storage and service management artifacts
- File systems: ext4 (most common), XFS, Btrfs, ZFS — each with unique forensic metadata

3.2 Linux File System Structures

Term	Definition
inode	Data structure storing file metadata: permissions, owner UID/GID, file size, timestamps (atime/mtime/ctime), and pointers to data blocks. The filename is stored in the directory entry, not the inode.
ext4 Journal	Transaction journal tracking file system changes — enables recovery of recently deleted or modified files and metadata changes even after deletion.
Superblock	First block of the file system containing critical metadata: block size, total inode count, free block count, last mount time, last check time, and UUID.
Directory Entry	Maps filenames to inode numbers. When a file is deleted, the directory entry is cleared but inode data may remain — carving tools exploit this for recovery.
Unallocated Space	Disk blocks not currently assigned to any file — may contain remnants of deleted files recoverable through file carving (Foremost, Scalpel, PhotoRec).

Linux Timestamps

Timestamp	Full Name	Description & Forensic Note
atime	Access Time	Last time file content was read. Often disabled via noatime mount option for performance. Check /etc/fstab.
mtime	Modification Time	Last time file data content was changed. Most forensically significant timestamp on Linux systems.
ctime	Change Time	Last time inode metadata changed (permissions, owner, links). NOT creation time — common misconception.

crttime	Creation Time (birth)	File creation time on supported file systems (ext4, Btrfs). Accessible via stat or debugfs on ext4.
----------------	------------------------------	---

3.3 Linux Log Files — Locations & Forensic Value

Authentication & Authorization Logs

- `/var/log/auth.log` (Debian/Ubuntu) — SSH logins, sudo commands, PAM authentication events, su usage
- `/var/log/secure` (RHEL/CentOS/Fedora) — Same as `auth.log` but for Red Hat-based distributions
- `/var/log/wtmp` — Binary file; use 'last' command to view successful login history with duration
- `/var/log/btmp` — Binary file; use 'lastb' command to view failed login attempts with source IP
- `/var/log/lastlog` — Binary file; use 'lastlog' to view last login time for every user account

System & Kernel Logs

- `/var/log/syslog` — General system messages from daemons, kernel, cron, and services (Debian-based)
- `/var/log/messages` — Similar to `syslog` on Red Hat-based systems
- `/var/log/kern.log` — Kernel-specific messages including hardware events and driver activity
- `dmesg` — Kernel ring buffer messages from current boot session; temporary and lost on reboot

Application & Web Server Logs

- `/var/log/apache2/access.log` — Every HTTP request: client IP, timestamp, method, URL, response code, user agent
- `/var/log/apache2/error.log` — Server errors, PHP warnings, module failures
- `/var/log/nginx/access.log` and `error.log` — Similar to Apache logs for Nginx web server
- `/var/log/mysql/` or `/var/log/mariadb/` — Database query logs and error messages

3.4 Systemd Journal Analysis

Modern Linux systems (systemd-based) use `journald` as the primary log manager. The binary journal stores structured log entries from the kernel, boot process, services, and applications.

- Journal location: `/var/log/journal/` (persistent) or `/run/log/journal/` (volatile — lost on reboot)
- `journalctl --list-boots` — Show all recorded boot sessions (b0=current, b-1=previous boot)
- `journalctl -b -1` — Review complete previous boot session — critical for post-incident analysis
- `journalctl -u sshd.service` — All messages from the SSH daemon service
- `journalctl --since '2024-06-01 00:00' --until '2024-06-01 23:59'` — Time-bounded search
- `journalctl -p err` — Show only error-level messages and above (warning, critical, alert, emerg)
- `journalctl _UID=1000` — Show all messages generated by a specific user ID
- `journalctl --verify` — Verify journal integrity to detect tampering

WARN

The journal is stored in binary format. An attacker who deletes individual EVTX/log files can still leave journal artifacts. However, an attacker with root access can clear the entire journal with: `journalctl --rotate && journalctl --vacuum-time=1s`

3.5 Linux User Account Artifacts

Account Database Files

- `/etc/passwd` — User accounts: username, UID, GID, GECOS field, home directory, default shell. World-readable.
- `/etc/shadow` — Hashed passwords with salt, password expiry policy, last change date. Restricted to root.
- `/etc/group` — Group definitions and membership — check for unauthorized addition to `sudo/wheel/admin` groups

User Activity History

- `~/.bash_history` — Command history (typically last 500–1000 commands). Attackers may clear this file.
- `~/.zsh_history` — Zsh history with optional timestamps if `HISTTIMEFORMAT` is set in the shell profile
- `~/.ssh/known_hosts` — Fingerprints of SSH servers this user has connected to — reveals lateral movement
- `~/.ssh/authorized_keys` — Public keys permitted to authenticate as this user — check for attacker-added backdoor keys
- `/var/log/auth.log` — Every `sudo` command records: username, TTY, working directory, and full command executed

3.6 Linux Persistence Mechanisms

Attackers who gain access to Linux systems typically install persistence mechanisms to maintain access after system reboots or password changes. Investigators must check all of the following:

- Cron Jobs: `/var/spool/cron/crontabs/` and `/etc/cron.*/` directories — inspect for unusual scheduled tasks
- Systemd Services: `/etc/systemd/system/` — malicious `.service` files installed here survive reboots
- Shell Initialization: `~/.bashrc`, `~/.bash_profile`, `/etc/profile`, `/etc/profile.d/` — code injected here runs on every login
- SUID Binaries: `find / -perm -4000 -type f` — files with SUID bit run with owner's (often root) privileges
- SSH Backdoors: Unauthorized keys in `~/.ssh/authorized_keys` bypass password authentication entirely
- Rootkits: Modified system binaries (`ls`, `ps`, `netstat`) that hide attacker activity — compare against known-good hashes
- PAM Modules: `/etc/pam.d/` — malicious PAM modules can log or bypass all authentication

3.7 Linux Memory Forensics

Memory Acquisition Tools

- LiME (Linux Memory Extractor) — Loadable kernel module; provides most complete physical memory dump
- AVML (Microsoft) — User-space memory acquisition tool; does not require kernel module insertion
- /dev/mem — Kernel memory interface (access limited by kernel security settings, not recommended)
- Swap analysis: /swapfile or swap partitions may contain pages from terminated processes

Volatility Linux Analysis

- Requires Linux ISF (Intermediate Symbol File) matching the exact kernel version — build with dwarf2json
- linux.pslist — Running processes from task_struct kernel data structure
- linux.pstree — Process hierarchy revealing parent-child relationships and unusual orphans
- linux.bash — Recover bash command history from memory (useful when .bash_history was cleared)
- linux.malfind — Detect anonymous RWX memory regions — common indicator of shellcode injection
- linux.netfilter — Detect malicious kernel-level iptables hooks (rootkit indicator)
- linux.check_syscall — Check system call table for hooking (classic rootkit detection)

3.8 Linux Forensic Investigation Workflow

The following workflow applies to a suspected Linux system compromise. Always prioritize volatile evidence collection before any analysis that might alter system state.

Step 1: Initial Response (Live System)

- Do NOT reboot — volatile evidence will be permanently lost
- Isolate network connection (firewall rule or physical disconnect) to prevent attacker from destroying evidence
- Document running processes: `ps auxf`
- Document network connections: `ss -tnp` or `netstat -tnp`
- Document logged-in users: `w` and `who`
- Acquire RAM using LiME kernel module

Step 2: Evidence Collection

- Create forensic image of storage with `dcfldd` or `dd` — verify with MD5/SHA-256
- Collect all log files from `/var/log/` — preserve original timestamps
- Export systemd journal: `journalctl --no-pager > journal_export.txt`
- Collect user home directories and shell history files

Step 3: Timeline Analysis

- Use log2timeline (plaso) to create a super-timeline from all artifacts
- Analyze with psort.py to filter and sort events
- Focus on authentication events, file system changes, and process executions around the incident window

KEY	log2timeline/plaso can ingest: file system metadata (ext4 MFT-equivalent), auth.log, syslog, bash history, wtmp/btmp, browser artifacts, and many other sources into a single normalized timeline — the most powerful tool for Linux incident reconstruction.
-----	---

3.9 Key Linux Forensic Tools Summary

Term	Definition
log2timeline / plaso	Creates a super-timeline from multiple forensic sources. Essential for correlating events across file system, logs, and user artifacts in a single chronological view.
Sleuth Kit (tsk)	CLI toolkit for file system analysis. Key tools: fls (list files), icat (extract file by inode), mmls (partition layout), fsstat (file system statistics).
Autopsy	GUI forensic platform built on Sleuth Kit. Supports keyword search, hash filtering, timeline analysis, file carving, and report generation.
Volatility Framework	Cross-platform memory forensics. For Linux: requires ISF profile for exact kernel version. Plugins: linux.pslist, linux.bash, linux.malfind, linux.netfilter.
LiME	Linux Memory Extractor — loadable kernel module for acquiring complete physical memory dumps from live Linux systems. Outputs raw, padded, or LiME format.
Foremost / Scalpel	File carving tools that recover deleted files from unallocated disk space based on file header/footer signatures (magic bytes).

Quick Reference Summary

Tools by Task

Task	Windows Tool	Linux Tool
Disk Imaging	FTK Imager, EnCase, dd	dd, dcfldd, Guymager
Memory Acquisition	WinPmem, DumpIt, Magnet RAM	LiME kernel module, AVML
Memory Analysis	Volatility (windows.*)	Volatility (linux.*)
Log Analysis	Chainsaw, Hayabusa, Splunk	grep/awk/sed, journalctl, GoAudit
Registry Analysis	RegRipper, Registry Explorer	N/A (no registry on Linux)
File System	Autopsy, MFTECmd, TSK	Autopsy, Sleuth Kit, debugfs
Timeline	log2timeline, KAPE	log2timeline/plaso, Timesketch
File Carving	Foremost, Scalpel, Photorec	Foremost, Scalpel, Photorec

— End of Course Notes —