4 Ways Adversaries Hijack DLLs | CrowdStrike

By Falcon OverWatch Team

Archived: 2026-04-06 00:59:52 UTC

Dynamic link library (DLL) hijacking is frequently written about by defenders due to its applications in evading automated detections. This technique is even more frequently used by adversaries in interactive intrusions. Despite the wealth of literature available to increase defenders' awareness of DLL hijacking, CrowdStrike® Falcon OverWatch™ threat hunters see adversaries gravitate toward this tradecraft time and again to load malicious code. Put simply, adversaries do this because it works.

A DLL is a file containing code that can be loaded by an application. The use of DLL files is commonly seen in the Microsoft Windows operating system, along with others. According to Microsoft, the purpose of DLL files is to "promote modularization of code, code reuse, efficient memory usage and reduced disk space."

What is DLL Hijacking?

DLL hijacking is a technique used to load malicious code for the purposes of defense evasion, persistence and privilege escalation. Rather than execute malicious code directly via an executable file, adversaries will leverage a legitimate application to load a malicious DLL file. This technique may enable malicious code to bypass application allowlisting or other automated controls; further, casual inspection of the running process only shows the legitimate application running.

One reason DLL hijacking remains difficult to mitigate with automated defenses alone is the technique offers adversaries so much flexibility and variability in its implementation. And so, the cat-and-mouse game between defender and adversary continues.

This blog examines four implementations of DLL hijacking, as well as how Falcon OverWatch threat hunters see them used in the wild, fine-tune their hunts and augment CrowdStrike's automated detection capabilities accordingly.

Watch this short video to see how Falcon OverWatch proactively hunts for threats in your environment.

To hijack a DLL, an adversary typically needs three things: a malicious DLL, a legitimate application to hijack (ideally one configured to run with elevated privileges), and working knowledge of Microsoft Windows, including how it determines which DLL files an application should load and which actions to take when a DLL's location is ambiguous.

Adversaries are known to use the following four methods of DLL hijacking, all described in greater detail below:

1. Search order hijacking
2. Relative path DLL hijacking
3. Phantom DLL hijacking

4. DLL redirection

1. Search Order Hijacking: Taking Advantage of Predictable OS Behavior

Search order hijacking — perhaps the DLL hijacking example best known to security testers — is when an adversary takes advantage of the well-documented behavior of the Windows operating system to "trick" it into running malicious code under a legitimate process.

Imagine an application developer creates an executable file written to `c:\app\app.exe` that loads a DLL `code.dll` by name only and relies on Windows to identify the correct location. The developer assumes the DLL will be at `C:\shared\code.dll` and that the `C:\shared` directory is added to the PATH environment variable when the application is installed. However, a knowledgeable attacker knows the Windows operating system searches a predefined list of locations for a DLL when the DLL's location is ambiguous and not explicitly defined. These locations vary depending on how the operating system is configured but often look something like this:

1. The directory from which the application loaded
2. The system directory
3. The 16-bit system directory
4. The Windows directory
5. The current directory
6. The directories that are listed in the PATH environment variable

If an adversary can write a malicious `code.dll` file to any one of the other locations listed before the PATH environment variable, then the malicious file will be loaded first. Further, if the `app.exe` file is executed with elevated privileges, then the adversary's malicious code will be loaded and executed with the same privileges, thereby facilitating an unauthorized elevation of privileges.

Adversaries like search order hijacking because it only requires them to drop a single DLL to the right location. This technique typically involves exploiting a pre-existing installation and is often done to elevate privileges. However, Falcon OverWatch threat hunters rarely see this technique, likely due to the amount of effort required for an adversary to identify vulnerable installed applications.

2. Relative Path DLL Hijacking: Abusing the Search Order by Moving the Executable

One of the most common DLL hijacking techniques that Falcon OverWatch sees is a variation of search order hijacking known as relative path DLL hijacking. This is when the adversary writes (and typically renames) a legitimate executable file, alongside their malicious DLL, to a folder they have adequate permissions to write to.

This technique requires a legitimate executable that does not specify an absolute path for DLL files. If an absolute path is not specified, then Windows operating systems will search for the DLL file following the predefined search order. As noted above, this search order can vary between operating systems and the settings configured, but one of the locations that is often early in the search order is the directory from which the application loaded, known as the relative path (i.e., `./`). A number of executable files, including some published by Microsoft, behave this way.

In one investigation, Falcon OverWatch observed an adversary write a renamed copy of `applaunch.exe` — a Microsoft executable file — to the `c:\users\public` directory. The file was renamed to make it blend in with the normal operation of the host. The adversary also wrote their malicious DLL named `mscoree.dll` to the same directory. When the renamed executable file was launched, it loaded the malicious DLL and executed the adversary's code.

To mitigate attempts to hide attacks in unexpected directories, Falcon OverWatch actively hunts for execution from unusual locations and maintains a list of executable files that might load a DLL from a relative file. These efforts, combined with looking for rare files across CrowdStrike telemetry, allow threat hunters to detect attacks that may otherwise go unseen.

3. Phantom DLL Hijacking: Taking Advantage of Missing DLLs

The Windows operating system references a surprising number of DLL files that don't exist. Phantom DLL hijacking is when the adversary writes a malicious DLL to the location of one of these missing files. This DLL is then loaded when the operating system runs the code that references that file.

Phantom DLL hijacking is best demonstrated with an admittedly simple example. The IKEEXT service is present on many versions of Windows, runs at startup and is used for authentication and key exchange in Internet Protocol security. When it starts, IKEEXT attempts to load the file `C:\Windows\System32\wlbsctrl.dll` — however, this DLL doesn't exist. If an adversary can write a malicious DLL file to this location (or other locations not covered here), their malicious code can be executed when the IKEEXT service is (re)started.

'In the above example, note an adversary would need to already have administrative privileges to be able to write to the System32 directory. This example is a persistence mechanism since the adversary configured the IKEEXT service to start when the system boots. Leveraging services that are expected to run on startup would likely evade detection by cursory inspection, reinforcing the need for proactive and comprehensive hunting.

Many malicious versions of this file being written will be detected and blocked by the CrowdStrike Falcon® platform — usually because they are known to be malicious or they share characteristics with previously observed malicious files. Falcon OverWatch hunts for activity that is difficult to detect or prevent. One way hunters do this is by looking for files being written that are rare or unique across the CrowdStrike telemetry. Low-prevalence files are considered suspicious and warrant further investigation. If an investigation deems a file malicious, Falcon OverWatch will not only notify the victim organization but also tag the file as malicious so the Falcon sensor can autonomously detect future attempts to use the file across the entire Falcon install base. This is true of all malicious files identified by Falcon OverWatch.

4. DLL Redirection: Changing the Search Order to Suit the Adversary's Needs

DLL redirection is perhaps one of the most novel ways to hijack a DLL. Instead of leveraging the predefined search order, in DLL redirection attacks the adversary changes the location at which the operating system searches for the DLL file. For example, an adversary can make changes to the registry to modify the search order and cause a program to run a different DLL file.

The MSDTC service, which is used to manage transactions across multiple servers, is another example of a Windows service that attempts to load a missing DLL and is vulnerable to the phantom DLL hijacking method discussed above. An adversary could write a malicious DLL to the default location `C:\windows\system32\oci.dll` and (re)start the MSDTC service to load their malicious code. However, this would be quickly detected by Falcon OverWatch, as hunters look for rare files being written to this location. Some adversaries will therefore attempt the more evasive method of changing the location that Windows checks when loading this DLL. By modifying the following registry key, an adversary can change the name of the file that Windows will use when starting the service:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSDTC\MTxOCI\OracleOciLib
```

By default, this key contains the value `oci.dll`, but an adversary could change this value to any filename — `evil.dll`, for example. The adversary would then simply write their file to `C:\Windows\System32\evil.dll` and restart the MSDTC service, and their malicious code would be executed. This would bypass the detection technique used above. Falcon OverWatch also looks for changes to registry keys like this, so it would be detected and investigated.

Which DLL Hijacking Method Do Adversaries Prefer?

The most common form of DLL hijacking observed by Falcon OverWatch is relative path DLL hijacking. This is likely due to the minimal effort it requires:

- The adversary doesn't need to know much about their target system, so the amount of reconnaissance required is reduced, and therefore so is the perceived likelihood of being detected.
- They don't have to tamper with any other part of the system, such as the registry, which they believe reduces their chances of them being detected.
- They can operate from almost any directory, which increases the search space for defenders, which again lowers the perceived likelihood of being detected.

While DLL redirection and phantom DLL hijacking are less common — likely due to the overhead required to identify suitable attack paths — Falcon OverWatch has seen these leveraged by sophisticated state-nexus adversaries with relative frequency and so they should not be overlooked.

How Falcon OverWatch Identifies and Prevents DLL Hijacking

Adversaries routinely hijack DLLs to attempt to circumvent automated security controls. Using a global-scale dataset, Falcon OverWatch threat hunters can quickly and accurately identify these DLL hijacking attempts.

The power of the Falcon OverWatch threat hunting model is the volume of data that hunters can leverage to quickly pinpoint whether a particular DLL is malicious, based on its prevalence within a global real-time dataset. Falcon OverWatch continuously hunts for globally rare DLL files, files written to suspicious locations and programs executing from unusual locations. These hunting techniques rely on baselines that come from extensive real-time data, something that cannot be replicated by any individual organization in isolation. Falcon OverWatch can augment even the most mature security programs with the power of global data. By being part of the Falcon OverWatch ecosystem, your organization benefits from the efforts of the global threat hunting team.

See for yourself how the industry-leading CrowdStrike Falcon platform protects against modern threats. Start your 15-day free trial today.

Additional Resources

- *Read about the latest threat hunting trends in the 2022 Falcon OverWatch Threat Hunting Report.*
- *Learn more about Falcon OverWatch's proactive managed threat hunting.*
- *Discover the power of tailored threat hunting provided by Falcon OverWatch Elite.*
- *Find out why part-time threat hunting is simply not enough.*
- *Learn more about the CrowdStrike Falcon® platform.*
