Current Trends in Packing and Obfuscation[1]

How they fit into today's landscape of Advanced Persistent Threat and Cyber Crime as a Service.

Authors
Paul Moon MSc (Royal Holloway, 2014)
William Rothwell, Abatis (UK) Ltd

Overview

Malicious software has been around nearly as long as software itself and so too has computer security software to defend against it. The terms malware and anti-virus (AV) have been very common in the world of computer security for decades. These concepts have evolved a great deal over that time, but throughout the existence of malicious software many methods of packing and obfuscating a payload have been used to prevent them from being detected and analysed, and ultimately prevent the infected victims from being remediated.

In a time where security vendors face state-sponsored threats, hacktivism, cyber crime as a service and destructive malware attacks wiping or encrypting personal files, what impact do obfuscation and packing have on the industry? This article will look to examine these trends and why modern malware uses obfuscation and packing in the way it does, and what, if anything, the security industry can do to defend against this.

What are packers? These are tools that take a malicious file and package it to look like a harmless one. Think about a drug smuggler looking to evade customs. They will take their malicious payload and package it into something they can take across the border. A packed file cannot be identified using signatures and has totally different properties to the original file.

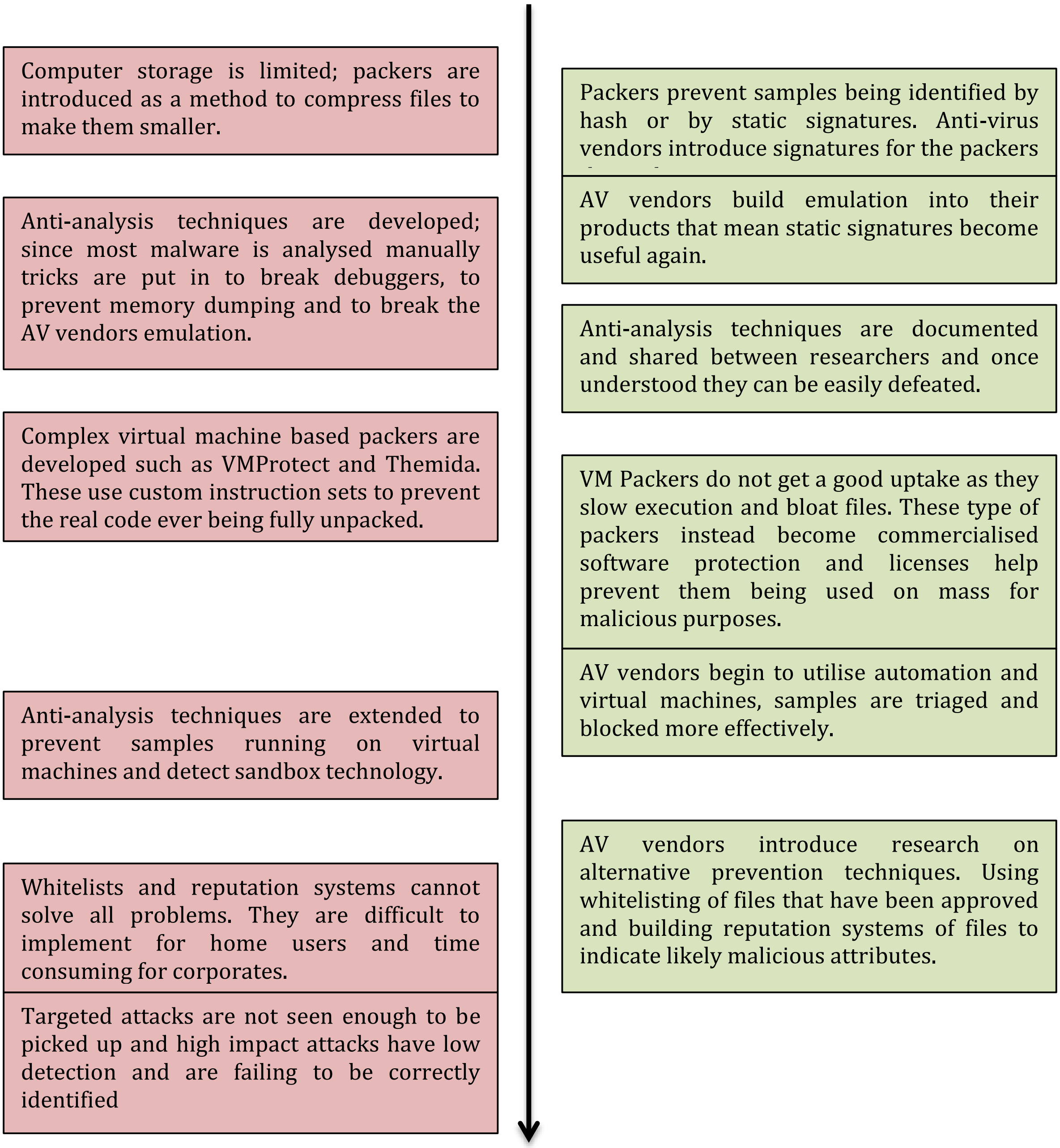
What is obfuscation? This is a method of taking the code and rearranging it to make it harder for computers and humans to interpret it. It is one of the most common methods used by packers but it can also be applied before a file is packed when the software is first compiled.

---

[1] This article is to be published online by Computer Weekly as part of the 2015 Royal Holloway info security thesis series. The full MSc thesis is published on the ISG's website.

Evolution of Packing and Obfuscation

Before looking at the current trends of packing and obfuscation it is useful to know how things got to where they are now. In the timeline below, the left shows the bad guys' bright ideas of new ways to avoid the security industry. The right shows the security industries own ways to combat these.



Computer storage is limited; packers are introduced as a method to compress files to make them smaller.

Packers prevent samples being identified by hash or by static signatures. Anti-virus vendors introduce signatures for the packers

AV vendors build emulation into their products that mean static signatures become useful again.

Anti-analysis techniques are developed; since most malware is analysed manually tricks are put in to break debuggers, to prevent memory dumping and to break the AV vendors emulation.

Anti-analysis techniques are documented and shared between researchers and once understood they can be easily defeated.

Complex virtual machine based packers are developed such as VMProtect and Themida. These use custom instruction sets to prevent the real code ever being fully unpacked.

VM Packers do not get a good uptake as they slow execution and bloat files. These type of packers instead become commercialised software protection and licenses help prevent them being used on mass for malicious purposes.

AV vendors begin to utilise automation and virtual machines, samples are triaged and blocked more effectively.

Anti-analysis techniques are extended to prevent samples running on virtual machines and detect sandbox technology.

AV vendors introduce research on alternative prevention techniques. Using whitelisting of files that have been approved and building reputation systems of files to indicate likely malicious attributes.

Whitelists and reputation systems cannot solve all problems. They are difficult to implement for home users and time consuming for corporates.

Targeted attacks are not seen enough to be picked up and high impact attacks have low detection and are failing to be correctly identified

A lot of the technologies used by both attackers and defenders in relation to packers and obfuscation still exist in modern malware. Anti-analysis techniques vary massively across families and packers are not always favoured in certain attacks. Obfuscation, however, is used across the board. It is one of the most commonly used and most developed techniques throughout the evolution of malware. This is simply because it is one of the most difficult problems to solve using traditional methods, and affects both manual and automated analysis.

For this reason many people believe AV is dead, certainly we believe when it comes to solely static signature based detection it is.

What are anti-analysis techniques? These are technical measures to prevent a malware sample from being analysed either manually or via automation and make detection and identification more difficult. Anti-analysis can be applied at the packing layer to prevent unpacking or within the unpacked sample itself. The table below shows some malware analysis techniques and how the bad guys seek to thwart them:

Malware Analysis Technique	Anti-Analysis Counter
Debugging: Step through the code as it is executing to see what it is doing.	**Anti-debug:** A technique to thwart manual unpacking and analysis. The sample would detect if a debugger was attached and change its behaviour.
Emulation: One of the techniques adopted early by AV vendors was to emulate the code, a compromise between executing the sample and analysing it statically so ideal for unpacking.	**Anti-emulation:** Techniques to detect emulation or cause it to fail such as using instructions the emulator does not recognise or running for a long time, since emulation time is usually limited.
Memory dump: At some point the unpacked sample must be resident in memory. If the memory can be dumped at this point it saves a lot of time on unpacking.	**Anti-dumping:** A technique to prevent taking a dump of process memory. Usually done by modifying information required to take an accurate dump such as header information or even large chunks of code that are removed and placed elsewhere in memory.
Virtualisation: Most malware analysis takes place on a Virtual Machine (VM) as they can be reset to a clean image and re-infected very quickly.	**Anti-VM:** Attackers know this so deploy mechanisms to detect a virtual machine and alter the samples behaviour if one is detected.
Disassembly: The best way to understand the code is to read it. A static disassembler takes the byte code and makes it semi-readable.	**Code Obfuscation:** A technique to make reading the disassemble code more difficult and time consuming. Usually the analyst will have to write more code to de-obfuscate it before they can make sense of it.

Sandboxing: A process to take many samples and automate their analysis using virtual machines. They will be executed in a way to make them believe they are on a victim and information about the malicious behaviour will be recorded.	**Anti-Sandbox:** A more recent development in the anti-analysis tool box. The sample will attempt to identify if it is running in a sandbox, through connectivity checks, known sandbox identification or simply waiting longer than the sandbox execution time to preform its malicious behaviour.

Current Trends

Virus Total gives a good indication of the current status of modern malware. More than fifty AV providers and over 200,000 malware samples a day are listed. This demonstrates a huge resource, both from the security industry and from the attackers. In the majority of cases identification of the files is simply "generic bad". This shows an inability to accurately detect or identify the majority of these samples.

Targeted Attacks	**Crimeware for financial gain**
To stay undetected for as long as possible and maintain a foothold on the victim.	To stay undetected long enough to carry out the attackers goals, and cash in.
To avoid being analysed by security researchers by staying undetected.	Slow down analysis by security researchers as wide deployment means detection cannot be avoided but prevention can be slowed.
They have a low deployment surface and infect only those victims of interest.	They have a high deployment surface and attempt to infect as many victims as possible.
They are likely not to be packed; packing can sometimes make a file appear malicious. If a packer is used it will likely be a custom one that focuses on making the output file look legitimate.	Heavily packed samples are usually repacked once they are detected and sometimes automatically repacked on a timed schedule creating thousands of packed copies of a single sample.
Instead of packing, custom obfuscation to prevent detection and hinder analysis is used. Even in very basic samples efforts are taken to hide obvious artefacts such as human readable strings.	Packing can contain complex code obfuscation and the unpacked sample can utilise this technique too. Packer obfuscation is usually used to prevent static signature detection where as obfuscation in the unpacked sample is designed more to slow analysis.
Either a very simple throwaway tool that is just redeveloped once detected or very	The main payload is usually complex but simple loaders and droppers are

complex implants where a lot of effort has gone into detection prevention. Campaigns tend to be very modular with a range of tools and implants only deployed to victims, as they are required.	used as a first line of defence to ensure a foothold on the victim before this is deployed.
Developed specifically for the needs of the actor.	Likely both the packing service and malware are purchased by the actor and not customised for their use.

All this malware can loosely be categorised into two main threat actor groups: those that are performing targeted attacks, generally for intelligence or political purposes, and those that are performing crimeware attacks, mass infecting victims for financial gain. Each have different needs and thus use packing and obfuscation in very different ways:

An aspect often overlooked is how each of the two groups above unwittingly helps the other one out. For example an automated packing solution for crimeware, such as the one used in the Shylock botnet before it was taken down, would produce thousands of samples per day. These would for all intents and purposes be a single sample but it would take thousands of automated analysis runs to establish this. This weighs heavily on the processing load of the security industry and leads to confusion with sample naming. The effect of all this is that it is harder to identify targeted attacks through standard AV product lines.

It is not a one-way street either. More recently mass criminal malware has taken a leaf out of the targeted attack manual and shifted to use a modular approach, not exposing the crown jewels too early in the infection process. This helps to shield against sandboxes and prevent vendors getting good signatures to detect the main payloads.

The very different approach to compromising a victim taken by these groups means the security industry has to defend two fronts. Security companies need to make a decision about their focus. Those that are concentrating their effort on mass infection will need to optimise detection for heavily packed samples. It is likely they will have less interest in sample identification and current Virus Total results demonstrate this to be the case for most desktop AV providers. Those that focus on targeted attacks need to be intelligence-led. Identification is of equal importance to detection and grouping samples together is required to identify campaigns based upon the actor's intentions and victims. This is a looking-for-the-needle-in-the-haystack type approach that works best for Incident Response (IR) companies or those who provide threat intelligence for corporate customers that have a better insight into the context of the files they are analysing.

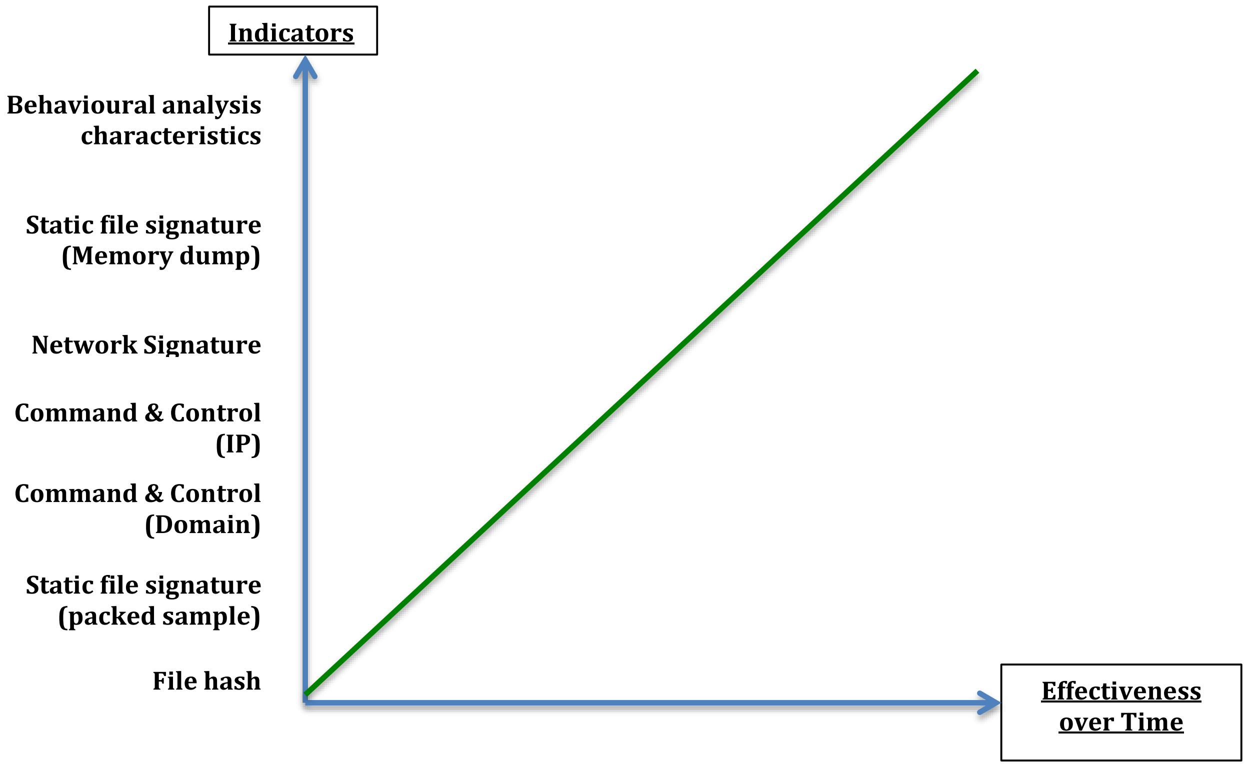
In both cases it seems that success relies upon good detection, good identification and, due to sheer numbers, intelligent use of automation. Packing and obfuscation makes both detection and identification of malware harder. For this reason it is not a trend that is going to decrease in the future. As new technologies develop and with them new malware it is easy to spot these old trends being recycled and reused. Mobile applications such as Android are a good example of this and code obfuscators and application protectors are already widely available for this platform.

Is the Security Industry Winning?

This is probably not a question that can be answered with a simple yes or no. There are certainly more and more blogs and exposés being published on success stories of defeating malware. This could be a sign of an improving security industry or just an increase in the number of attacks. In our opinion the utilization of resources is a good way of assessing who has the upper hand, since neither attackers nor defenders have infinite resource so success comes down to who uses what they have most effectively. If it takes a team of attackers 2 years to break into a company's network with only a single defender standing in their way then that is an extremely effective defence.

Another demonstration of resources is to look at how a mass malware campaign may work. A service is set up that takes a single malware sample and automatically repacks the file every 10 minutes. This service can produce thousands of packed files per day. Switching roles to the defender: is there an automated analysis system that could automatically unpack, detect and identify all of these files? To our knowledge there isn't. Manual effort would be required to reverse and identify one of the samples. From this rules could be written that may detect the rest but of course this cannot be guaranteed. If failures can be detected the analyst can be prompted to re-examine failed detections to fix the problem. Unfortunately the likelihood is the analyst will never know when the detections fail. Therein lies the defender's biggest problem: you don't know what you don't know. The conclusion is that the computer security industry needs more resources to achieve the same level of success as the attackers.

The trends in anti-analysis also highlight the security companies being on the back foot. A technique would be used in the wild and would thwart analysis until a counter measure was identified and deployed. This is a very reactive blacklisting model. If you have to identify every bad file and add it to a blacklist then it is always going to play to the attackers' advantage and exhaust defenders resources first. To detect all bad files a number of malicious indicators are required, but packing and obfuscation cause this list to grow and make each indicator less effective. The graph below shows a number of indicators and how effective they are over time as a result of packing and obfuscation techniques:



Whitelist technologies are not a panacea though. They too require a constant resource to vet every application before adding to the approved list. It may work well in controlled environments but proves tricky in open settings where users want to install new and untested applications; could your nan decide whether a copy of "Angry bird.exe" should be legitimately allowed to run?

There is a good example of whitelisting working commercially as a model: the Apple App Store. All applications must be signed by Apple to show they have been tested and are approved and no unsigned code can run on an iPhone or iPad. This is actually a good security model, even if user security may not have been the primary motivation. The effect of this approach is clear when comparing the malware landscape of iPhone with that of Android. This does not mean it is completely secure. Devices can still be exploited, a phone can have a "jailbreak" applied to remove this security and of course the review process could always miss something. What it does do though is to raise the bar for the attackers and surely it is about time they had to work a bit harder?

Summary

Whilst operating from a blacklist model, packers and obfuscation make it impossible for the security industry to win. By increasing the number of indicators and the resource required for developing and maintaining them attackers simply have the upper hand. The masses of samples mean small deployments such as targeted attacks go unnoticed or are hard to identify.

There are some approaches the security industry can take to level the playing field. Crimeware defenders must understand that packing and obfuscation make most indicators useless and effort should be put towards development of automated behavioural analysis techniques. For targeted attacks an intelligence-driven approach is needed, focusing on extracting and grouping samples away from the noise. Where custom obfuscation or packers are used these can make good signatures if they are unique to that group. Working towards identifying the actor's techniques and motivations is an effective way to develop defence for future attacks.

Until there is a drastic shift in the computer security model it will be an uphill fight for the industry and it is likely that packers and obfuscation will continue to enable attackers for the foreseeable future.

Biographies

Paul Moon's current role is as a senior security researcher for Crowdstrike, looking mostly at complex targeted attacks but also large-scale botnets. The job mainly involves reverse engineering malicious files, developing code and trying to make it a bad day for the adversaries. Before that he performed similar duties for the UK Government in an attempt to secure UK networks of significance.

William Rothwell is a qualified security professional and an experienced security practitioner with over 25 years in the information security field with clients including government, military, finance and commercial organizations. He specializes in security architecture and design, crypto application and system intrusion defence research. In 2004 he founded Abatis Limited, which provides a unique, patent protected, non-signature based solution to protect computers from malware infection and hacking intrusions.