Ghost Machines on the Macintosh: VM-Concealed Remote Access and Control in Compromised macOS Systems

An Examiner's Guide to the Tradecraft, the macOS Evidence Record, and the Technical Analysis

Author: Christopher L. T. Brown, Technology Consulting in Digital Forensics, AI & InfoSec, November 2026

Executive Summary

A Macintosh that looks clean can be running a second computer. Attackers who gain a foothold on a host increasingly install a headless virtual machine on it, load their tooling and command channel inside the guest, and operate from a position that host-centric endpoint monitoring was never designed to see. The tradecraft is documented: QEMU guests running Tiny Core Linux were dropped on victim hosts by UNC1945 as far back as 2020, and the same construct recurs in ransomware and espionage operations reported through 2026. [1][2][3] This paper explains what ghost machines are, how they arrive, what the compromised Macintosh itself records about them, and how examiners can assemble that record into an identification of a system under continued remote control. It is written to be the working knowledge base behind a Mac-based compromise examination, whether the control channel is a plain remote management agent or a virtual machine hiding in plain sight. [4]

1. The construct is real and repeatable: From Mandiant's UNC1945 QEMU deployments in 2020 through Sophos-documented Payouts King and CitrixBleed2 operations in 2026, attackers have used small headless Linux guests as their private workspace on compromised hosts, carrying the tooling, the tunnels, and increasingly the automated agents that drive intrusions [1][3]. On Apple silicon, a full macOS guest can be run rootless under the system's own Virtualization framework, entirely inside a user-writable directory. [5]

2. The Macintosh records the intrusion on the host: The unified log, the launchd and Background Task Management stores, the TCC privacy database, quarantine event records, and the traces of remote management and virtualization software survive on the host even when the attacker's workspace lives inside a guest. [6][7][8][9][10][11] That record is bounded, and an examiner must understand its limits, but it is the reliable account of what happened. [12][13]

3. Host visibility ends at the hypervisor: Endpoint monitoring sees the virtualizer process, not the guest's processes, memory, or files [14]. The practical consequence: the container is the principal detectible object on the host, and the network is frequently the only observer of what happens inside the guest. [15][16]

4. The analysis methods are established: Operating system log forensics, network-behavior detection of remote access trojans, memory forensics across Intel and Apple silicon, and the forensic study of virtual machines each contribute validated technique [17][16][18][19]. The macOS threat landscape has itself been characterized at scale, with more than 57,000 samples mapped to adversary tactics. [20]

The synthesis for an examiner: a Mac under continued remote control, including VM-concealed control, leaves a host-side record that can answer how access was obtained, how it is maintained, and how it is concealed, within a bounded evidentiary window. The following sections present the tradecraft, the record, and the analysis.



Figure 1. Verified operational benchmarks for the ghost machine problem space. Time-to-exploit per the Techno Security West 2026 lecture and Zero Clock data; UNC1945 dwell per Mandiant; guest footprint and exposed RMM server counts per stated sources.

1. The Ghost Machine: Definition and Tradecraft

1.1 What a Ghost Machine Is

A ghost machine is a virtualization container running on a compromised host, operated headless, and treated by the attacker as a separate system inside the victim's trusted boundary. The definition carries the threat: host process monitoring, host EDR, and host-centric detection logic point at the host, while the adversary operates inside the guest. Guest activity then reads as benign host overhead, or as nothing at all. The tooling is open source, cross-platform, and command-line driven: a virtualizer such as QEMU, configured with CPU, memory, networking, and a small Linux guest image such as Alpine at roughly 150 MB, launched with no graphical display so the user never sees a window. [4][21]

The construct has a documented lineage. Mandiant's UNC1945 operated from pre-configured QEMU virtual machines preloaded with post-exploitation tooling [1]. Sophos and NVISO document QEMU guests launched from disguised image files on Windows hosts [3][22]. On Apple silicon, an unprivileged user can run a full macOS guest under Virtualization.framework with only an ad-hoc-signed binary carrying the virtualization entitlement, and the resulting guest is substantially invisible to the host's security stack. [5][11] Figure 2 presents the lifecycle. [4][1]

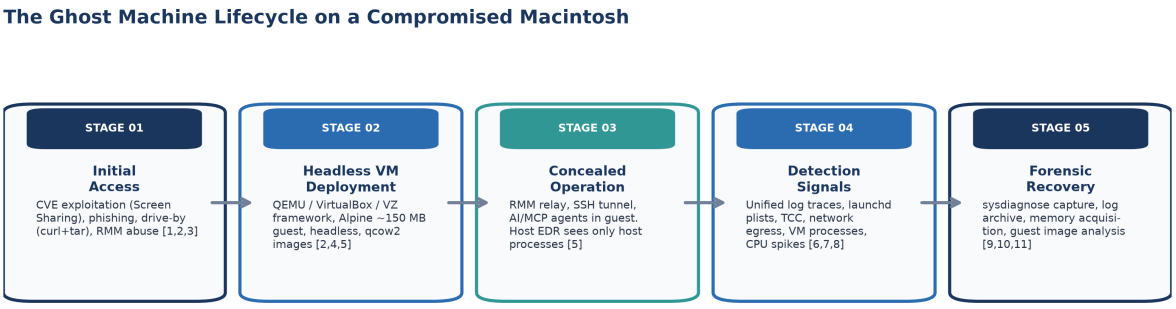


Figure 2. The ghost machine lifecycle on a compromised Macintosh, from initial access through headless VM deployment and concealed operation to the host-side detection signals and forensic recovery paths examined in this paper.

The threat model has two gradations. At the thin end stands plain remote control software: a remote management agent installed deliberately or by deception, visible if looked for, carrying the same access. At the thick end stands the ghost machine proper, whose guest survives reboots, endpoint agent reinstallation, and the casual process scan. Both ends matter to an examiner, because the same intrusion frequently uses both. [14][2]

1.2 Documented Campaigns: From UNC1945 to Payouts King

The record begins in earnest with Mandiant's UNC1945 analysis published in November 2020. UNC1945 obtained access to Solaris, Windows, and Linux systems through third-party networks, exploited the Oracle Solaris PAM zero-day CVE-2020-14871, and maintained access with SSH port forwarding. On multiple hosts the group dropped a custom QEMU virtual machine running Tiny Core Linux, preloaded with Mimikatz, PowerSploit, Responder, CrackMapExec, and post-exploitation tooling, launched by a start.sh script whose TCP forwarding, chained through the SSH tunnels, obscured interaction with command infrastructure. Mandiant measured a dwell time of approximately 519 days and observed log-cleaning utilities LOGBLEACH and STEELCORGI. [1]

By 2025 and 2026 the pattern is routine. Sophos documented Payouts King, first observed November 2025, where a TPMProfiler scheduled task launched QEMU as SYSTEM with an Alpine 3.22 guest drawn from vault.db and bisrv.dll-disguised images, AdaptixC2 inside the guest, and reverse SSH tunnels on ports 32567 and 22022, with follow-on reporting tying the operators to former Black Basta affiliates. [3][23] The distinct CitrixBleed2 campaign, tracked as STAC3725 on CVE-2025-5777 in February 2026, installed ScreenConnect for persistence and then dropped a QEMU Alpine VM that compiled Impacket, BloodHound, and Metasploit inside the guest. [3] Bitdefender's Curly COMrades analysis documented Hyper-V enabled on a victim Windows machine via dism, with a hidden Alpine Linux VM of 120 MB disk and 256 MB RAM named WSL, hosting a reverse shell and reverse proxy tunneled through the host's NAT. [24] Microsoft Defender Experts described rogue QEMU virtual machines delivered through Teams vishing, launched by scripting engines, with payloads stored in qcow2, ISO, and IMG images and parameters varied frequently to defeat signature detection. [2]

Two further cases anchor the pattern. CRON#TRAP, reported November 2024, delivered a phishing ZIP that unpacked to a QEMU binary renamed fontdiag.exe and a TinyCore pivot box running a Chisel tunnel over

WebSockets. [25] The 3AM ransomware intrusion in Q1 2025 used phone vishing spoofing an IT helpdesk plus email bombing, then Quick Assist, then a QEMU Windows 7 guest with a QDoor backdoor, dwelling nine days in front of an EDR-equipped victim. [26] The historical precedent runs deeper: Kaspersky's Carbanak reporting established in 2015 that remote control of the analyst's own machines, screen-video surveillance, and human-directed operations were sufficient for an attempted theft approaching one billion dollars across roughly 100 institutions, without a single novel stealth primitive. [27]

On the macOS side the documented record is younger but coherent. The offensive research titled Rootless, Native, Invisible demonstrates a complete macOS guest under Virtualization.framework running from a user-writable directory, with all attacker logic inside the guest, file-based command and control through Virtio-FS shared directories, and persistence through a user LaunchAgent labeled com.apple.virtualization.helper. [5] Patrick Wardle's VM escape research shows that macOS-on-macOS guests are not cleanly isolated: shared folders are implemented as APFS snapshots readable and writable from the guest, and clipboard, TCC, and network state are shared by design, which is a defense opportunity exactly as much as an isolation failure. [28] The timing of the related Citrix Bleed events is worth keeping straight: CVE-2023-4966 was disclosed October 10th, 2023, exploited as a zero-day since August 2023, and weaponized by LockBit 3.0 affiliates, while the QEMU-VM CitrixBleed2 campaign is a separate 2026 event on CVE-2025-5777. [29][30][3]

2. How Ghost Machines Arrive: Initial Access and Remote Management

Ghost machines do not self-install; every campaign begins with access, and the access record is dominated by exploited remote management software. The ScreenConnect chain is the cleanest example. On February 19th, 2024, ConnectWise disclosed CVE-2024-1709, an authentication bypass scored 10.0, paired with the CVE-2024-1708 ZipSlip path traversal. Huntress recreated a proof of concept the same day and coined the chain SlashAndGrab, observing more than 8,800 vulnerable servers exposed at disclosure and documenting immediate post-exploitation: privileged account creation, ransomware deployment, and ScreenConnect persistence. CISA added the vulnerability to the Known Exploited Vulnerabilities catalog on February 22nd, 2024. [31][32]

ScreenConnect is widely abused because a self-hosted instance lets the operator produce a differently-hashed deployment for every victim, defeating hash-based blocking. [4][31] The broader remote management abuse record is quantified by Huntress's 2025 Cyber Threat Report, which found more than 75% of remote-access incidents involved remote access trojans, with legitimate tools such as TeamViewer and LogMeIn weaponized for persistence and lateral movement. [33] CISA, NSA, and MS-ISAC documented in AA23-025A an October 2022 campaign in which phishing led victims to download legitimate ScreenConnect and AnyDesk portable executables used for refund scams, noting that RMM products generally do not trigger antivirus and portable installs bypass software controls. [34]

Microsoft's Storm-1811 reporting ties the delivery chain together: vishing plus email bombing, Quick Assist misuse, then scripted cURL downloads of batch and ZIP payloads leading to Qakbot, ScreenConnect, NetSupport Manager, Cobalt Strike, and Black Basta ransomware. Scripted download and execute is a post-access mechanic. [35] AhnLab documented the DPRK-aligned Kimsuky group installing Chrome Remote Desktop with a PIN for unattended control alongside AnyDesk and ngrok. [36] RustDesk is cataloged as abused when installed as a service for persistent re-entry, with known ports 21115 through 21119. [37] The supply-chain angle is real as well: AnyDesk disclosed on February 2nd, 2024, that attackers compromised its

production systems and stole code-signing certificates, forcing a full credential reset for a remote access vendor. [38] And the social engineering channel that installs all of this software is itself quantified in peer-reviewed literature on technical support scams, which depend on victims granting remote control. [39]

3. What the Macintosh Records: The Forensic Information Surface

macOS has answered the forensic question differently since Sierra: unified logging replaced the old text log files with a central binary store managed by logd, kept as compressed tracev3 files under /private/var/db/diagnostics with format string components in /private/var/db/uuidtext, and queryable through log show, log stream, and Console. [40][41][42][6] The store records login windows, process launches, network and proxy activity, subsystem events, and per-boot baselines. [4][6] An empirical APFS timestamp study already uses log show against the tracev3 store as a forensic source, and later macOS work tracks deleted-file spoliation through file metadata sources the same store records. [43][44]

3.1 The Unified Log and Its Limits

Two characteristics of the record bound any examination built on it. First, attrition: roughly 22% of entries live only in memory, and logd prunes the main persist store to a budget around 520 to 530 MB, so an always-on busy Mac retains on the order of twenty hours of routine activity, while fault and error entries survive longer. Second, redaction: message data is censored to the <private> placeholder, and a privileged privacy profile un-redacts only entries written after its installation, never retroactively. [12][13] sysdiagnose remains the one-command collection path: it packages a complete log archive together with hardware, network, crash, and process state into a tar bundle under /private/var/tmp that can be pulled remotely. [45][6]

3.2 Persistence: launchd and the Background Task Management Store

Persistence is the second pillar of a macOS compromise investigation, and launchd is its center of gravity. Job plists in ~/Library/LaunchAgents, /Library/LaunchAgents, /Library/LaunchDaemons, and searchable inside app bundles carry RunAtLoad, KeepAlive, and ProgramArguments; the Background Task Management store under /private/var/db/com.apple.backgroundtaskmanagement outlives the plist files that created the jobs, making BTM prime evidence after cleanup, readable with sfltool dumpbtm. Since macOS 13, SMAppService can register launch agents and daemons bundled inside an application, so the plist never appears in the classic directories. [7][46] The NukeSped and Esilet macOS backdoors anchored persistence in a user LaunchAgent using RunAtLoad and KeepAlive with an Apple-masquerading label, and Elastic's hidden plist rule targets exactly the dot-prefixed plist pattern that family still uses. [47][48]

3.3 Provenance and Permission Records: TCC and Quarantine

The TCC database registers privacy decisions in a system and a per-user SQLite store: Screen Recording, Accessibility, Input Monitoring, Full Disk Access, and protected folders are the services remote-control and virtualized tooling require, and denied rows are themselves evidence of attempted capability requests. Absence proves nothing, because tccutil reset removes rows and a root attacker can forge a TCC database outright via CVE-2023-40424. [8][49] Quarantine attributes record downloaded-file provenance in a per-user SQLite store whose entries survive deletion of the file, and joined with the com.apple.quarantine extended

attribute via UUID they reconstruct the distribution chain, with one forensic tell: curl, wget, scp, and git never set quarantine, so an unsigned tool with no quarantine record is the classic shape of an in-shell second stage. [9]

3.4 Remote Access and Virtualizer Traces

Remote access itself leaves concrete traces. Screen Sharing and Apple Remote Desktop write authentication events with viewer IP addresses into the unified log, keep state in com.apple.RemoteManagement preferences, and record per-application usage in the Remote Management caches; outbound control history lives in the Screen Sharing connections store. [10] Third-party tools keep findable logs: RustDesk maintains a rotating log set under ~/Library/Logs/RustDesk, AnyDesk writes ad.trace and connection traces, TeamViewer keeps incoming and outgoing connection logs, ScreenConnect stores sessions in SQLite, and Chrome Remote Desktop logs under its own directory. [50][51] And the virtualizer footprint is host-visible: a Virtualization.framework guest appears as the XPC service com.apple.Virtualization.VirtualMachine with ParavirtualizedGraphics lines in the unified log at startup, after which host logging of guest activity falls off; UTM keeps each VM as a self-contained bundle with config.plist and qcow2 drives, and raw QEMU leaves qemu-system processes with image files anywhere user-writable. [11][52][5]

4. Where Host Visibility Ends

The central limitation is documented in one sentence by UltraViolet Cyber's desktop hypervisor advisory: endpoint detection sees only the hypervisor service or virtualization process, not the internal workings of the guest where the attacker's implants operate. The same advisory observes that VM network traffic is often indistinguishable from ordinary host traffic and that a VM enclave survives reboots and even endpoint agent reinstallation. Hypervisors operate like regular applications with a minimal forensic footprint. [14] Microsoft's Defender Experts reach the same conclusion operationally, noting that activity inside the rogue VM is invisible to host EDR, which is precisely why the operators vary RAM size, network setup, and configuration-file usage to defeat signature detection of the launch parameters. [2]

What remains observable is the correlated set of host signals: virtualizer process presence and launch parameters including headless flags and network forwarding, disk image corpora with hostile extensions, launchd and BTM persistence writes, TCC grants to remote tools, unified log records of remote management and virtualizer startup, and network egress to RMM relays or tunnel endpoints. Vendor detections already encode pieces of this: Elastic ships a QEMU traffic tunneling rule keyed on -netdev hostfwd arguments, and Splunk publishes a suspicious QEMU execution rule mapping headless flags and image paths to ATT&CK techniques T1564.006 and T1036. [53][54] Memory remains the richest source: in one documented case the QEMU command line was recovered from host memory when no disk artifacts existed. [22] The network layer is frequently the only observer of the guest at all, and peer-reviewed network-behavior detection shows that reverse connections and external control patterns are distinguishable from flow characteristics alone. [16][55]

5. The Evidence Base: Peer-Reviewed Foundations

5.1 Log Forensics and the macOS Record

The academic foundation for log-driven investigation is well established. A comprehensive survey of operating system log forensics taxonomizes collection, analysis, and anomaly detection techniques and catalogs the public datasets that still underpin evaluation. [17] Applied to macOS, empirical work confirms the tracev3 store records system time changes and file operations retrievable with log show, and forensic research demonstrates that macOS file metadata sources can systematically track evidence spoliation when carving and recovery reach their limits. [43][44] Framework-based agentless acquisition, validated against agent-based platforms, shows reproducible, minimally intrusive remote acquisition of volatile memory and system artifacts is viable and compliance-aware. [56]

5.2 Evasion and Virtualization-Aware Detection

Evasion research quantifies the adversarial baseline a ghost machine exploits. A longitudinal study of 45,375 samples from 2010 through 2019 systematizes 92 evasive techniques and shows environment detection is a persistent and growing behavior; a complementary survey finds environment analysis is the single most common concealment strategy in the literature, at 48% of concealment-focused works. [57][58] Static detection of anti-VM techniques is feasible: a 2026 study consolidated 94 anti-analysis techniques into nine categories and evaluated 82 YARA rules across 459,508 malware samples, with strong precision for firmware and BIOS fingerprint checks. [59] EDR platforms themselves introduce telemetry surfaces that evasion attacks user-mode hooking and stealth injection patterns to defeat. [60]

From the defense side, the consistent scientific conclusion is that security must move outside the guest. A hypervisor-layer intrusion detection model fed time-series system call data detects malware inside virtual machines and outperforms in-guest alternatives. [61] Virtual machine introspection with memory forensics and machine learning at the hypervisor detects hidden, dead, and suspicious processes at 99.55% accuracy, and the VMGuard architecture demonstrates that tenant-VM security is unreliable exactly because malware can disable it; monitoring must run at the virtualization layer. [62][15]

5.3 macOS Malware, Persistence, and Detection

The macOS threat landscape is now systematically characterized: analysis of 57,636 macOS malware samples collected between 2006 and 2024 maps recurring TTPs onto the MITRE ATT&CK framework, with persistence and defense evasion the dominant recurring patterns. [20] Persistence point taxonomy on macOS has been published since at least 2014, with launchd agents and daemons at its core. [63] Memory-based detection of macOS userland malware is possible at scale with purpose-built Volatility plugins derived from real espionage and criminal campaigns [64], and kernel-level behavioral monitoring frameworks resistant to VM detection discovered previously unseen trojans in the APT32 OceanLotus ecosystem. [65] Domain-specific static features, including persistence techniques and entitlements, reach 98.5% detection on a 41,129-sample macOS corpus in the most recent ACM work, with persistence features among the highest-value signals. [66] Fileless and memory-only execution research rounds out the picture, documenting how execution without persistent files defeats signature inspection and why memory and configuration-state evidence become decisive. [67]

5.4 Remote Access and Network Behavior

Remote access is a network phenomenon before it is anything else, and peer-reviewed detection literature is consistent: reverse connections and external control are distinguishable from benign traffic using flow-level features. Four uncorrelated TCP-session behavioral features detect reverse RAT connections, including encrypted sessions, with AUC above 0.94; external control is detectable at network borders from packet direction and payload-size sequences with a false positive rate below 0.6%; encrypted RAT command and control traffic is modelable as behavior trees and remains robust against emerging benign applications; and hybrid host-plus-network feature engineering pushes RAT classification to roughly 98% accuracy. [16][55][68][69] These methods matter because the guest's outbound tunnel to a relay or C2 is exactly the reverse-connection pattern this literature measures.

5.5 Memory Forensics and the Virtualization Evidence Question

Volatile memory is the artifact of last resort and often the only one. A comprehensive survey documents the trade-offs between capture quality, overhead, and security and ranks memory examination as decisive for fileless and stealth malware. [18] On Intel Macintoshes, memory forensics under remote attack has been exercised end to end, with memory dumps analyzed through the Volatility framework recovering process, network, and injected-code artifacts, and with kernel access requirements and System Integrity Protection identified as the defining constraints. [70] On Apple silicon, foundational work compares Volatility behavior on M1 versus Intel and documents the plugin-level gaps introduced by the ARM transition. [71]

Virtualization sits on both sides of the forensic divide. It is an evidence container whose disk images and snapshots can be acquired and analyzed, and it is an anti-forensic instrument when an attacker deletes or resets the guest after use, an operation the forensic literature explicitly models. [19] Cloud anti-forensics research provisioned virtual machines with destructive and obfuscating techniques and demonstrated that many render recovery unlikely or impossible, which is precisely why host-side artifacts acquired before guest destruction are the reliable record. [72] For ghost machines on macOS, the guest is thereby best treated as a recoverable treasure in life and a destroyed record in death; the host artifacts cataloged in Section 3 are the evidence that survives both outcomes. [52][1]

6. Recommendations and Implications

1. Collect the record before the record rolls: The unified log keeps roughly a day of routine activity, and a sysdiagnose bundle can be pulled from a remote device in one command. When a Mac is suspected of remote compromise, the log archive, persistence stores, TCC and quarantine databases, and the virtualizer and disk image inventory should be preserved first, before any remediation touches the system. [45][12][7]

2. Treat every discovered virtual machine as an evidence object: Hash and acquire the guest's disk images read-only, mount them without modification, and enumerate guest-side persistence and tooling, including the AI agent skill files and command logs the attacker left in the guest workspace. The forensic literature on virtual machine evidence supports analyzing the VM itself, and UNC1945's guest carried the entire post-exploitation toolkit. [19][52][1]

3. Correlate the host record with the network record: The guest's outbound tunnel, its reverse SSH ports, its relay connections, and its unusual egress patterns are frequently the only signal that observes the guest at all, and flow-level RAT detection methods quantify how distinguishable those patterns are. [16][55][54]

4. Establish what is normal for each Mac before calling anything anomalous: Legitimate developer virtualization is pervasive, and the difference between a developer's UTM virtual machine and an attacker's ghost machine is baseline knowledge: who runs virtualization on this device, which remote management tools are approved, and what the user's machine did yesterday. [14][4]

5. Govern the instrumentation itself: Requiring endpoint monitoring inside approved virtual machines, or denying unauthorized virtualization outright, converts the concealment mechanism into a policy violation that hunt teams can action, and the same governance applies to remote management tools, whose legitimate presence is exactly what SlashAndGrab-era intrusions used to walk in. [4][14][34]

None of these measures is prospective. Every collection point, every signal, and every detection method cited in this paper exists today, documented and verified. The working knowledge an examiner needs to identify a Macintosh under continued remote control, including control concealed inside a ghost machine, is compiled in the preceding pages: the tradecraft, the record, and the analysis.

References

[1] Mandiant (J. Moore, W. Ledzion, et al.), "Live off the land? How about bringing your own island? An overview of UNC1945," *Google Cloud Blog / Mandiant,* https://cloud.google.com/blog/topics/threat-intelligence/live-off-the-land-an-overview-of-unc1945, November 2nd, 2020, Last accessed November 1st 2026

[2] Microsoft Defender Experts (A. Rajak and B. Venkatesh), "From social engineering to rogue VMs: The emerging tradecraft in human-directed ransomware attacks," *Microsoft Tech Community,* https://techcommunity.microsoft.com/blog/microsoft-defender-experts/from-social-engineering-to-rogue-vms-the-emerging-tradecraft-in-human-directed-r/4434219, July 18th, 2025, Last accessed November 1st 2026

[3] M. Demboski (Sophos), "QEMU abused to evade detection and enable ransomware delivery," *Sophos Blog,* https://www.sophos.com/en-us/blog/qemu-abused-to-evade-detection-and-enable-ransomware-delivery, April 16th, 2026, Last accessed November 1st 2026

[4] J. Stanton, "Ghost machines threats (lecture, Techno Security West 2026)," *My Private lecture material,* 2026, Last accessed November 1st 2026

[5] hxr1, "Rootless, native, invisible: Offensive operations via Apple's Virtualization Framework," *hxr1 research blog,* https://hxr1.ghost.io/rootless-native-invisible-offensive-operations-via-apples-virtualization-framework/, 2026, Last accessed November 1st 2026

[6] Mac Forensics, "Unified logs: macOS tracev3 system logging," *Mac Forensics (macforensics.app),* https://www.macforensics.app/en/artifacts/unified-logs, 2026, Last accessed November 1st 2026

[7] Mac Forensics, "Investigating launchd persistence on macOS," *Mac Forensics (macforensics.app),* https://www.macforensics.app/en/blog/launchd-persistence-investigation, September 28th, 2026, Last accessed November 1st 2026

[8] Mac Forensics, “TCC database forensics: macOS privacy permissions,” *Mac Forensics (macforensics.app),* https://www.macforensics.app/en/blog/tcc-database-forensics, September 28th, 2026, Last accessed November 1st 2026

[9] Mac Forensics, “Quarantine events and Gatekeeper: Tracing Mac downloads,” *Mac Forensics (macforensics.app),* https://www.macforensics.app/en/blog/quarantine-events-and-gatekeeper, September 28th, 2026, Last accessed November 1st 2026

[10] Mac Forensics, “Screen sharing and Apple Remote Desktop artifacts,” *Mac Forensics (macforensics.app),* https://www.macforensics.app/en/artifacts/screen-sharing-ard, 2026, Last accessed November 1st 2026

[11] H. Oakley, “What happens when you run a macOS VM on Apple silicon?,” *The Eclectic Light Company,* https://eclecticlight.co/2023/10/21/what-happens-when-you-run-a-macos-vm-on-apple-silicon/, October 21st, 2023, Last accessed November 1st 2026

[12] H. Oakley, “Inside the Unified Log 3: Log storage and attrition,” *The Eclectic Light Company,* https://eclecticlight.co/2025/09/29/inside-the-unified-log-3-log-storage-and-attrition/, September 29th, 2025, Last accessed November 1st 2026

[13] H. Oakley, “Inside the Unified Log 4: Log entries (redaction and privacy),” *The Eclectic Light Company,* https://eclecticlight.co/2025/10/06/inside-the-unified-log-4-log-entries/, October 6th, 2025, Last accessed November 1st 2026

[14] UltraViolet Cyber (TIDE), “Threat advisory: Desktop hypervisor threats,” *UltraViolet Cyber,* https://www.uvcyber.com/resources/threat-advisories/threat-advisory-desktop-hypervisor-threats, November 19th, 2025, Last accessed November 1st 2026

[15] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, “VMGuard: A VMI-based security architecture for intrusion detection in cloud environment,” *IEEE Transactions on Cloud Computing, vol. 8, no. 4, pp. 1270-1284, 2020,* https://consensus.app/papers/details/b63992087584573a886571b3b00b2b34/, 2020, Last accessed November 1st 2026

[16] H.-Y. Zhu, R.-L. Yang, and J.-Z. Wang, “A network behavior analysis method to detect reverse remote access Trojan,” *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS),* https://consensus.app/papers/details/98b0b9b29ae65e5287068455ef60a53a/, 2018, Last accessed November 1st 2026

[17] H. Studiawan, F. Sohel, and C. Payne, “A survey on forensic investigation of operating system logs,” *Digital Investigation, vol. 29, pp. 106-114, 2019,* https://consensus.app/papers/details/5c4e1695c64e5489866e4f733522f248/, 2019, Last accessed November 1st 2026

[18] H. Nyholm, R. Nemec Zlatolas, and L. Gartner, “The evolution of volatile memory forensics,” *Journal of Cybersecurity and Privacy, vol. 2, no. 3, pp. 538-560, 2022,* https://consensus.app/papers/details/4abed1773edb5b09915836c9fb5bfe1c/, 2022, Last accessed November 1st 2026

[19] H. Riaz, A. Iqbal, and W. Javed, “Analysis of VMware virtual machine in forensics and anti-forensics paradigm,” *2018 6th International Symposium on Digital Forensic and Security (ISDFS), IEEE,* https://consensus.app/papers/details/9d3f77954b9d5452b67e480731173024/, 2018, Last accessed November 1st 2026

[20] D. Miro, P. Valtchev, and J. R. C. Nurse, "Characterizing tactics, techniques, and procedures in the macOS threat landscape," *Computers & Security, 2025,* https://consensus.app/papers/details/2aebde7ae7e95cd897e9cd056c5db4f6/, 2025, Last accessed November 1st 2026

[21] QEMU Project, "About QEMU and System Emulation documentation," *QEMU Documentation,* https://www.qemu.org/docs/master/about/index, 2026, Last accessed November 1st 2026

[22] S. Chatzimangou (NVISO), "The detection and response chronicles: Covert operations through QEMU," *NVISO Blog,* https://blog.nviso.eu/2026/06/04/the-detection-response-chronicles-covert-operations-through-qemu/, June 4th, 2026, Last accessed November 1st 2026

[23] B. Toulas, "Payouts King ransomware uses QEMU VMs to bypass endpoint security," *BleepingComputer,* https://www.bleepingcomputer.com/news/security/payouts-king-ransomware-uses-qemu-vms-to-bypass-endpoint-security/, April 17th, 2026, Last accessed November 1st 2026

[24] V. Vrabie (Bitdefender), "Curly COMrades: Evasion and persistence via hidden Hyper-V virtual machines," *Bitdefender Business Insights Blog,* https://www.bitdefender.com/en-us/blog/businessinsights/curly-comrades-evasion-persistence-hidden-hyper-v-virtual-machines, November 4th, 2025, Last accessed November 1st 2026

[25] B. Toulas, "Windows infected with backdoored Linux VMs in new phishing attacks (CRON#TRAP)," *BleepingComputer,* https://www.bleepingcomputer.com/news/security/windows-infected-with-backdoored-linux-vms-in-new-phishing-attacks/, November 4th, 2024, Last accessed November 1st 2026

[26] S. Gallagher, R. Weiland, and C. Cowie (Sophos), "A familiar playbook with a twist: 3AM ransomware actors dropped virtual machine with vishing and Quick Assist," *Sophos Blog,* https://www.sophos.com/en-us/blog/a-familiar-playbook-with-a-twist-3am-ransomware-actors-dropped-virtual-machine-with-vishing-and-quick-assist, May 20th, 2025, Last accessed November 1st 2026

[27] Kaspersky Lab, "Carbanak APT: The great bank robbery (v2.1)," *Kaspersky Lab,* https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/08064518/Carbanak_APT_eng.pdf, February 2015, Last accessed November 1st 2026

[28] S. Gogola (Gogola Nexus), "Apple Silicon VMs not safe for macOS malware (Patrick Wardle VM-escape research)," *Gogola Nexus,* https://sebastiangogola.me/apple-silicon-vms-not-safe-for-macos-malware/, 2025, Last accessed November 1st 2026

[29] Cybersecurity and Infrastructure Security Agency (CISA), "Guidance for addressing Citrix NetScaler ADC and Gateway vulnerability CVE-2023-4966, Citrix Bleed," *CISA,* https://www.cisa.gov/guidance-addressing-citrix-netscaler-adc-and-gateway-vulnerability-cve-2023-4966-citrix-bleed, October 2023, Last accessed November 1st 2026

[30] CISA, FBI, MS-ISAC, ASD ACSC, "StopRansomware: LockBit 3.0 ransomware affiliates exploit CVE-2023-4966 Citrix Bleed vulnerability (AA23-325A)," *CISA Cybersecurity Advisory AA23-325A,* https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-325a, November 21st, 2023, Last accessed November 1st 2026

[31] Huntress, "A catastrophe for control: Understanding the ScreenConnect authentication bypass (CVE-2024-1709 and CVE-2024-1708)," *Huntress Blog,* https://www.huntress.com/blog/a-catastrophe-for-control-

understanding-the-screenconnect-authentication-bypass, February 19th, 2024, Last accessed November 1st 2026

[32] CISA, "CISA adds one known exploited ConnectWise vulnerability, CVE-2024-1709, to catalog," *CISA Alert,* https://www.cisa.gov/news-events/alerts/2024/02/22/cisa-adds-one-known-exploited-connectwise-vulnerability-cve-2024-1709-catalog, February 22nd, 2024, Last accessed November 1st 2026

[33] Team Huntress, "Huntress 2025 cyber threat report: Proliferating RATs, evolving ransomware, and other findings," *Huntress,* https://www.huntress.com/blog/huntress-2025-cyber-threat-report-proliferating-rats-evolving-ransomware-and-other-findings, February 11th, 2025, Last accessed November 1st 2026

[34] CISA, NSA, and MS-ISAC, "Protecting against malicious use of remote monitoring and management software (AA23-025A)," *CISA Cybersecurity Advisory AA23-025A,* https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-025a, January 25th, 2023, Last accessed November 1st 2026

[35] Microsoft Threat Intelligence, "Threat actors misusing Quick Assist in social engineering attacks leading to ransomware (Storm-1811)," *Microsoft Security Blog,* https://www.microsoft.com/en-us/security/blog/2024/05/15/threat-actors-misusing-quick-assist-in-social-engineering-attacks-leading-to-ransomware/, May 15th, 2024, Last accessed November 1st 2026

[36] AhnLab ASEC, "Kimsuky threat group using Chrome Remote Desktop," *AhnLab Security Emergency Response Center,* https://asec.ahnlab.com/en/55145/, June 2023, Last accessed November 1st 2026

[37] LOLRMM (magicsword-io), "RustDesk: Security risks, IOCs and detection," *LOLRMM Catalog,* https://lolrmm.io/tools/rustdesk/, 2026, Last accessed November 1st 2026

[38] Akamai, "The AnyDesk breach: Overview and recommendations," *Akamai Blog,* https://www.akamai.com/blog/security-research/anydesk-breach-what-to-know-mitigations-and-recommendations, February 2024, Last accessed November 1st 2026

[39] N. Miramirkhani, O. Starov, and N. Nikiforakis, "Dial one for scam: A large-scale analysis of technical support scams," *Network and Distributed System Security Symposium (NDSS) 2017,* https://consensus.app/papers/details/9b4486ffdbbc5f9cb5d593b752b4ca5e/, 2016, Last accessed November 1st 2026

[40] Apple Inc., "Logging (unified logging system documentation)," *Apple Developer Documentation,* https://developer.apple.com/documentation/os/logging, 2026, Last accessed November 1st 2026

[41] Apple Inc., "Unified logging and activity tracing (WWDC 2016 Session 721)," *Apple WWDC 2016,* https://devstreaming-cdn.apple.com/videos/wwdc/2016/721wh2etddp4ghxhpcg/721/721_unified_logging_and_activity_tracing.pdf, 2016, Last accessed November 1st 2026

[42] S. Edwards, "New macOS Sierra (10.12) forensic artifacts: Introducing unified logging," *mac4n6.com,* http://www.mac4n6.com/blog/2016/11/13/new-macos-sierra-1012-forensic-artifacts-introducing-unified-logging, November 13th, 2016, Last accessed November 1st 2026

[43] S. H. Kim, D. Kim, and S. Lee, "A study on the APFS timestamps in macOS," *International Journal of Engineering & Technology, vol. 7, no. 2.33, 2018,* https://consensus.app/papers/details/e9ff40f35ba75eb397873a53e0ba0e9f/, 2018, Last accessed November 1st 2026

[44] J. Joun, J. Kim, S. Lee, and S. Lee, “Discovering spoliation of evidence through identifying traces on deleted files in macOS,” *Forensic Science International: Digital Investigation, 2023,* https://consensus.app/papers/details/8d9b56ad08af5a7b9c4003452b0df462/, 2023, Last accessed November 1st 2026

[45] Apple Inc., “Use diagnostics to research device issues (sysdiagnose),” *Apple Platform Support Guide,* https://support.apple.com/guide/platform-support/supd3f43814e/26/web/26, 2026, Last accessed November 1st 2026

[46] C. Fitzl (theevilbit), “macOS Service Management: The SMAppService API, quick notes,” *theevilbit blog,* https://theevilbit.github.io/posts/smappservice/, September 28th, 2023, Last accessed November 1st 2026

[47] P. Wardle, “From the DPRK with love (NukeSped/Esilet launch-agent persistence analysis),” *Objective-See Blog,* https://objective-see.org/blog/blog_0x6E.html, May 9th, 2022, Last accessed November 1st 2026

[48] Elastic Security, “Persistence via a hidden plist filename (prebuilt detection rule),” *Elastic Security Documentation,* https://www.elastic.co/docs/reference/security/prebuilt-rules/rules/macos/persistence_hidden_plist_filename, 2026, Last accessed November 1st 2026

[49] C. Fitzl (theevilbit), “CVE-2023-40424: How malware can bypass Transparency, Consent, and Control,” *theevilbit blog,* https://theevilbit.github.io/posts/cve-2023-40424/, May 24th, 2024, Last accessed November 1st 2026

[50] RustDesk Project, “FAQ (access log paths and rotation),” *rustdesk/rustdesk Wiki, GitHub,* https://github.com/rustdesk/rustdesk/wiki/FAQ, 2026, Last accessed November 1st 2026

[51] m31r0n (D0V0), “RMM-Forensic-Analyzer (RMM artifact path catalog),” *GitHub,* https://github.com/m31r0n/RMM-Forensic-Analyzer, 2026, Last accessed November 1st 2026

[52] UTM Project, “Drive: UTM documentation (utm bundle and QCOW2 storage),” *UTM Documentation,* https://docs.getutm.app/settings-qemu/drive/drive/, 2026, Last accessed November 1st 2026

[53] R. Tait (Splunk), “Detection: Windows suspicious QEMU execution,” *Splunk Security Research,* https://research.splunk.com/endpoint/c4c0a2d4-0675-4fde-97b7-115145ba257c/, May 13th, 2026, Last accessed November 1st 2026

[54] Elastic Security, “Potential traffic tunneling using QEMU (prebuilt rule),” *Elastic Security Documentation,* https://www.elastic.co/guide/en/security/current/prebuilt-rule-8-19-34-potential-traffic-tunneling-using-qemu.html, 2026, Last accessed November 1st 2026

[55] S. Wu et al., “Detecting remote access Trojans through external control at area network borders,” *2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS),* https://consensus.app/papers/details/5aa4b4a9c57559e4acc78b4a578b3ecc/, 2017, Last accessed November 1st 2026

[56] D. García-Fernández, L. Tobarra, A. Robles-Gómez, and R. Pastor Vargas, “Agentless and automated acquisition of digital evidence in corporate and industrial networks: Architecture, validation, and compliance,” *Electronics, vol. 15, no. 4, 2026,* https://consensus.app/papers/details/1c346268713157e69c42d9c9c72ff6bb/, 2026, Last accessed November 1st 2026

[57] N. Galloro, M. Polino, M. Carminati, A. Continella, and S. Zanero, “A systematical and longitudinal study of evasive behaviors in Windows malware,” *Computers & Security, vol. 108, p. 102550, 2021,*

https://consensus.app/papers/details/60257734a3855974bc64807ee8bd4934/, 2021, Last accessed November 1st 2026

[58] J.-X. Geng et al., "A survey of strategy-driven evasion methods for PE malware: Transformation, concealment, and attack," *Computers & Security, vol. 133, p. 103595, 2023,* https://consensus.app/papers/details/6b844ecdef485c628d1d3a4539a91cbc/, 2023, Last accessed November 1st 2026

[59] S. Kanj, A. Dabbous, and R. State, "Automating the detection of evasive Windows malware: An evaluated YARA rule library for anti-VM and anti-sandbox techniques," *Journal of Cybersecurity and Privacy, 2026,* https://consensus.app/papers/details/0308320369a15aa6a2184c64aaf501b7/, 2026, Last accessed November 1st 2026

[60] E. Negoescu et al., "Endpoint security evasion: A survey of detection mechanisms and injection techniques," *2026 16th International Conference on Communications (COMM), IEEE,* https://consensus.app/papers/details/6c2e980a0c075146b5c4c8e5d1007686/, 2026, Last accessed November 1st 2026

[61] A. A. R. Melvin et al., "A deep learning model leveraging time-series system call data to detect malware attacks in virtual machines," *International Journal of Computational Intelligence Systems, 2025,* https://consensus.app/papers/details/872cb0f9b0975538b1d9e93a9cb5dfe6/, 2025, Last accessed November 1st 2026

[62] C. Jaidhar, "Leveraging virtual machine introspection with memory forensics to detect and characterize unknown malware using machine learning techniques at hypervisor," *Digital Investigation, vol. 21, pp. 99-112, 2017,* https://consensus.app/papers/details/841824b62ee4529e9ab05581bdd40e76/, 2017, Last accessed November 1st 2026

[63] P. Wardle, "Methods of malware persistence on Mac OS X," *Virus Bulletin Conference 2014,* https://consensus.app/papers/details/38a365f669415cb588e51726bdf5c7fb/, 2014, Last accessed November 1st 2026

[64] M. Manna, A. Case, and G. G. Richard III, "Modern macOS userland runtime analysis," *Digital Investigation, vol. 38, p. 301221, 2021,* https://consensus.app/papers/details/5367d2fdfcce5cbdb1aeea6d3f023e59/, 2021, Last accessed November 1st 2026

[65] D.-P. Pham et al., "Mac-A-Mal: macOS malware analysis framework resistant to anti evasion techniques," *Journal of Computer Virology and Hacking Techniques, vol. 15, pp. 249-257, 2019,* https://consensus.app/papers/details/f8b0ac3f0ddd5d87862afabc3de6bbd3/, 2019, Last accessed November 1st 2026

[66] B. Montaruli et al., "The role of domain-specific features in malware detection: A macOS case study," *Proceedings of the ACM Asia Conference on Computer and Communications Security (ASIACCS), 2026,* https://consensus.app/papers/details/cf9a9d7017f05c39975ad67e08d2e8b5/, 2026, Last accessed November 1st 2026

[67] Sudhakar and S. Kumar, "An emerging threat Fileless malware: A survey and research challenges," *Cybersecurity, vol. 3, no. 1, 2020,* https://consensus.app/papers/details/5656c3cf400850f1b89b898c3092857a/, 2020, Last accessed November 1st 2026

[68] C. Dong, P. Wang, J. Chen, and X. Zhang, "MBTree: Detecting encryption RATs communication using malicious behavior tree," *IEEE Transactions on Information Forensics and Security, vol. 16, pp. 4291-4304, 2021,* https://consensus.app/papers/details/772065ddf4445122a8c02ac9c4bda4eb/, 2021, Last accessed November 1st 2026

[69] A. M. Aburbeian et al., "Improving remote access Trojans detection: A comprehensive approach using machine learning and hybrid feature engineering," *AI, vol. 6, no. 3, 2025,* https://consensus.app/papers/details/c4a2f431e4475046ad8a97f366d5254c/, 2025, Last accessed November 1st 2026

[70] F. Alif et al., "Intel-based Macintosh memory forensics: Eavesdropping attack analysis through memory dump," *2025 International Conference on Data Science and Its Applications (ICoDSA), IEEE,* https://consensus.app/papers/details/6621cabd28b2540fa3d21d72018631f0/, 2025, Last accessed November 1st 2026

[71] J. Duke, "Memory forensics comparison of Apple M1 and Intel architecture using Volatility Framework," *Masters thesis, Louisiana State University, 2021,* https://consensus.app/papers/details/527dabf044ff56ff9e2c99e24bc06a93/, 2021, Last accessed November 1st 2026

[72] C. Lowetz, G. Grispos, and K. Mahar, "Anti-forensics under scrutiny: Assessing the effectiveness of digital obfuscation in the cloud," *2024 12th International Symposium on Digital Forensics and Security (ISDFS), IEEE,* https://consensus.app/papers/details/f8df5f3fe5bc5cffbd5382e34f3a082b/, 2024, Last accessed November 1st 2026