Explainer: Packed Malware

By Shellseekerscyber

Published: 2024-01-07 · Archived: 2026-04-05 18:16:34 UTC

What is Packed malware?

Packed Malware is malicious code that uses compression or encryption to hide its malicious features. This explainer will focus on compression.

Packed malware requires the following 3 elements:

1. **The packed executable** — the compressed/encrypted part that, when unpacked (decompressed, decrypted or both), carries out the malicious activity.
2. **The stub** — the part that tells the victim host how to unpack the packed executable. Stub is often used interchangably with the term *wrapper*.
3. **The packer** — the program that the malware author uses to pack (compresses, encrypts or both) the malware. This is not delivered to the victim.

Why pack malware?

The primary motivation to pack malware is to bypass security measures and make the malware more difficult to analyze.

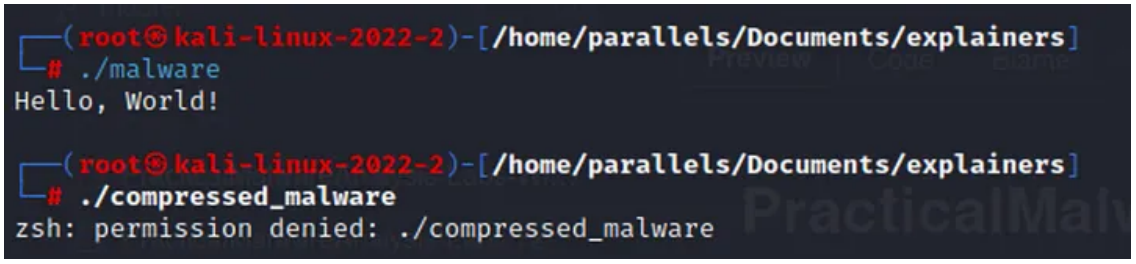
If we take a simple example of a compiled *Hello, World!* binary and compress it, we can see why packed malware is more likely to bypass security tools for two reasons:

Packed malware has a different signature. When you pack malware, the compressed executable will have a different hash from its unpacked equivalent. If this new hash has not been added to the signature library of an antivirus or EDR tool, there is a chance it will remain undetected. If our antivirus was looking for the hash of the malicious file *malware*, ‘packing’ our *compressed_malware* has evaded this detection.

Press enter or click to view image in full size

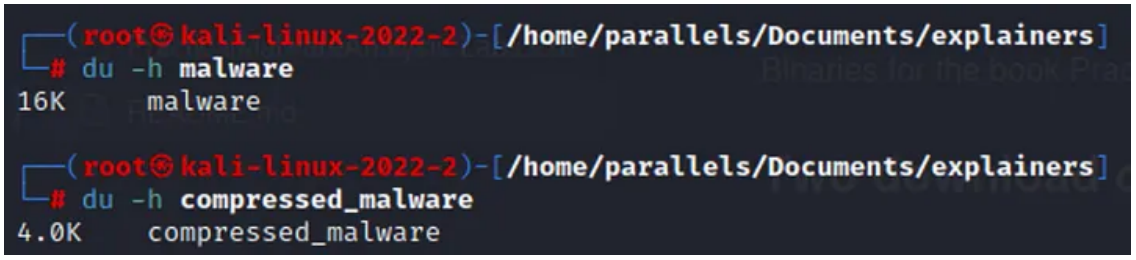
Sandbox tools are more likely to fail to analyze packed malware. More advanced security tools (most modern EDR platforms) use sandboxing techniques to detonate (run) unknown software, analyze its behaviour and then classify it as malicious or benign. With more advanced packers, it is likely that the sandbox will be capable of working out how to ‘unpack’ (decompress) the packed malware, and as such will not be able to detonate it.

Press enter or click to view image in full size



Packed malware tends to have a smaller file size. This tends to be the least important of these 3 reasons, but is a welcome consequence of packing malware. Like normal file compression, packing reduces the filesize of the malware, which makes it easier to deliver to the victim.

Press enter or click to view image in full size

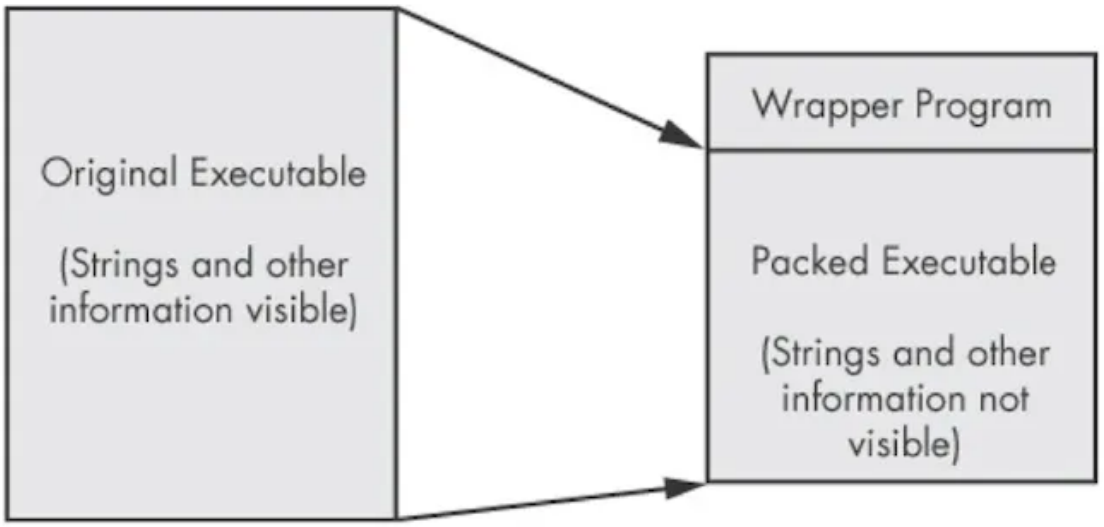


This demonstration is not realistic. However, even simply compressing this file exemplifies some of the advantages of packing malware.

What does packed malware look like?

To write packed malware, authors write a malicious program, then ‘pack’ it with a stub which instructs the victim’s host on how to unpack the malware so it can be run. When the packed malware executes on the victim host, the host interprets the stub, which unpacks the malicious executable. That unpacked executable is then able to run its malicious instructions.

The stub itself is not packed (compressed or encrypted) — if it was, the victim host would not not be able to understand how to unpack the malicious program that will then be executed.



Diagrom showing the general structure of unpacked (R) vs packed (L) malware. Here, the term wrapper is used instead of stub. *Source: Page 13 of Practical Malware Analysis, Sikorski and Honig*

Simplifying the concept

A helpful analogy might be furniture: you can buy a fully-assembled table from a shop that you can take home, put in your kitchen and use immediately. Alternatively, you can buy IKEA flat-pack furniture. The IKEA box arrives with 2 components in the box, the flat-pack furniture and the instructions.

Get Shellseekerscyber's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

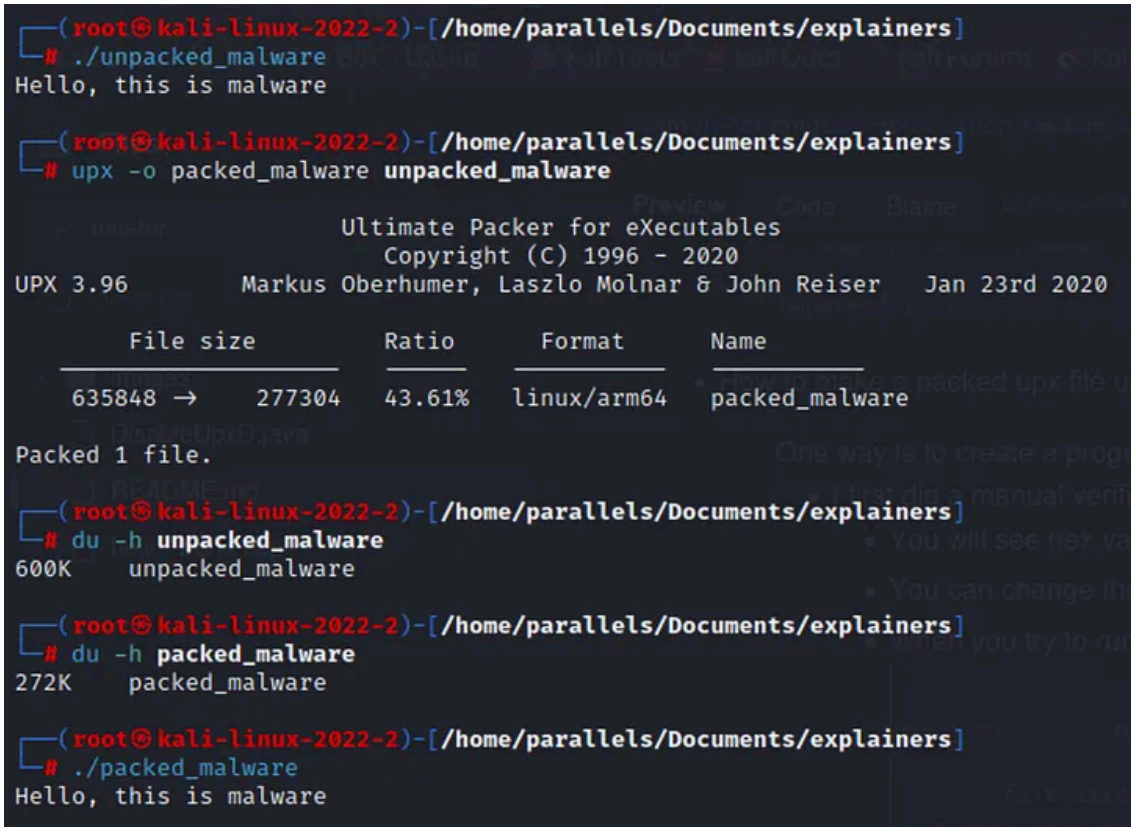
The flat-pack table does not look like a table or perform the function of a table until it is assembled using the instructions (for this analogy to work, you also have to pretend that you use the instructions when assembling IKEA furniture… bear with me).

What is the packer?

The packer is the program that compresses or encrypts the packed executable. The most common packer is UPX (Ultimate Packer for eXecutables) — an open source tool.

Packing can be as complex or as simple as the malware author chooses. Below we see an example of UPX being used to pack a binary. Keep in mind, UPX packs binaries with a stub by default, so unlike our example with GZip earlier, the UPX-packed binary will still run, because the stub instructs the operating system on how to unpack the binary.

Press enter or click to view image in full size



Here, unpacked malware is run, then packed, showing the drop in file size due to compression. However, since UPX includes a stub in packed_malware, the operating system is still able to execute the malware.

How does the stub work?

The decompression stub is typically located at the beginning of the packed executable. It is a relatively small portion of code, often just a few hundred bytes in size.

During runtime, the decompression stub executes its instructions to locate and decompress the compressed sections within the packed file.

Some decompression stubs include **anti-analysis techniques** to impede dynamic analysis tools and sandboxes. These techniques might include checks for virtualized environments or delays in unpacking to make automated sandbox analysis more challenging.

Some advanced decompression stubs may possess polymorphic capabilities. This means they can generate different instances of themselves dynamically, making it more difficult for static signature-based detection mechanisms to identify the decompression stub itself.

How can you analyze packed malware?

Fundamentally, the challenge is in identifying the method used to pack the malware. However, it should be noted that this is an enormous topic in and of itself, and this write-up only attempts to outline the processes without going into any meaningful detail.

Firstly, the analyst must identify packing signatures. This can be using a tool such as PEiD (PE iDentifier) that can be used to identify packers, cryptors, and compilers used in PE (Portable Executable) files.

Finally, the analyst must actually unpack the malware. This could be as simple as using:

```
upx -d packed_malware.exe
```

However, more advanced malware is likely to use custom or obfuscated packing techniques, making it difficult for the analyst to identify any packing signatures.

Summary

The article explains packed malware, a malicious code that hides its features through compression or encryption. It consists of three elements: the packed executable, the stub (or wrapper), and the packer. The main purpose of packing malware is to bypass security measures and complicate analysis by changing signatures and reducing file size. It should be noted that this article has examined the surface level of this extremely deep topic.

Press enter or click to view image in full size

A picture of a laptop. Fascinating stuff. Photo by Philipp Katzenberger on Unsplash

References

Michael Sikorski and Andrew Honig. 2012. *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software* (1st. ed.). No Starch Press, USA.
