

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.

ADVANCED OPERATING SYSTEM FORENSICS & WIRELESS SECURITY

*Comprehensive Reference Manual & Technical Lecture Notes
Covers macOS Forensics, Anti-Forensics, Network Forensics, and
Mobile/Wireless Architectures*

ADVANCED macOS FORENSICS

1.1 File System Architecture: Deep Dive into APFS and HFS+

The evolution of Apple filesystems from Hierarchical File System Plus (HFS+) to Apple File System (APFS) represents a paradigm shift in how data storage, allocation, and metadata management operate under the hood. For a forensic investigator, understanding these foundational changes is paramount to performing low-level recovery, timeline reconstruction, and direct block-level content analysis.

1.1.1 HFS+ Architecture and Legacy Tracking

HFS+ uses a series of structured system files located at the root of the volume layout to map data blocks and metadata. These system files are fundamentally structured around balanced trees (B-Trees). Key structures include:

- **Allocation File:** Tracks the allocation status of every block on the volume using a bitmap where each bit indicates a free or occupied sector.
- **Catalog File:** A B-Tree that defines the folder-and-file hierarchy. It records records for files and directories, tracking name, size, folder parent IDs, and crucial timestamps (Creation, Modification, Attribute Modification, Access).
- **Extents Overflow File:** Tracks data block allocation when a file exceeds its baseline allocation mapping blocks (extents) inside the Catalog File.
- **Attributes File:** Holds extended attributes (xattrs) for items, which often contain critical forensic contextual pointers such as quarantine flags, original download URLs, and enterprise classification metadata.

1.1.2 APFS Containerization, Space Sharing, and Copy-on-Write

Introduced uniformly in macOS High Sierra, APFS replaces physical partition structures with a logical abstraction known as a Container. A single physical partition (GUID Partition Scheme) hosts an APFS Container, which in turn dynamically hosts multiple logical Volumes.

Space Sharing: Unlike HFS+ where volume bounds are static, APFS volumes share the pool of unallocated storage within the parent container. A volume expands and contracts on-demand, which removes traditional physical boundaries and renders physical 'slack space' across partitions nearly non-existent within the container environment.

Copy-on-Write (CoW): APFS does not overwrite blocks in-place during standard file modifications. Instead, it writes updated metadata and modified data payloads to newly allocated blocks. Once the write operation completes safely, the references are updated. This mechanism fundamentally preserves historical chronological artifacts, as unreferenced older blocks remain available in storage until reclaimed by the system trim or garbage collection algorithms.

1.1.3 Forensic Impact of APFS Snapshots and Clones

APFS Clones allow for instantaneous file duplication without consuming additional disk blocks. The clone file shares the exact block pointers as the source file. Only when one file is modified does CoW allocate new blocks for the differential modifications. Investigators looking at block metadata must verify block sharing ratios to prove file duplication versus unique file creation.

Snapshots provide a read-only, point-in-time state representation of a volume. Even if an attacker deletes malware or alters application logs, an automated system snapshot (e.g., generated during Time Machine intervals or OS updates) may preserve the entire historic state of the system volume intact. To list and analyze snapshots via terminal during live response, investigators employ:

```
tmutil listlocalsnapshots /
tmutil listlocalsnapshotdates
```

Forensic Parameter	Hierarchical File System Plus (HFS+)	Apple File System (APFS)
Core Metadata Layout	Static B-Trees (Catalog, Allocation, Extents Overflow)	Object Maps, Object Map B-Trees, Inode Records
Block Modification	In-place overwriting of blocks; metadata updated directly.	Copy-on-Write (CoW). Historical metadata remains until trimmed.
Volume Boundary Definition	Rigid, physical boundaries per defined partition scheme.	Dynamic Space Sharing across multiple volumes in a Container.
Native Encryption	Layered via CoreStorage volume wrapper (FileVault 2).	Native cryptographic implementation at the individual volume level.

1.2 macOS Configuration & Behavioral Artifacts

Understanding specific user-level and system-level behavioral artifacts is critical to mapping user activity, program execution, and establishing intent in malicious actions.

1.2.1 Property Lists (.plist) Deep Dissection

Property lists are configuration files that dictate how the OS and individual applications behave. They exist primarily in two formats: XML (human-readable) and Binary (bplist). Binary plists must be converted during analysis using standard forensic platforms or utility tools:

```
plutil -convert xml1 /Paths/To/target.plist -o /Output/target_converted.xml
```

Key Paths for Persistence Configuration: LaunchDaemons (running as root during system initialization) and LaunchAgents (running within a specific user context upon login) are prime vectors for persistent malware. The core directories that an analyst must comb through include:

- /System/Library/LaunchDaemons/ and /System/Library/LaunchAgents/ (Apple Native)
- /Library/LaunchDaemons/ and /Library/LaunchAgents/ (Global Third-Party Applications)
- ~/Library/LaunchAgents/ (User-specific execution contexts)

1.2.2 The KnowledgeC Database Architecture

The CoreDuet subsystem tracks user activity continuously via the `knowledgeC.db` database. It provides an exceptional high-fidelity chronological view of system events, application execution, device connections, and explicit user interaction history.

- Database Path: `~/Library/Application Support/Knowledge/knowledgeC.db`
- Critical Tables: The `ZOBJECT` table contains raw timestamps, event type codes, and foreign keys. The `ZSTRUCTUREDSTRING` and `ZSOURCE` tables contain corresponding textual metadata detailing exact bundle identifiers, application names, and focus times.
- Forensic Value: Provides an immutable timeline proving that a specific software package was running actively in the foreground, detailing precise epoch timestamps down to the second, matching user keystroke activities or mouse clicks.

1.2.3 Transparency, Consent, and Control (TCC) Framework

The TCC framework governs app permissions regarding privacy-sensitive subsystems (e.g., Camera, Microphone, Full Disk Access, Location Services).

- System TCC Path: `/Library/Application Support/com.apple.TCC/TCC.db`
- User TCC Path: `~/Library/Application Support/com.apple.TCC/TCC.db`
- Investigative Insight: Parsing the SQL structure reveals explicit entries showing which binaries requested access, whether the authorization state was granted (1) or denied (0), and the exact timestamp of modification. Attackers modifying this database using stolen privileges can bypass system controls; discrepancies between execution and database updates indicate tampering.

1.3 The Unified Logging System (ULS) and Diagnostic Log Analysis

Introduced across modern versions of macOS, the Unified Logging System (ULS) centralizes system logs into a high-performance, complex binary tracking mechanism, shifting away from standard plaintext log files (like `/var/log/system.log`).

1.3.1 Log Repositories and File Architecture

ULS stores log events across two principal diagnostic directories:

- `/var/db/diagnostics/`: Contains the actual `.tracev3` compressed binary log files that store indexed log messages sequentially.
- `/var/db/uuidtext/`: Holds metadata logs and text strings mappings that allow the system to parse and reconstruct the compressed binary files into readable events dynamically.

1.3.2 Parsing and Extracting Logs Natively

Because `.tracev3` files are strictly binary and compressed, standard command-line tools like `cat`, `grep`, or `awk` cannot read them. The analyst must use the native `log` utility on a live system or supply an explicit archive path using a forensic workstation. Key query syntaxes include:

```
# Show log entries matching a specific process name over a specified timeframe
log show --predicate 'process == "sshd"' --style syslog --backtrace

# Query logs for system wake/sleep transitions to determine physical interaction
log show --predicate 'eventMessage contains "Wake reason"' --info
```

Analytical Caution: Because of the high volume of modern operating system telemetry, the rollover rate for `.tracev3` files is aggressive. Depending on enterprise utilization, data may persist for only a few days to a week. Ensuring timely forensic capture via `log collect` is critical during live incident response.

1.4 Hardware-Level Security Challenges: T2/M-Series Enclaves & FileVault 2

Apple hardware architectures introduce severe bottlenecks for traditional offline physical acquisitions. The integration of the Apple T2 security chip and Apple Silicon (M1, M2, M3, M4 series) has effectively ended traditional 'dead-box' drive removal forensics.

1.4.1 Secure Enclave Processor (SEP) and Hardware-Bound Encryption

The storage controller is physically embedded into the Apple Silicon / T2 chip architecture. The flash storage chips soldered to the motherboard are encrypted using a unique 256-bit AES Hardware Key that is bound inextricably to the Secure Enclave Processor (SEP).

- Implication: De-soldering or lifting the flash memory storage chips directly off a compromised logic board yields only completely unreadable, heavily encrypted data streams, because the cryptographic keys cannot be extracted from the silicon.

1.4.2 FileVault 2 Encryption Architecture

FileVault 2 operates as a full-disk volume encryption mechanism using XTS-AES-256 protocols. The volume encryption key is unlocked via a user password or an enterprise institutional recovery key (IRK). This workflow generates specific artifacts on disk that can be validated via terminal query interfaces during live assessments:

```
fdsetup status  
fdsetup list -extended
```

CRITICAL FORENSIC PRACTICE NOTE FOR MODERN MAC ACQUISITION:

Because offline logical or physical image creation requires entering correct user credentials or bypassing secure enclaves, modern macOS acquisition must focus on target-based live extraction or native Apple Diagnostic / Share Disk utility mechanisms while maintaining a valid cryptographic context.

ANTI-FORENSICS AND ADVANCED COUNTER-MEASURES

2.1 Covert Data Hiding, Metadata Manipulation, and Timestomping Detection

Anti-forensics represents deliberate operational methods utilized by adversaries to limit, disrupt, corrupt, or altogether prevent the collection and interpretation of forensic artifacts during post-incident analysis.

2.1.1 Extended Attributes (xattrs) and Hidden APFS Resource Inodes

Attackers often utilize filesystem-level structures to bury secondary payloads or command configurations out of plain sight. In macOS, extended attributes can hold substantial payloads. An attacker can write an executable into a file's extended attribute field using simple command streams:

```
xattr -w com.attacker.payload "$(cat malicious_binary)" benign_file.txt
```

To inspect and identify anomalies across extended attributes, forensic analysts evaluate system contexts using the `-l` flags to flag unusual data-heavy namespaces:

```
xattr -l benign_file.txt
```

2.1.2 Timestomping: Manipulating MACB Metadata

Timestomping is the intentional alteration of file creation, modification, access, and metadata change timestamps to throw off sequence timelines. Adversaries mask files by aligning their dates to matches with native system binaries (e.g., matching everything to January 1, 2020). On UNIX-based architectures like macOS, the `touch` utility is heavily leveraged:

```
touch -mt 202001010000 malicious_agent
```

2.1.3 Forensic Detection of Timestamp Discrepancies

While standard folder and application lists might show the manipulated timestamp, anomalies can be caught by evaluating filesystem internal logs. In Windows (NTFS), analysts compare the `$STANDARD_INFORMATION` attributes with `$FILE_NAME` attributes within the Master File Table (MFT). The `$FILE_NAME` timestamp cannot be altered by standard API calls like `touch`, resulting in a clear visual delta when plotted. In macOS APFS, analysts compare Inode creation dates inside the APFS Object Map against transactional checkpoints or Unified Logging System initialization times to spot temporal fragmentation.

2.2 Log Wiping, Selective Artifact Destruction, and File Carving Mitigations

To limit attribution and trace mechanisms, attackers routinely pursue selective destruction of event logs and structural system artifacts.

2.2.1 Wiping Mechanisms and Command Realities

Attackers frequently attempt log zeroing or direct unlinking via destructive scripts:

```
# Zero out user history logs
cat /dev/null > ~/.bash_history
cat /dev/null > ~/.zsh_history

# Force shredding of localized log paths
rm -rf /var/log/*
```

2.2.2 Detecting Log Gaps and Inconsistencies

Wiping actions leave prominent operational signatures. A sudden, unexplained chronological drop or complete gap in sequential event log serial numbers or unified log transaction IDs points directly to administrative tampering. Furthermore, if a log indicates that the logging service itself was stopped or restarted unexpectedly with zero subsequent entries, this establishes a clear window of adversarial presence.

2.2.3 Anti-Carving Techniques and Secure Deletion

File carving relies on locating known file magic numbers (headers and footers) across unallocated storage space. Attackers thwart file carving using specific scrubbing or block overwrite utilities (e.g., `srm`, `shred`, or `dd` random filling).

Anti-Forensic Method	Mechanism of Action	Forensic Detection Strategy
Timestomping	Altering MACB attributes via low-level file APIs or 'touch'.	Compare system logs vs. file structural attributes; check MFT/APFS structural timeline offsets.
Log Zeroing / Clearing	Redirecting null streams to files or clearing event paths.	Examine sequential event ID gaps; inspect volatile memory for persistent log-writer processes.
Block-Level Shredding	Overwriting data blocks multi-pass with pseudorandom characters.	Verify entropy levels of unallocated space blocks; isolate high-entropy non-allocated blocks.

2.3 Memory-Only Malware, Fileless Attacks, and Volatile Memory Reconstruction

Modern malware often evades static signature scanners entirely by operating strictly inside volatile RAM, bypassing traditional storage media attachment entirely.

2.3.1 Process Injection and Living-off-the-Land (LotL)

Fileless attacks utilize legitimate system administration applications (e.g., PowerShell on Windows, native shell binaries or terminal wrappers on macOS) to download payloads directly into memory via network streams. Process injection variants (e.g., Process Hollowing, API Hooking) allow a malicious thread to live inside a completely authorized, signed system container (such as `lsass.exe` or `launchd`).

2.3.2 Volatile Memory Preservation and Analysis

Because memory-only malware disappears when a machine loses power, preserving volatile memory (RAM) is paramount. Analysts pull memory dumps using utilities like LiME or WinPmem on Windows, or specific active agents on macOS, before executing any commands that modify storage state.

Analysis frameworks such as Volatility are then leveraged to inspect the dump. Key plugins help flag process anomalies:

```
# Volatility command syntax example for identifying injected code
vol -f memory_dump.raw windows.malfind

# Enumerate active network connections mapped from memory control blocks
vol -f memory_dump.raw windows.netscan
```

2.4 Strategic Countermeasures, Cryptographic Integrity, and SIEM Detection Playbooks

Countering anti-forensic maneuvers requires a combination of automated defense policies, strict continuous logging infrastructures, and centralized data validation workflows.

2.4.1 Immutable Centralized Log Management

To defeat log wiping, systems must forward event data instantaneously to a remote, read-only SIEM (Security Information and Event Management) collector. An attacker can clear a localized log file, but they cannot erase the event from a secure external log aggregator.

2.4.2 Cryptographic Baseline Auditing

File integrity monitoring (FIM) systems calculate cryptographic hashes (SHA-256) of core operating system binaries regularly. Any unauthorized variation flags an alert immediately, exposing file modifications or trojan replacement attempts.

ADVANCED NETWORK FORENSICS & TRAFFIC ANALYSIS

3.1 Deep Packet Capture Analysis: Advanced PCAP/PCAPNG Dissection

Network forensics analyzes network traffic data to identify security incidents, trace threat progression, and extract evidence. Unlike endpoint tracking, network traffic provides an objective view of communications between infected hosts and external adversaries.

3.1.1 Capture Mechanics via *Tcpdump* and *Wireshark*

Capturing network traffic safely requires deploying continuous collection agents or tapping infrastructure lines. On the command line, `tcpdump` is standard. Analysts enforce Berkeley Packet Filters (BPF) to narrow down high-volume traffic lakes to target areas:

```
# Capture traffic traversing port 80 or 443, excluding an authorized admin IP address
tcpdump -i eth0 -w threat_capture.pcapng 'tcp and (port 80 or port 443) and not host 192.168.1.50'
```

3.1.2 Reassembling TCP Streams and Extracting Payloads

Raw PCAP files contain thousands of individual frames that must be pieced together sequentially. Analysts track sequence numbers (`'Seq'`) and acknowledgment numbers (`'Ack'`) to map conversations. Using automated terminal parsers like `'tshark'`, analysts can reconstruct and extract malicious binaries transferred over unencrypted protocols directly from the packet stream:

```
tshark -r threat_capture.pcapng --export-objects http,./extracted_payloads/
```

3.2 Enterprise Network Logs, Flow Records (NetFlow/IPFIX), and Protocol Fingerprinting

When packet-level PCAP data is unavailable due to storage limits or end-to-end encryption, network logs and metadata become primary sources of evidence.

3.2.1 NetFlow and IPFIX Forensic Values

NetFlow records omit the data payload entirely, tracking only communication metadata (the 'phone bill' of the network). Key fields captured in NetFlow collections include:

- Source and Destination IP Addresses
- Source and Destination Port Numbers
- Layer 4 Protocol Type
- Packet and Byte Counts
- Timestamps for Flow Initialization and Termination

Forensic Application: Flow records are highly effective for tracking internal data staging and large-scale data exfiltration. A sharp, anomalous increase in total bytes transferred from an internal database server to an unknown foreign IP address establishes clear exfiltration metrics, even if the payload is encrypted.

3.2.2 TLS Handshake Analysis and JA3/JA4 Fingerprinting

Because the majority of modern network traffic is encrypted via TLS/SSL, analysts inspect the initial unencrypted portions of the TLS handshake—specifically the Client Hello message.

JA3/JA4 Fingerprinting: Compiles parameters from the Client Hello (e.g., supported cipher suites, extensions, elliptic curve formats) into a MD5 hash. Because specific malware variants employ custom compiled network communication libraries, their JA3 hash remains unique and constant, enabling analysts to identify malicious agents inside an encrypted HTTPS stream without breaking decryption keys.

3.3 Intrusion Detection/Prevention Systems (IDS/IPS) Alert Triage and Log Correlation

IDS/IPS platforms (e.g., Snort, Suricata, Zeek) evaluate raw network streams against signature databases and behavioral heuristics to flag malicious events.

3.3.1 Correlating Alerts via Zeek Metadata Log Aggregation

Zeek transforms raw packet streams into structured tab-separated log structures (e.g., `conn.log`, `dns.log`, `http.log`). This enables programmatic correlation. For instance, an investigator can track an initial alert down a sequential chain: a suspicious DNS request leads to an HTTP connection download, which then initiates an anomalous persistent internal SSH connection.

Log Type	Core Analyzed Elements	Forensic / Threat Extraction Goal
dns.log	Queries, Response Codes (NXDOMAIN), TTL values, Record Types (TXT/A).	Detect Command & Control (C2) setups utilizing Fast-Flux or DNS Tunneling channels.
http.log	User-Agent Strings, URIs, Response Status Codes, Method verbs.	Isolate initial infection vectors, automated staging scripts, or cleartext file downloads.
conn.log	Duration, Bytes, State flags (S0, SF), Connection frequencies.	Identify long-duration active command linkages or horizontal network scanning loops.

WIRELESS AND MOBILE DEVICE SECURITY

4.1 Wireless Security Evolution: WEP, WPA2, WPA3, and Enterprise EAP Implementations

Wireless networking increases the organizational attack surface by extending data access beyond physical walls. Securing and analyzing wireless networks requires a deep understanding of standard 802.11 security architectures and cryptographic authentication frameworks.

4.1.1 Structural Flaws of Legacy Systems (WEP & WPA)

- **Wired Equivalent Privacy (WEP):** Relies on the RC4 stream cipher combined with a short, 24-bit initialization vector (IV). Because the IV is sent in cleartext and rolls over under heavy traffic loads, passive monitoring tools can collect identical IV pairs, allowing attackers to calculate and crack the master key within minutes.
- **Wi-Fi Protected Access (WPA):** Implemented the Temporal Key Integrity Protocol (TKIP) as a temporary fix. TKIP mixed keys on a per-packet basis to resolve WEP's IV reuse flaws, but it still relied on RC4, rendering it vulnerable to structural key-recovery exploits.

4.1.2 WPA2 Architectures and the 4-Way Handshake Vulnerability

WPA2 standardized the use of the Advanced Encryption Standard (AES) cipher coupled with the Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP).

The 4-Way Handshake: This protocol establishes a secure connection by verifying that both the client (Supplicant) and the access point (Authenticator) possess the correct Pre-Shared Key (PSK) without transmitting the key itself. It generates a Pairwise Transient Key (PTK) using random nonces (ANonce, SNonce) and MAC addresses. Attackers capturing this handshake offline can launch dictionary and brute-force attacks against the Pre-Shared Key.

4.1.3 WPA3 Advancements and Enterprise EAP Deployment

WPA3 addresses WPA2's architectural gaps by replacing Pre-Shared Keys with Simultaneous Authentication of Equals (SAE), a secure key-exchange protocol based on Diffie-Hellman curves. SAE blocks offline dictionary attacks by requiring active interaction for every authentication attempt. Furthermore, WPA3 enforces Protected Management Frames (PMF) natively, preventing attackers from forging administrative frames to disconnect clients.

Enterprise Authentication (802.1X/EAP): Enterprise environments shift away from static keys toward centralized RADIUS or TACACS+ authentication servers using the Extensible Authentication Protocol (EAP). Common profiles include:

- **EAP-TLS:** Requires both client-side and server-side digital certificates, providing robust cryptographic security.
- **PEAP (Protected EAP):** Establishes a secure TLS tunnel using a server certificate first, protecting subsequent inner authentication steps (such as MS-CHAPv2 passwords).

4.2 Wireless Threat Frameworks: Rogue APs, Evil Twins, and Vulnerability Exploits

Wireless infrastructure is highly susceptible to proximity-based attacks and signal manipulation.

4.2.1 Rogue Access Points vs. Evil Twin Implementations

- **Rogue Access Point:** An unauthorized physical access point connected directly to an enterprise corporate switch network without IT approval. It bypasses corporate security filters, granting untracked access to the internal network.
- **Evil Twin Attack:** A spoofed access point configured to mirror the exact SSID, MAC address (BSSID), and configuration parameters of a legitimate corporate wireless network. Attackers broadcast high-gain signals to force target clients off the legitimate network and onto the rogue node, enabling full man-in-the-middle (MitM) traffic interception.

4.2.2 Management Frame Deauthentication Attacks

Because legacy 802.11 standards transmit management frames unencrypted, an attacker can spoof a victim's MAC address and broadcast a continuous stream of forged Deauthentication/Disassociation frames. This severs the client's connection instantly, forcing a reconnect cycle that allows the attacker to capture the 4-way handshake.

4.3 Mobile OS Security Paradigms: iOS Sandbox vs. Android Hardening & MDM Frameworks

Mobile operating systems diverge from traditional desktop architectures by implementing strict application containment and restrictive operational permissions by default.

4.3.1 iOS Architecture and Core Containment

Apple's iOS enforces a closed ecosystem centered around security mechanisms:

- **App Sandboxing:** Every app runs within an isolated sandbox environment container. It cannot access files stored by other apps or interact with system resources without explicit user permissions.
- **Mandatory Code Signing:** Every executable binary must be signed by an authorized Apple developer key or enterprise profile, preventing unverified file execution.
- **Secure Enclave Processor (SEP):** A dedicated hardware security processor separate from the primary application CPU. It manages file encryption keys, processes biometric verification (Touch ID/Face ID), and ensures cryptographic protections remain secure even if the primary OS kernel is compromised.

4.3.2 Android Architecture and Permission Hardening

Android relies on an open-source architecture built on the Linux kernel:

- **Application Sandbox via Linux UID:** Each application is assigned a unique Linux User ID (UID) upon installation. The kernel enforces file access controls based on these UIDs, meaning applications run in isolated virtual environments and cannot read or write to each other's storage spaces.

- Google Play Protect and Runtime Permissions: Continuously scans device behavior for anomalies while requiring applications to request explicit user authorization before accessing sensitive system resources (such as Location data or Contacts).

4.3.3 Enterprise Mobile Device Management (MDM) Controls

Organizations manage mobile risk via Mobile Device Management (MDM) platforms. MDM agents enforce corporate security baselines over-the-air, such as requiring storage encryption, setting minimum password complexities, blocking sideloaded applications, and enabling remote data wiping for lost or stolen hardware.

4.4 Mobile Forensics Extraction Methodologies, SQLite Database Dissection & Cellular Logging

Extracting data from mobile devices requires a structured methodology that accounts for storage encryption and hardware controls.

4.4.1 Acquisition Typologies

- Logical Acquisition: Extracts accessible files, application configurations, and metadata records via standard backup APIs. It captures standard user data but misses deleted file blocks or hidden system data.
- Physical Acquisition: Creates a bit-stream copy of the entire physical flash memory container. This enables the recovery of deleted records from unallocated space but is frequently blocked on modern hardware by full-disk encryption and locked bootloaders.

4.4.2 SQLite Database Dissection

The vast majority of user application data on mobile platforms (e.g., chat logs, contact lists, call histories, location tracking coordinates) is stored inside SQLite databases. When evaluating an extracted database file, an investigator must look at both the active database and its secondary journaling files:

- Write-Ahead Logs (.wal): APFS and Android ext4 systems write database modifications to a secondary `.wal` file before merging them into the primary database. Forensic tools parse these `.wal` entries to recover deleted chat rows, historical edits, and deleted messages that have not yet been overwritten.

4.4.3 Cellular Network Artifact Tracking

Mobile devices continuously log cellular connections within internal diagnostic logs and provider databases. Key parameters tracked include:

- International Mobile Equipment Identity (IMEI): A unique hardware serial number that identifies the physical device on any cellular network.
- International Mobile Subscriber Identity (IMSI): A unique identification number bound to the SIM card that maps the user's explicit subscriber account.

- Cell Tower Logs and Call Detail Records (CDRs): Maintained by cellular providers, these records log the precise cell tower sectors a device connected to during calls, texts, or data transmissions, enabling forensic geographic tracking over time.