UPX Packers and Unpacking

Malware Analysis

By Jeffrey Farnan

TABLE OF CONTENTS

INTRODUCTION

In this report we will look at UPX Packers, which are tools used to hide malicious files by encrypting and compressing the file to make the code unrecognizable allowing it to bypass antivirus software. I will explain how packers work and look at the different types of packers. There are ways to unpack malware, we will look at how this process is accomplished and the tools used to achieve it.

What are Packers

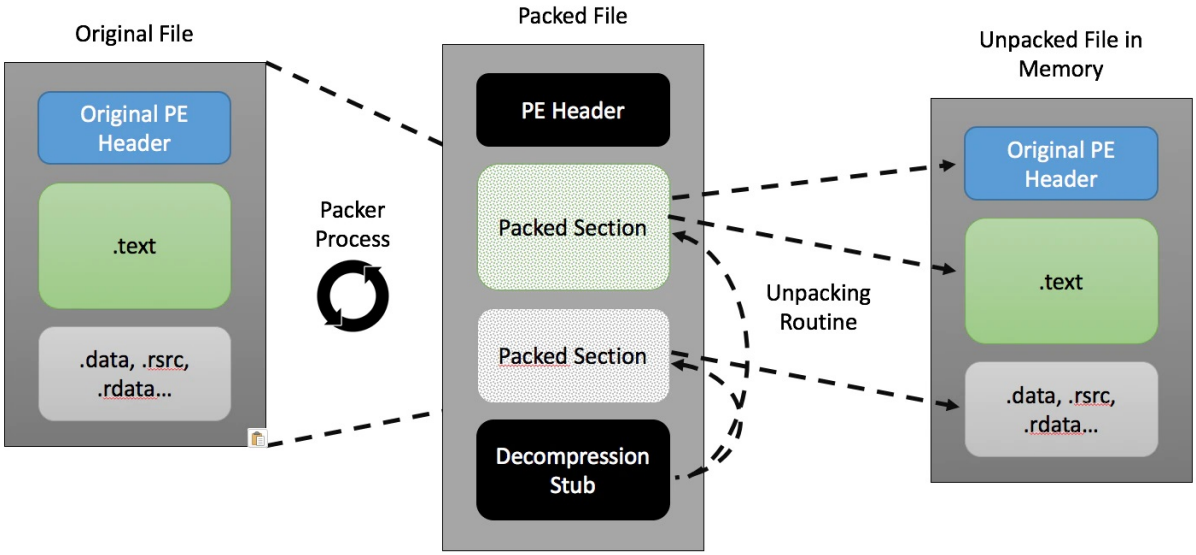
- The term “packed” can refer to any file which and been compressed or encrypted.
- A Packer or “*runtime packer*”is a special tool designed to automatically decrypt itself and executes its payload in memory, allowing it to avoid detection and analysis by security researchers and antivirus products.
- Packers make it harder to analysis malware by security researchers, by not been able to identify the behaviour of the malware, and the increased amount of time required to for analysis.
- The packers themselves are not malware.

Packing

- Packing, a.k.a “Executable compression” is a means of compressing an executable file and combining the compressed data with the decompression code into a single file.
- Packing has been used for many years to bypass security vendors and antivirus products. This type of malware is usually known as polymorphic malware, as the packing application can repackage the same executable file, creating different MD5’s which are distributed to victims, so static signatures become useless.
- When an executable is packed an encoded version of the malware can be stored in a variable, possibly encoded with a key. At execution time, the program generates a key and decodes the malware. The malware is loaded into memory, the program is unpacked and then jumps to the address and executes the malicious payload.

HOW PACKERS WORK

- A packer is simply a tool to armour the binary. The whole binary is protected. The more advanced organizations or cybercriminal groups may employ custom packers or implement protection inside malicious files.
- The original executable is compressed/encrypted, then wrapped in a new executable which contains code to bring it back to its original state.
- When packers encrypt or compress a file, a stub is also created. This stub is a piece of code that contains the decompression or decryption routine.
- Once the file is running, the decompression stub stored in the packed file will decompress the packed section. The original .exe file is then loaded into memory than executed.



- The packer produces its own header Section. This PE Header can be modified by scrubbing out information or changing information making it harder for Malware Analysis's to identify the packer and determine its correct size.

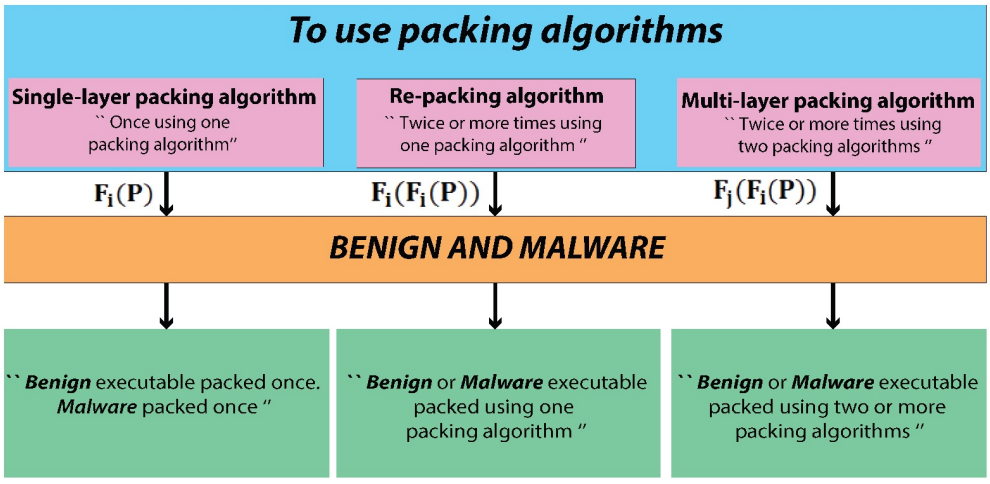
DIFFERENT TYPES OF PACKERS

There are many different types of Packers:

- **Commercial Packers:** UPX, PECompact, MPRESS, Exe Packer 2.300, ExeStealth, ASProtect.
- **Modified Packers:** These are packed with common packers but the executables is modified so not to be detected by antivirus products.
- **Mult-iPacked:** Files that have been packed several times, using a variety of packers.
- **Unknown Packers**: Files that have been compressed by packers that are rarely encountered – for example, packers that demonstrate a proof of concept.
- **Custom Packers**: Use proprietary algorithms to bypass standard detection techniques. Many of the emerging custom packers are polymorphic, which simply means that they use an anti-detection strategy whereby the code itself changes frequently, but the purpose and functionality of the malware remains the same.

Multi-layer Packers

- Executables can be packed with single or multi-layer packing algorithms, used in both benign applications (Companies that don't want their software reversed engineered) and malware.



Polymorphic Packing

One of the main aims of Malware is to avoid detection, one of the ways it does this is use a Polymorphic Engine.

- **Polymorphic Engine:** This is a software component that uses polymorphic code to alter the payload while still providing the same functionality. These engines are almost exclusively used in malware, encrypting or obfuscating the payload.

- **File Binder**: Another added component is called a File Binder, which weaves malware into normal files, such as office documents. Since this type of malware is usually polymorphic, it is also known as a polymorphic packer.

UNPACKING MALWARE

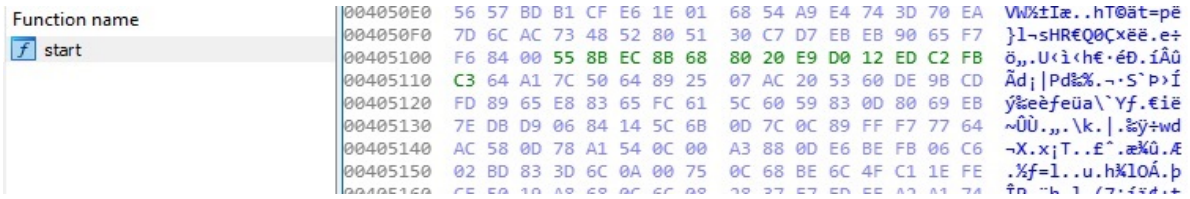
Unpacking is the process of restoring the original malware executable. Until we unpack a malware file, we cannot see even the basic functionalities and attributes of the packed file like strings, imported API's or exports of a file. In this section we will look at techniques and tools to unpack malware.

We'll be analyzing the unpacking of Malware in a Windows VM with some tools like ID and x64dbg.

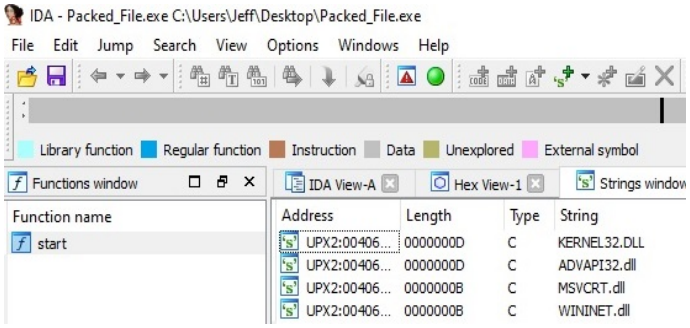
- Look inside a packed file with PEiD and IDA PRO.
- Monitor the unpacking of a executable with ProcMon.
- Unpack a standard UPX file.
- Look at how a UPX file can be modified so standard unpacking won't work.
- Manual unpacking a modified UPX in x64dbg with the plug-in Scylla .
- Since the unpacking happens in-memory, we will look at the memory allocation and we will use the x64dbg to place some breakpoints to see the process for unpacking.

Contents of a Packed File

Looking at a packed file in IDA Pro, we can see the contents of the file are encrypted and unreadable.

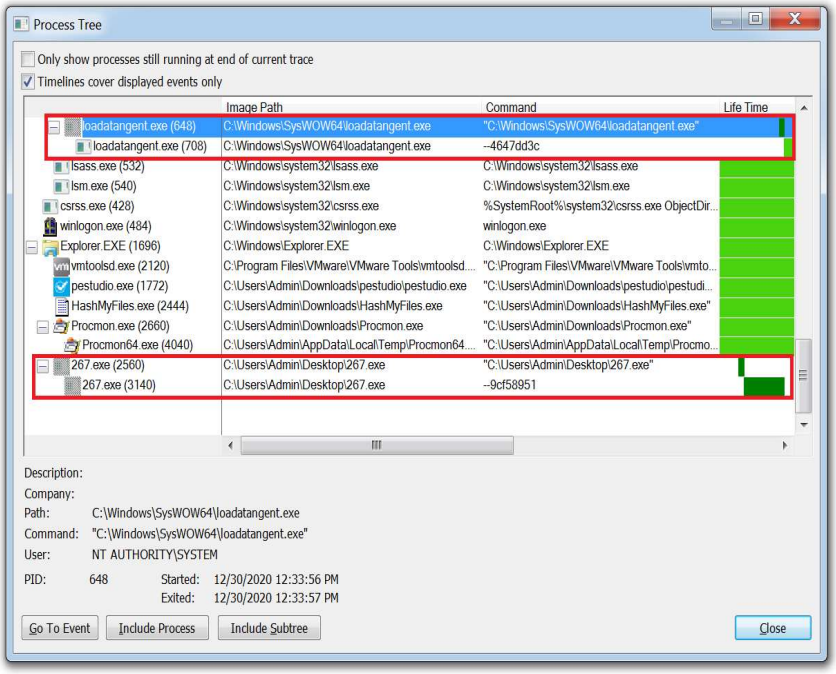


There is only one function available, the start function and the strings show the most common used API strings.



Unpacking with ProcMon

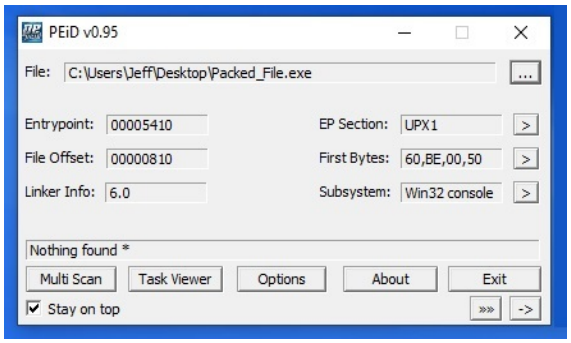
- One technique to monitor the unpacking of a piece of malware on a system is to use Process Monitor (ProcMon)
- ProcMon records filesystem activity on the machine it is running and can be filtered to show any new processes which have been created.
- A piece of packed malware on the system is called ‘267.exe’, to unpack itself the malware creates a new process, this is the child process with the same name.
- The new child process is then allocated free space within it by the parent process. The unpacked code is then injected into the free space within this newly created child process, this is called process injection



UNPACKING UPX

UPX shorthand for the Ultimate Packer for executables.

- UPX is a free, portable, extendable, high-performance executable packer for several executable formats.
- It typically compresses better than WinZip/zip/gzip.
- Its written in C++ and because of its of its class layout it's very easy to add new executable formats or new compression algorithms.
- A typical malware file packed with UPX will look something like this in PEiD



Unpacking UPX

UPX uses the command line to unpack the file with the command:

upx –o Unpacked_File.exe Packed_File.exe

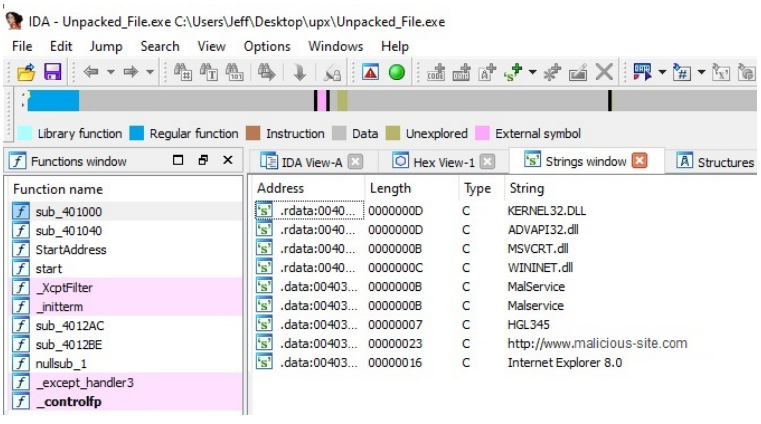
```
Unpacked 0 files.

C:\Users\Jeff\Desktop\upx>upx -o Unpacked_File.exe -d Packed_File.exe
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96w       Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
     16384 <-      3072   18.75%    win32/pe     Unpacked_File.exe

Unpacked 1 file.
```

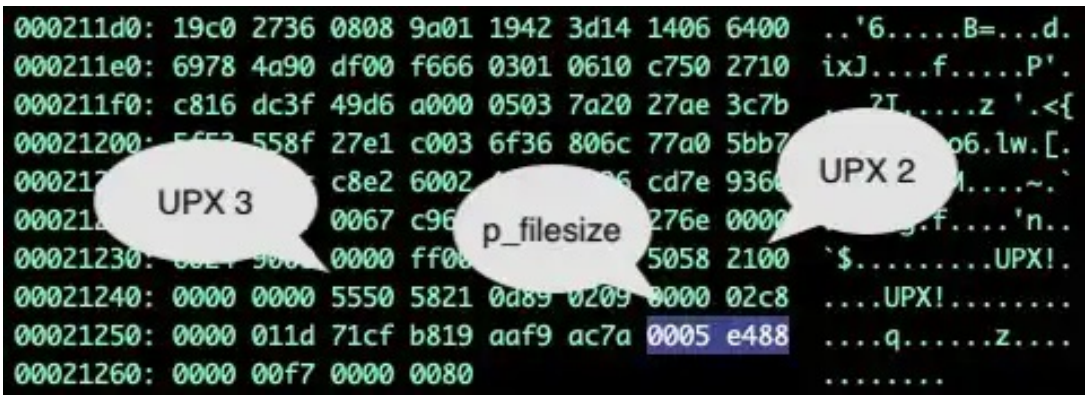
Looking at the unpacked file in IDA Pro we can see more functions, plus more strings are available. We can see from the strings this file is indeed malware.



MODIFED PACKED EXECUTABLES

Malware Authors will try many techniques to make it harder for security researchers to unpack files while still able to unpack and execute on their target systems.

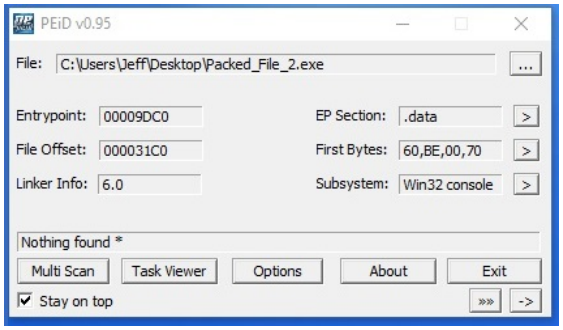
- **Section Header**: Tampering with the Section Header prevents commonly used unpacking tools from gathering information about the image sections, and prevents the UPX tool from easily unpacking their malware.



- **Self-Debugging:** Self-debugging is used to prevent another debugger from being attached to the parent process.
- **Exceptions:** Using exceptions can make your disassembler/debugger do all kinds stuff like crashing, suddenly exiting to running in a loop.

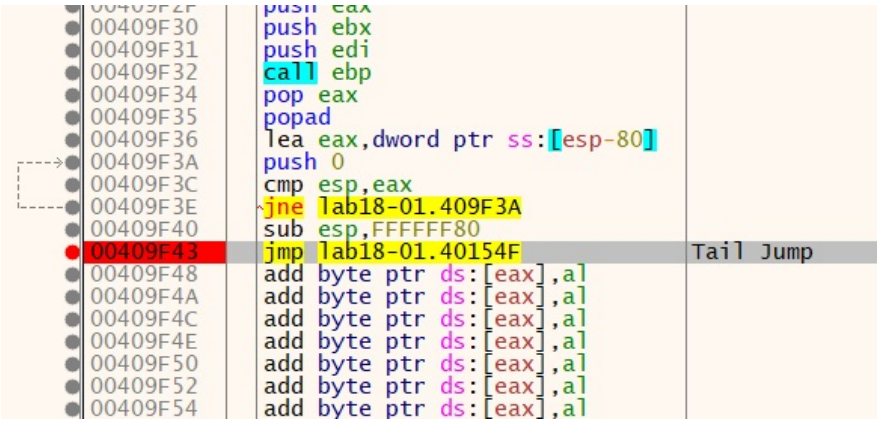
UNPACKING A MODIFIED UPX PACKER

The next packed file when placed in PEid does not detect the packer



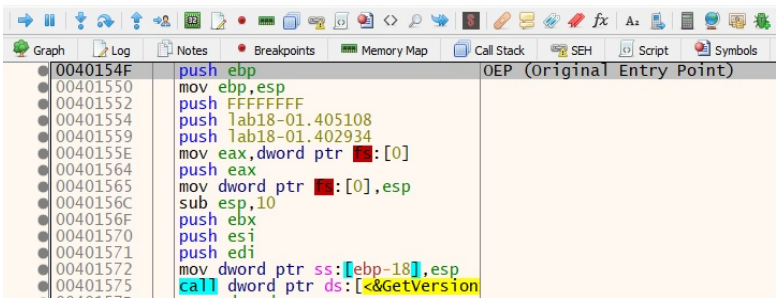
This file is packed with a slightly modified version of UPX. A section in the file named UPX 2 made me suspect this was a UPX-like packer. A modified UPX packer makes it more resistant to signature detection.

Loading this file into a debugger and searching for the tail jump. The tail jump is a jump instruction to the original entry point to the malware.



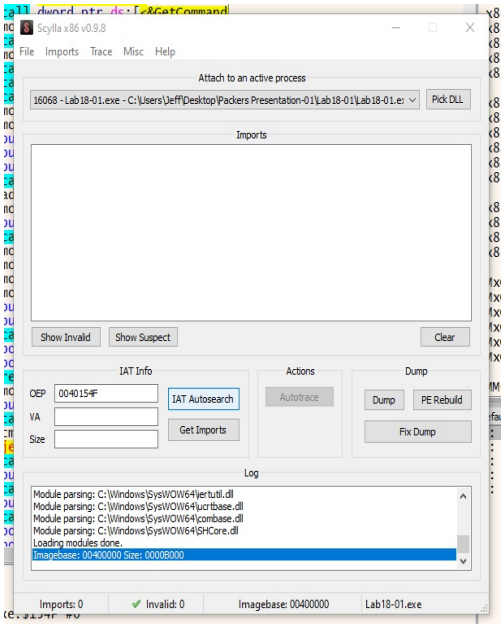
We set a breakpoint at this location **40154F**. This is the original entry point of the program. We run the program and step into this location.

Here we hit the original entry point.

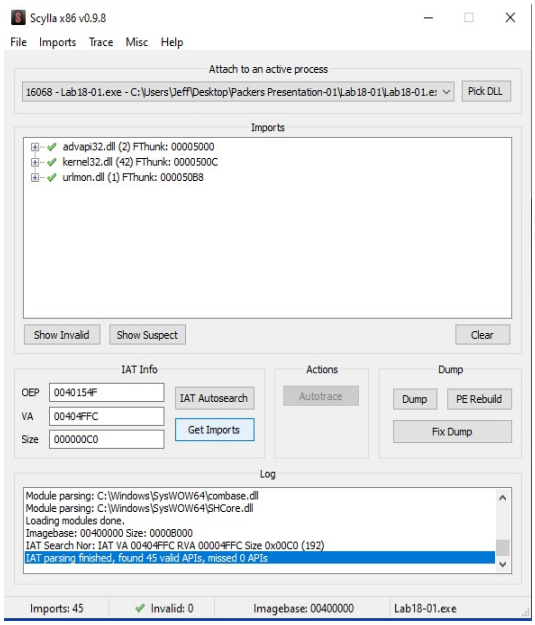


Scylla Plug-in

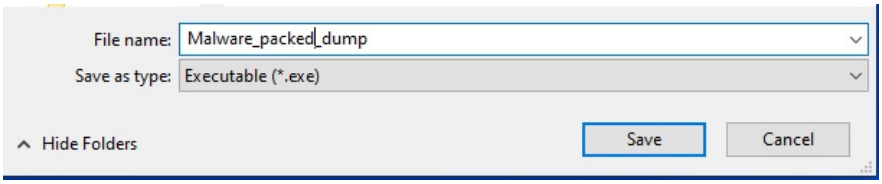
We use the Scylla Plug-in in x64dbg to extract the binary.



I have to click Autosearch to scan for Imports. When that is complete I click on Get Imports to get a list of all the imports found.



I click on Dump to dump the extracted binary.



I save the binary and open it in IDA pro where now I have access to the full executable.

USER PROTECTION AGAINST PACKED MALWARE

Packers are not inherently bad, they help to protect files, data and applications. They are a great resource for Malware developers helping to obfuscate file code making it difficult to detect and be analyzed. Although unpacking a suspicious file is normally beyond most users, here are some helpful measures you can take:

- Have anti-malware software on all your devices, having it updated can help protect you against suspicious packers.
- If the file is packed with UPX a simple solution is to download UPX and using the following command line the file can be unpacked: **upx -d -o unpacked.exe packed.exe**
- If a file is packed by an unknown vendor or one of those vendors mentioned above, the best course of action is to delete it. If the packed file comes from an unknown vendor or untrusted website and you are suspicious of it, again the best course of action is to delete it.