An analysis of usage of the packer by malware

1310707 Atsushi Tomizawa

Summary:
The active region of the malware has been expanding as usage of information devices which connect to the internet continue at an explosive pace and it leaves us open to the attack of them. Due to this background, studies on the malware analysis gains a momentum. One of the research areas on the malware study is one on the usage analysis of so-called "packer" by the malware and its obfuscation techniques: Packer is a software which is generally developed for such purposes as cutting down the footprint of the software or holding countermeasures against reverse engineering, but these characteristics are also favorable for the malware to make their activities easy, resulting in the fact that almost all malware are now "packed" themselves.
There are two main targets in the packer analyses: packer identification and unpack of the packed malware. If we can successfully identify the packer, it helps unpack and retrieve the payload, in this case the malware itself.
It is the well-known method for identification of packers to analyze the binary signature, which is a pattern of binary appearing in the software code. There are special tools available to directly observe the binary signature of PE (Portable Executable) header of the software and we can easily start analyzing, however, it is not an effective method as packer has some countermeasures as ones of their obfuscation techniques against the debugging (self-modification and code layout of the file are typical examples).

In Ogawa Lab, applying BE-PUM (Binary Emulator for PUshdown Model generation) to the packer analyses is under studying. BE-PUM produces the control flow graph of the program and suitable for suitable for analyzing malware whose footprint is small enough to calculate. BE-PUM extended its support of IA32(x86) instruction sets and Windows API which enables to analyze typical obfuscation techniques and identification of packers.
BE-PUM is also applicable to unpack the packed program: in one of the researches using BE-PUM as a generic unpacker, EMDIVI, one of the famous malware in Japan, is analyzed with BE-PUM and it identified the destination server information (such as server name, username, password) to gather the stolen information. In addition, BE-PUM also identified the packer used by EMDIVI (UPX v3.0) and the Original Entry Point (OEP) of the malware with a help of the fact that UPX v3.0 ends its unpacking code with the instruction POPA.

In this study, the total of over 5000 data of the malware sample which are available in VXHeaven (a public database of malware related information) , disassembled by BE-PUM(that is, whose usage of packer are also identified) , is manually observed to try to identify the OEP of each sample. From the existing research, it is found out that there are typical obfuscation techniques for packers and that each packer has its own tendency of applying them. With the fact, a hypothesis is made that each unpacked code of the packer has its unique sequence in the execution of x86 sets. From the hypothesis, the special subset of instruction sets retrieved from the disassemble code of the malware samples are prepared. The retrieved instructions are followings: call, ret, popa, pusha together with calling Windows API which we find as results of the execution of obfuscation techniques or start of the unpacked code or malware. By observing the subset of the disassembled results and seeking a pattern of occurrence of the instructions, an attempt is made to extract unpacking code by packer. In case of successfully extraction, as next step, each original disassembled data is observed in detail, finding commonly appearing set of instructions between them. The OEP is assumed to be located at the end of them.

The observed samples grouped by packers are ASPack(194samples), PECompact (494), Upack (382) and UPX (2535).
In ASPack, a pattern of the occurrence of the instructions can be identified followed by a candidate of possible OEP, which comes after the instruction sequence of orl -> movl -> jne -> pushl -> ret (operand omitted) in the extracted data sets.

There are data in sample which can be considered as not completely disassembled by BE-PUM. It is assumed that it is because of the occurrence of the instructions or Windows API which are not supported by BE-PUM. Finding possible OEP is also successful in the sample of PECompact: it appears after the existence of popl -> popl -> popl -> leave -> ret instructions. In Upack there's no regularity in the occurrence of instruction set. In the inspection of each sample data, it is found out that there's few disassembled data and is concluded BE-PUM is presumed that it cannot finish disassembling. In case of UPX a pattern can be found in some sample data together with popa instruction, which is reported as the signal of end of the unpacking code, in the vicinity. In addition, same occurrence of instruction set is also found after the possible OEP in some sample data: the reason cannot be surmised but it suggests that malware themselves can also have some implementation in common between themselves or the end instruction of unpacking code is not POPA. The followings are also observed across the entire data samples: possible variety of malware, which are judged after the naming, has same tendency of occurrence of instruction. Even though the name of the malware is totally different, some samples have the same tendency of occurrence. There's some malware whose size of instructions disassembled is not big enough to identify the packer but BE-PUM did.

In this study, retrieved instruction and Windows API are fixed throughout the experiment. If observation is done with another set of choice, another knowledge could be obtained. Together with the fact that the preparation of the data and process of observation is fixed, the method can be suitable to apply for the robotic process automation or mixed with machine learning with different set of instructions to be extracted (in fact, preparation of the sample data is partly done by implementing a data shaping program in the experiment). Other possible prospect for the studies are followings: applying the observation method to recently active malware sample (because sample of VXHeaven is old in whole and difficult to get the same version of packer used which is necessary to create test program to observe the unpacking process). Giving feedback of the observation results for additional extension of BE-PUM.
(1058 words)