International Journal of Multidisciplinary Research and Growth Evaluation.

Fighting Living off the Land Attacks – Where they are not

Anand Athavale
Independent Researcher, Decades of Industry experience in Data Management, USA

* Corresponding Author: **Anand Athavale**

Article Info

ISSN (Online): 2582-7138
Impact Factor (RSIF): 7.98
Volume: 06
Issue: 04
July - August 2025
Received: 22-06-2025
Accepted: 23-07-2025
Published: 26-08-2025
Page No: 1437-1440

Abstract
This article explores an approach for security analysts to reduce the burden of looking for LOLBINs. Given attackers these days do not place malware in the target environments and instead orchestrate attacks living off the land, the detection of these is a massive undertaking for security analysts. Since the LOLBINs are common day and trusted binaries, the attacker tools are basically inseparable from legitimate daily use tools. Even if there is a known list of LOLBINs, keeping an eye on every use of these for legitimate vs. suspicious use is a daunting task. The devices to look for such behaviors needs to be narrowed down. A different approach for the process of elimination can help identify a narrowed list of devices to focus on when looking for suspicious behaviors of LOLBINs. At minimum, prioritizing the list of devices to go after, with a "red team" mentality can help.

DOI: https://doi.org/10.54660/.IJMRGE.2025.6.4.1437-1440

Keywords: Leaving Off the Land, LOLBINs, Threat Hunt, CVE, KVE, Blue Team/Red Team

1. Introduction

The term "Living off the land" means surviving using resources already in the environment. The term LOLBIN was coined by the security researchers who often played the role of an attacker, often referred to as "Red Team" members. The attackers have pivoted to this method because the use of existing tools allows them to download payload, execute code, move laterally, dump credentials while making the detection extremely hard if not impossible. While there are many tactical rules and tools to get alerts for LOLBINs, they are prone to false positives. In large environments, collection of telemetry is challenged and not guaranteed. This is why it needs to be approached from another angle besides brute force. This paper tries to give a different perspective on where to start.

LOLBINs and use of LOLBINs by attackers

LOLBINs are executable binaries where each one is shipped for a specific purpose. These are the choice of attackers because most of these are trusted binaries pre-installed by the operating systems and historically whitelisted by security tools. As an example, consider certutil.exe. It is trusted pre-installed binary whitelisted and trusted by security tools and used by sysadmins frequently for routine tasks like viewing and installing certificates, checking certificate revocations and online status and similar tasks [1]. While it is meant for such tasks, attackers use it because it can download files and decode them. Another example is powershell.exe. While the purpose is Command-line shell and scripting for task automation and configuration management, it can be also used for remote management and Active Directory Management. Attackers use powershell.exe because it can download and execute payloads directly in memory (fileless attacks) bypassing writing to disk thus avoiding detection. These are the typical characteristics of any LOLBINs [2]. These binaries are trusted and can be used for unintended purposes while avoiding detection because those are frequently used for legitimate purposes.

To give an analogy, a car without a license plate will definitely raise suspicion and has a high chance of getting caught when used for committing a crime or theft. But a registered ambulance will be difficult to spot and differentiate whether it is being

used to transport a patient, or a fugitive. This is why LOLBINs are one of the popular tools for attackers and one of the toughest challenges for security teams to track and catch when used for malicious purposes.

One important fact should be noted here. The examples discussed may point to Windows binaries only. But LOLBINs are not limited to Windows alone. Linux built in binaries such as bash, find, awk, curl, scp can be similarly abused to bypass restrictions, escalate privileges and movement of data [3]. The bad news is that even clouds are not immune to this issue. As an example, Azure has various custom extensions to serve thousands of administrators to orchestrate their Azure fleet. These Azure built-in capabilities can be abused to bypass any network defense lines which could be looked at as another type of LoLBins [4].

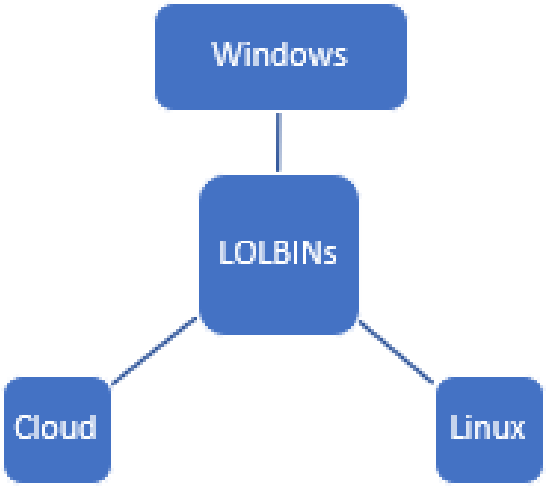

Fig 1: Possible Device Types/Locations for LOLBINs

Use of LOLBIN however requires an initial compromise using human compromise methods like vishing, password leak or, use of CVE exploit to gain initial access. LOLBINs themselves are not vulnerabilities allowing compromise. This is important to note because LOLBINs should not be approached or looked at from an initial compromise perspective. It is not wise to only focus on internet facing devices when looking for illegitimate LOLBIN activities. Where LOLBINs are present, is not something that indicates a possible attack or, compromise. The way LOLBINs get used is what determines the probability of an attack.

Existing methods and tools for LOLBINs unintended use detection

The list of steps to address this problem starts with knowing a list of such LOLBINs. Projects like LOLBAS give a static, community-driven reference catalog of LOLBINs, scripts and libraries. It documents the capabilities of LOLBINs in terms of behaviors, mappings to TTPs and notes on detection. This constitutes a list of what could be abused and how. Alternately, or additionally, feeds like Google Threat Intelligence bring real-world usage of LOLBINs adding context with threat actor behavior, recent campaigns, and indicators of compromise. This brings more up-to-date information on which LOLBIN is being wrongly used by specific ransomware families. Of course, the detection mechanisms still need to be built with such rich intelligence.

There are EDR/XDR tools which help monitoring and flagging illegitimate use of LOLBINs. Some track processes (including LOLBINs), aggregates CPU, memory, disk, and network usage across endpoints [5]. Built-in queries and dashboards can show CPU usage per binary over time across the fleet. Other track binaries like powershell.exe, rundll32.exe, etc., with telemetry on resource usage and execution context which are quarriable for hunting purposes. Other than detection, some experts suggest simply minimizing the administrative work to reduce the use of legitimate use itself. This can make detection easier since the use of LOLBINs itself will become rare.

Challenges with existing tools and methods

Monitoring arguments of LOLBIN arguments is easier said than done. The challenge of capturing the arguments lies in the volume.

On a single light office worker device, the number of spawned processes per day is somewhere between 5,000 to 10,000. On a busy user device, it can get higher with 15,000 to 25,000 processes being spawned. Concurrent process count can vary from 100 to 400 at a given moment. This gets worse when a shared Virtual Desktop Infrastructure device with more than a handful of users shares the host. Here, the process count can touch a million. The corresponding log size gets into MBs to GBs per day for a single device. For a large enterprise, this gets multiplied by 5,000 to 15,000 and starts becoming a daunting task for fast network transfers and storage, where this load competes with regular business applications load on the infrastructure.

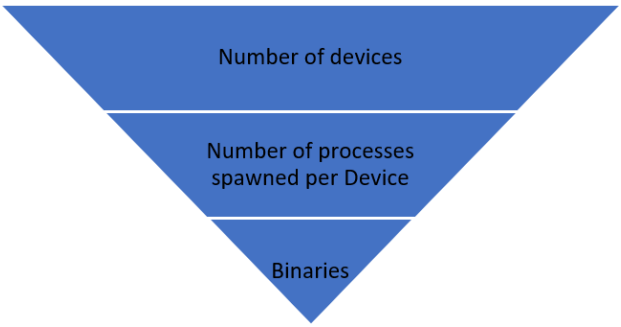

Fig 2: Volume explosion for Detection

This is the base process spawn load. Detection requires arguments and other context to be captured. No argument event size for a single process spawn event is roughly 500 to 800 bytes. Adding arguments capture increases that by 50 to 200% and gets into 2K to 4KB occasionally reaching 8 KB for LOLBINs. The raw event monitoring can be narrowed with the help of threat feeds and static list. But this only helps to a certain extent. LOLBIN binaries (like powershell.exe, cmd.exe, certutil.exe) are used legitimately thousands of times a day in many orgs. For example, in a fleet of 5,000 devices, powershell.exe alone can generate 50 to 100 K spawning events.

While volume is one of the challenges, there are other challenges with this method. Processes can be short lived and too quick for an agent's user mode hook or kernel callback to record it. Processes launched from kernel drivers, processes started in suspended mode with code injection replacing arguments and inherited handle by child process prevents argument capture. There are other issues such as containerized execution and use of nested VMs makes detection difficult by traditional monitoring agents. Even after capturing, the arguments can drop during compression or fail to reach central EDR due to network drops. Attackers also make copies of LOLBINs to escape this type of detection. While some tools are advanced to detect such copies based on context, argument capture and monitoring of LOLBINs is anything but clean and guaranteed.

A complementary approach to look where attackers would

Threat intelligence feed-based narrowing of LOLBIN to monitor is a good start. It can be augmented however with many more prioritizations based on environmental awareness. Here are examples of such environmental considerations which can be used to prioritize and focus. Some of the prioritization is from a defender's perspective and some of this prioritization stems from thinking like an attacker, or, "Red team" mentality.

Monitoring of critical assets and devices:

Tier-0, Tier-1 assets like domain controllers, PKI Infrastructures, backup servers along with critical finance, health or similar regulated industry application hosts are the ones which constitute these.

User Privilege level and Account type:

Here the devices are chosen by not what is installed on the devices, but instead who is using those devices. Typical examples are devices used by Domain admins, local admin users, and those used by DevOps and developers because the LOLBIN usage is heavily skewed on these. The rationale is, attackers may get caught easily if LOLBIN usage is found on less probable devices like general office workers.

Heavy service account usage devices and automation hosts:

The devices where service accounts are used where the passwords are not updated frequently are candidates for this type along with hosts which are used for running homegrown scripts and scheduled tasks.

RED Team mentality-based device identification:

Here, instead of thinking only from a defender angle where focus is primarily on crown jewels and high privilege activity, the tactics change to thinking like an attacker. Hence, this category for example, constitutes devices where a known CVE/KVE was found in the recent past, irrespective of whether CVE/KVE was patched. Here, the underlying reason is like physical security breach. Just because the door was closed, does not mean the intruder is no longer present. Monitoring the premise for any intruder activity is essential. Along the same lines, even if the CVE/KVE was addressed, it is important to monitor for any LOLBIN activity on such devices. Devices, hosts, servers which are not patched recently are another set to look at. Many times, applications do not get upgraded to latest versions, or, for some reason, may be running end of support versions. These types of application hosts make such a group for LOLBIN monitoring.

Historical behavior with smart grouping:

Often, some tools employ AI/ML to look at current activity to historical activity. However, storing history for even 30 days gets costly and complex. To address that, here is a variation in behavior monitoring to consider. Keep the historical data only for 7 days but instead of comparing each device with its own history, consider creating groups of devices based on above mentioned categories and then track if any device is behaving differently than its peers within the group.

Approaching the LOLBIN detection from this angle in conjunction with existing tools and techniques can help solve the volume and other detection challenges and can increase the chances of catching illegitimate use of LOLBINs quicker.

Conclusion

Living off the land attacks are complex to detect and harder to predict because the tools used for these attacks, referred to as LOLBINs, hide in plain sight. The use of existing high usage tools for illegitimate tools comes very close to taking advantage of crowded places to avoid surveillance. Monitoring the actions of each LOLBIN where the binary appears harmless becomes tricky. Trimming down the list to known LOLBIN use by attackers through feeds and projects helps somewhat. But it is still a challenge given high volume

use of LOLBINs, the pipeline issues of capture and propagation of such events and then segregation of good and suspicious behavior. However, complementing it through prioritization of devices through various other criterions including the one which takes a “Read team” approach can help improve efficacy and speed of detection of LOLBIN illegitimate use.

References

1. Microsoft Learn. Windows Commands: certutil [Internet]. 2025 May [cited 2025 Aug]. Available from: https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/certutil
2. SentinelOne. What are LOLBins? How attackers use LOLBins in fileless attacks [Internet]. 2020 Jul [cited 2025 Aug]. Available from: https://www.sentinelone.com/blog/how-do-attackers-use-lolbins-in-fileless-attacks/
3. GTFOBins. GTFOBins: a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems [Internet]. 2024 Oct [cited 2025 Aug]. Available from: https://github.com/GTFOBins/GTFOBins.github.io/blob/master/README.md
4. Pliskin R. Azure LOLBins: protecting against the dual use of virtual machine extensions [Internet]. Microsoft Defender for Cloud. 2021 Mar [cited 2025 Aug]. Available from: https://www.microsoft.com/en-us/security/blog/2021/03/09/azure-lolbins-protecting-against-the-dual-use-of-virtual-machine-extensions/
5. Keshet Y. What are LOLBins and how do attackers use them in fileless attacks? [Internet]. 2025 Jan [cited 2025 Aug]. Available from: https://www.cynet.com/attack-techniques-hands-on/what-are-lolbins-and-how-do-attackers-use-them-in-fileless-attacks/