CS-021 · Malware

Fileless Malware

Attacks that run in memory and abuse trusted built-in tools, leaving little on disk to find.

Executive Summary

Fileless malware carries out an attack without relying on a traditional malicious file on disk. Instead it runs in memory and abuses legitimate, already-installed system tools to do its work, a tactic often called living off the land. Because there is little or no file for signature-based tools to scan, and because the activity looks like normal administration, fileless attacks are stealthy and hard to detect.

What It Is

Fileless malware is a class of attack that minimizes what it writes to disk, running its malicious logic in a computer's memory and through trusted programs that come with the operating system. Rather than dropping an executable that antivirus can flag, it triggers built-in scripting and administration tools to load and run code directly. These trusted utilities are often called living-off-the-land binaries, and the technique is called living off the land because the attacker uses the resources already present rather than bringing their own tools. Fileless activity is not always completely file-free. Attackers may still store scripts or configuration in unusual places such as the system registry or scheduled tasks to survive a reboot. The defining trait is the reliance on memory and legitimate tools to blend in and avoid leaving obvious artifacts.

Why It Matters

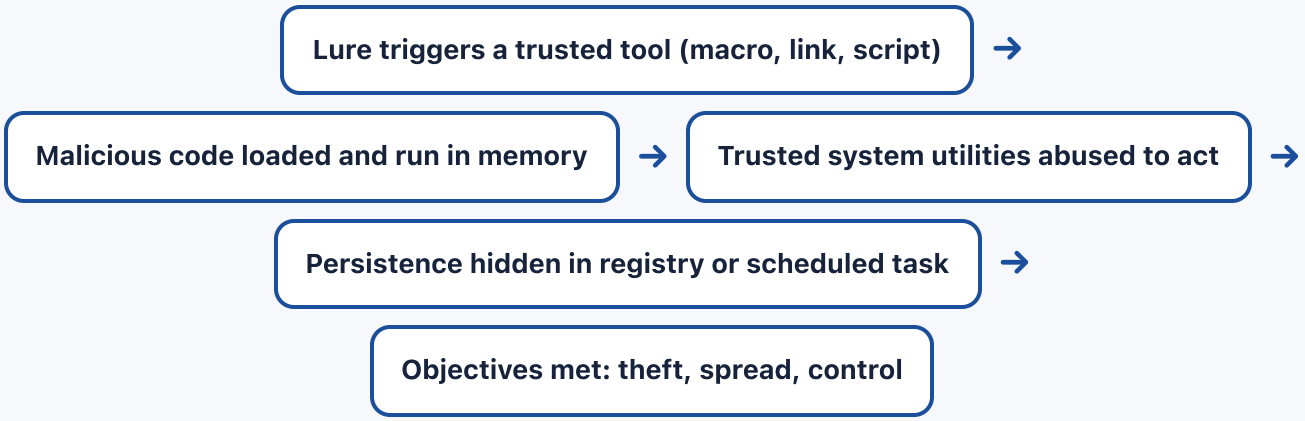
Fileless attacks defeat the assumption at the heart of older defenses, that malware is a file you can scan and quarantine. When the malicious behavior runs through tools an organization trusts and uses every day, simple signature detection sees nothing wrong. This makes fileless techniques popular in targeted intrusions and hands-on attacks where stealth and dwell time

matter. For defenders, it raises the bar: you must watch how trusted tools are being used, not just what files exist. For security professionals, understanding these techniques is essential because they appear in many modern breaches, from initial phishing payloads to lateral movement and persistence deep inside a network.

How It Works

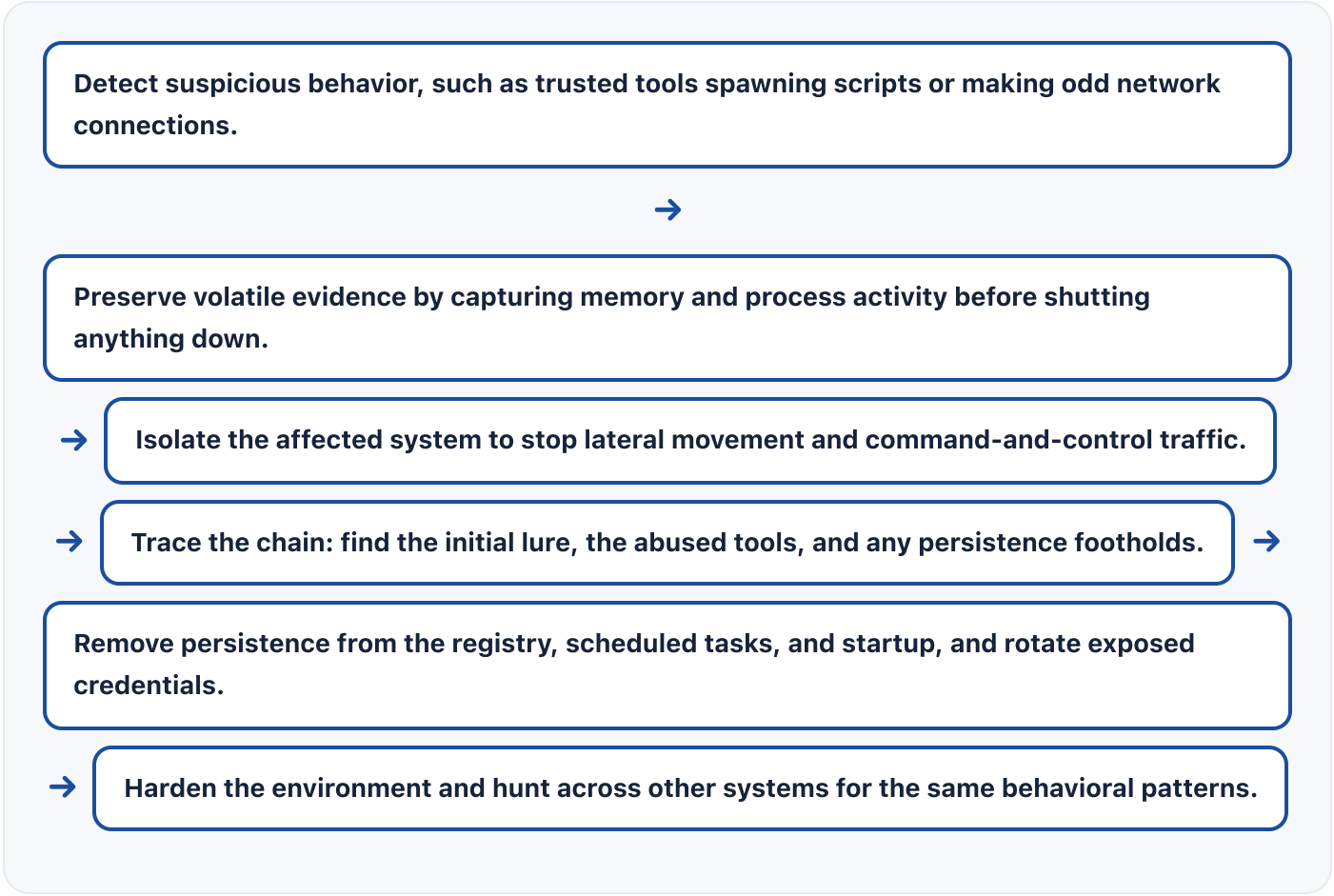
A fileless attack often begins with a lure, such as a phishing document with a malicious macro or a link that triggers a script. That initial action does not save a program; it launches a legitimate interpreter or administration tool and feeds it commands. The code runs in memory, where it can steal credentials, move to other systems, and reach out to attacker-controlled servers. To survive a restart, the attacker may plant a small trigger in the registry, a scheduled task, or a startup entry that relaunches the in-memory payload. Because each step uses tools already trusted on the system, the activity hides inside normal operations. Detecting it depends on spotting suspicious patterns, such as an office document spawning a scripting engine, an administration tool making unusual network connections, or unexpected commands passed to a system interpreter.

Architecture Diagram



A fileless attack triggers a trusted tool to run malicious code in memory, then uses a small foothold to persist.

Visual Workflow



Common Attacks

- Phishing documents with malicious macros that launch built-in scripting tools
- Abuse of legitimate administration and scripting utilities (living off the land)
- In-memory execution of code that never touches disk
- Persistence hidden in the registry, scheduled tasks, or startup entries
- Credential theft and lateral movement using trusted native tools

Common Mistakes

- Relying only on signature-based antivirus that looks for malicious files
- Not collecting or reviewing detailed process and command-line logs
- Leaving powerful scripting and admin tools unrestricted for all users
- Ignoring registry and scheduled-task changes as persistence hiding places
- Shutting down a machine before capturing memory, destroying key evidence

Best Practices

- Deploy endpoint detection and response that analyzes behavior, not just files
- Enable and centrally collect detailed process, command-line, and script logging
- Restrict or monitor scripting and administration tools with least privilege
- Disable or control macros from untrusted sources by default
- Use application control to limit how trusted tools can be launched and used
- Baseline normal tool usage and alert on unusual parent-child process chains

Quick Checklist

- [x] EDR deployed with behavioral detection tuned for living-off-the-land activity
- [x] Command-line and script logging enabled and centralized
- [x] Macros from the internet disabled or tightly controlled
- [x] Scripting and admin tools restricted to those who need them
- [x] Registry, scheduled-task, and startup changes monitored for persistence
- [x] Incident process captures memory before powering off a suspect system

Recommended Tools

Endpoint Detection and Response (EDR)
Detects malicious behavior and abuse of trusted tools in memory

SIEM with process and script logging
Correlates command-line and script activity to reveal fileless chains

Memory forensics tooling
Analyzes RAM to recover in-memory code and attacker activity

Application control
Limits how and when trusted system tools can execute

Industry Standards

MITRE ATT&CK
Documents living-off-the-land and in-memory techniques and defenses

NIST SP 800-61
Incident handling guidance applicable to stealthy intrusions

CIS Critical Security Controls
Logging and application-control safeguards that surface fileless activity

Career Relevance

Fileless malware is core knowledge for incident responders reconstructing stealthy intrusions, SOC analysts hunting anomalous tool usage, and security engineers tuning behavioral detection and logging. Malware analysts study these techniques to build detections, and threat hunters chase them across environments, all part of the audience AI-Governance-Jobs.com serves.

Interview Questions

- What makes malware fileless, and why is it harder to detect than a traditional file-based threat?
- What does living off the land mean, and why do attackers abuse built-in tools?
- If fileless malware runs in memory, how do attackers achieve persistence?
- What logging and detection would you prioritize to catch fileless activity?
- Why should you capture memory before powering off a suspected fileless infection?

Related Certifications

CompTIA Security+ | GIAC GREM | ISC2 Certified in Cybersecurity (CC)

Further Reading

- MITRE ATT&CK
- CISA: Cybersecurity Best Practices
- NIST Cybersecurity Framework

Key Takeaways

- Fileless malware runs in memory and abuses trusted built-in tools instead of dropping a file.
- Living off the land lets attackers blend into normal administration and evade signature scanning.
- It is often not fully file-free, since persistence may hide in the registry or scheduled tasks.
- Detection depends on behavioral analysis and detailed process and command-line logging.
- Restricting powerful tools, controlling macros, and capturing memory are key defenses.

FAQ

▶ **If it is fileless, is there ever anything on disk?**

▶ **Can traditional antivirus stop fileless malware?**

▶ **Why is capturing memory so important in these cases?**

Related Careers

RELATED ROLES

Incident Responder | Soc Analyst | Malware Analyst

RELATED CERTIFICATIONS

CompTIA Security+ | GIAC GREM | ISC2 Certified in Cybersecurity (CC)

CURRENT OPENINGS

Live roles are on the job board.

GUIDES & SALARY

Browse all jobs

Career guides | Role roadmaps | Salary data | Career tools

SUGGESTED LEARNING PATH

1. Ground the basics with CS-001 Cybersecurity
2. Study this sheet: **Fileless Malware**
3. Go deeper: Cybersecurity
4. Go deeper: Computer Viruses
5. Validate it: work toward **CompTIA Security+**
6. Find the role: browse current openings

RELATED SHEETS

CS-001 Cybersecurity | CS-011 Computer Viruses | CS-017 Rootkits | CS-019 Cryptominers | CS-032 Phishing