

基于精简指令集的软件保护虚拟机技术研究与实现

汤战勇,王怀军,房鼎益,陈晓江

(西北大学 信息科学与技术学院,西北大学—爱迪德信息安全联合实验室,陕西 西安 710127)

摘 要: 针对软件知识产权与版权保护需求,对多样性技术和基于虚拟机的软件保护技术进行分析和研究,设计了一种基于寄存器的精简指令集软件保护虚拟机 SPVM,并实现了一个具有高强度和扩展性的防逆向工程和篡改、防止核心算法破解和防盗版的软件保护虚拟机系统 VMDefender. 通过采用多样性和虚拟机等技术,最终实现了防止软件盗版和对软件机密信息以及软件核心技术的保护.

关键词: 虚拟机;软件保护;软件多样性;软件保护虚拟机

中图分类号: TP314

文献标识码: A

文章编号: 1000-7180(2011)08-0001-03

Research on Software Protection RISC-based Virtual Machine Technology

TANG Zhan-yong, WANG Huai-jun, FANG Ding-yi, CHEN Xiao-jiang

(School of Information Science and Technology, Northwest University,
NWU-Irdeto Net-Security Joint Lab (NISL), Xi'an 710127, China)

Abstract: In response to the needs of software intellectual property and copyright protection, the technology about diversification and virtual machine protection was analyzed and researched, a register-based software protection virtual machine based on reduced instruction sets (SPVM) is designed, and a software protection virtual machine system VMDefender is implemented. Through the diversification and virtual machine technology, software piracy is avoided, and the leakage of software confidential information and core technologies is prevented.

Key words: Virtual Machine; Software Protection; Software Diversity; Software Protection Virtual Machine

1 引言

计算机和互联网技术的飞速发展,使得计算机软件和网络服务得到了广泛应用. 软件的研发和维护需耗费大量人力物力,但大量的软件盗版和软件破解行为使得软件开发者的利益受到严重侵害,如何保护软件版权和核心技术日益成为困扰软件开发商以及整个软件产业发展的重要问题.

目前软件保护方法主要有基于软件和基于硬件两种类型^[1],基于软件的保护方法众多,易于实施,成本低,但保护强度不高;基于硬件的保护方法保护强度较大,但受硬件限制不易实施且不利于用户使

用,成本高. 基于虚拟机的软件保护方法^[3]是通过定义独立的指令系统,将基于 x86 系统的关键代码转化为只能由虚拟机解释执行的虚拟机伪代码,采用多样化^[4]处理,在加强了软件防破解和防盗版能力的同时,有效降低了软件保护的成本,增强了受保护软件的易用性,可有效的实现对软件的保护.

软件保护虚拟机具有一套自定义的指令系统,并将一系列指令字节码(byte code)放在对应虚拟机解释引擎中执行. 在该指令系统中,大部分常用的汇编指令都可以用虚拟机中的相应指令所代替,为了保护软件中关键的代码,将这些为人熟知的 x86 指令替换为虚拟机指令系统中的指令.

收稿日期: 2011-05-16; 修回日期: 2011-06-24

基金项目: 国家自然科学基金(61070176);陕西省教育厅产业化示范项目(2010JC24)和西北大学—爱迪德信息安全联合实验室项目支持

2 软件保护虚拟机处理流程

虚拟机软件保护可分为两个部分:①是保护部分,即把待保护的代码块转化成虚拟机代码,②是执行部分,即虚拟机解释器正确地执行虚拟机代码。

软件保护虚拟机处理流程如图 1 所示。

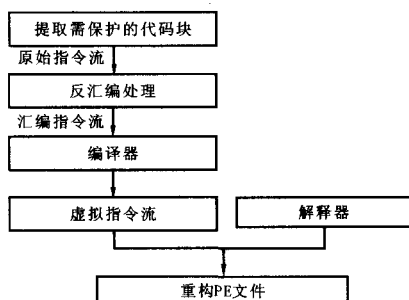


图 1 软件保护虚拟机处理流程

下面对每个主要流程进行简单介绍:

① 提取需保护的代码块:在进行代码虚拟化时,为防止软件执行速度和效率的大幅度下降,通常只选取重要的代码块加以保护,一般采用以函数体为单位进行保护。

② 反汇编处理:反汇编^[2]是指将 x86 机器码指令流反汇编成汇编指令流,VMDefender 借助现有的各种反汇编引擎实现,借助现有的成熟的 x86 汇编作为中间语言,一方面保证了中间语言的全面性和合理性,另一方面利于系统实现的简单高效。

③ 虚拟机编译器:将汇编指令转变为虚拟机自定义的伪指令,并保存在相应的结构和存储单元中。当真正需要执行该段代码时,再由虚拟机解释器进行解释执行。在编译时,利用随机变化方式^[4-5],将 x86 汇编指令与虚拟机 P-code 的映射为一对多或多对多,使攻击者在短时期很难找出规律^[7]。

④ 虚拟机解释器:解释执行虚拟机指令集,确保能正确地完成任务就是指令解码、分发和执行,其中也可以加入防篡改检测等保护机制。

⑤ PE 文件重构:被修改后的代码还必须遵循 PE 文件的基本格式,需要将虚拟化后的代码写回到原 PE 文件中^[2]。此外,一般需要扩大原程序的空间,把虚拟机代码和解释器附在程序的空闲区域或尾部。

根据上述流程,给出软件保护虚拟机 SPVM 的系统架构,如图 2 所示。

3 软件保护虚拟机系统 VMDefender 的设计与实现

3.1 VMDefender 系统结构

基于虚拟机 SPVM 的设计,软件保护虚拟机系

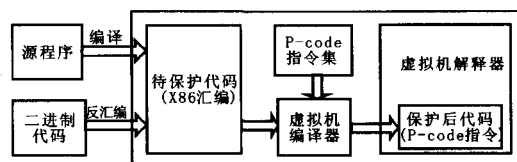


图 2 软件保护虚拟机 SPVM 系统结构

统 VMDefender 主要由四个部分组成:保护定义模块、指令生成模块、编译转换模块和解释执行模块。

保护代码定义与提取模块:完成受保护软件的关键代码块的标记以及提取;

指令生成模块:定义生成虚拟机 P-code 指令格式和指令表,并且加入多样化处理,对使用频繁的指令序列设置超级指令集;

编译转换模块:将 x86 二进制代码反汇编成汇编代码,并将汇编代码编译转换成虚拟机 P-code,转换时使用随机变换原则,防止攻击者通过对比猜测 x86 指令与虚拟机 P-code 映射关系;

解释执行模块:在受保护软件运行时,虚拟机被激活,虚拟机根据虚拟机指令表对虚拟机代码进行解码,然后分发执行,完成解释执行功能。

3.2 虚拟机的构造

软件保护虚拟机需要模拟 CPU、虚拟的内存,还有应用程序的堆栈,在这里我们使用 VMMachine 结构体来存储虚拟机寄存器和堆栈,其实现如下:

```

struct VMMachine
{
    DWORD eax;
    .....
    DWORD eip;
    .....
    BYTE OF;
    BYTE p_code;//伪代码
    DWORD lpVMStack[1024 * 1024 * 2];//堆栈基址
};
  
```

3.3 虚拟机 P-Code 实现

虚拟机伪代码(P-Code)设计是软件保护虚拟机中重要部分,本文通过学习 x86 指令的设计,给出了一个才采用定长和多样化处理的 P-Code 设计方法,对 P-code 的变换处理可以使用下面几种方式:①定长指令,通过查表完成;②多级表编码,提供简单的可变指令长度;③任意长指令,通过遍历二叉树解码。在 VMDefender 中,我们采用的是定长指令,通过查表完成。

3.4 虚拟机编译器

虚拟机编译就是将汇编代码转换成虚拟机字节

码的过程.为了设计的简洁和开发的高效,我们没有单独设计新的虚拟机高级语言(VM HLL),也没有为编译设计虚拟机汇编语言,而是采用了将原始 x86 指令代码反汇编后的结果作为虚拟机编译器的输入,通过将原 x86 汇编代码转换成虚拟机的 P-code 完成编译转换过程.在编译转换过程中采用多样性思想,采用随机生成映射表的方法降低攻击者破解的效用和影响力.VMDefender 的虚拟机编译器 VMCompiler 在实现时采用的是线性映射,使用 Switch 语句对 x86 汇编进行映射变化,并将虚拟机字节码填充到原始二进制文件中.

3.5 虚拟机解释器

针对软件保护虚拟机的特殊性,VMDefender 采用解释执行方式.虚拟机解释器的主要工作就是模拟 CPU 的功能,对虚拟机字节码进行解码、分发和执行.虚拟机解释器的实现和真实的 CPU 类似,具体处理流程如下:

① 取指令:当进入收虚拟机保护代码段时,启动虚拟机解析器 VMEngine,读取虚拟机字节码;

② 解码:在真实的 CPU 里这个过程会比较复杂,例如 x86 的指令集指令变长且数量众多,因此比较复杂,而在虚拟机里则不一样,一般 Opcode 和 Operand 可以很简单地被分开;

③ 执行:对指令解码后,派遣至各虚拟指令的真实执行处,通常针对每一个 Opcode 会有一段代码完成相应的功能;

VMDefender 采用分支派遣的解释执行的方法,针对每一个 Opcode 均有一个对应的解释函数,然后通过 switch 进行分派.

4 实验与分析

本文以 CrackMe 程序中的一个测试加密函数 bool Encrypt(BYTE * pInData, BYTE * pOutData)作为测试对象,该加密函数采用最为简单的异或加密方式进行.

首先用 IDA pro 分别对 CreakMe.exe 和受保护后的 CreakMe.vmd.exe 进行静态反汇编分析.CreakMe.exe 中 Encrypt 函数入口地址为 0x00401110,Push ebp,mov ebp,esp 是明显的函数入口标志,向下可以看到函数的异或加密实现部分,通过分析攻击者完全可以逆向出 Encrypt 的实现原理.而 CreakMe.vmd.exe 中 Encrypt 函数入口 0x00401110 则变成了 push 7F6D2814h,jmp loc_42BB06,并且 IDA 在对 Encrypt 进行智能识别时出

现 sp-analysis failed,已经看不到 Encrypt 的实现部分,该函数的原始实现已经被转换成虚拟机代码,其真实的执行也将由虚拟机解释器来完成.通过对比看出,虚拟机软件保护技术可以有效防止攻击者的逆向静态分析.

其次使用 Ollydbg 分别对 CreakMe.exe 和受保护后的 CreakMe.vmd.exe 进行动态分析.CreakMe.exe 中 Encrypt 函数入口在未使用虚拟机保护前清晰可见,用户可以进行很方便的调试,跟踪函数的加密过程,而使用虚拟机保护后,函数入口已经变为 Push 和 Jmp,已经很难通过动态跟踪看到异或加密实现的本体,其真正的实现过程由虚拟机完成,攻击者很难进行动态逆向分析.

综上所述,虚拟机软件保护可以有效实现对软件的静态和动态的逆向分析,如果在其中在嵌入反调试和防篡改的措施,则可有更好的保护强度和力度.

5 结束语

基于虚拟机的软件保护技术是软件保护技术里的一个新兴分支,虚拟机软件保护技术以虚拟机为平台,融合软件水印、代码混淆等保护技术和多样化思想,可提供一个具有高强度和扩展性的,防逆向工程和篡改,反静态分析和动态调试的软件保护解决方案.在保护软件核心算法破解和防盗版方面,具有广阔的发展前景.

参考文献:

- [1] Collberg Christian, Nagra Jasvir. Surreptitious software: obfuscation, watermarking, and tamper proofing for software protection[M]. New York: Addison - Wesley, 2009.
- [2] 段钢. 加密与解密[M]. 3 版. 北京:电子工业出版社,2009.
- [3] 舒柏程,李毅超,曹跃. 基于虚拟机的软件保护技术研究[J]. 计算机工程与科学,2008, 30(A1):97-102.
- [4] Matthias Jacob, Mariusz H. Jakubowski. The superdiversier: peephole individualization for software protection[J]. Advances in Information and Computer Security, 2008:100-120.
- [5] Anckaert B, Jakubowski M, Venkatesan, R. Proteus: virtualization for diversified tamper - resistance[C]// Proceedings of the ACM Workshop on Digital Rights Management, New York: ACM Press, 2006:47-58.

(下转第8页)

参考文献:

- [1] Samarati P, Sweeney L. Generalizing data to provide anonymity when disclosing information[C]// Proc of the PODS '98. New York, 1998: 188.
- [2] Sweeney L. Achieving K-anonymity privacy protection using generalization and suppression[J]. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 2002, 10(5): 571-588.
- [3] Sweeney L. K-anonymity: a model for protecting privacy[J]. International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, 2002, 10(5): 557-570.
- [4] Wong R, L J. (α , k)-anonymity: an enhanced k-anonymity model for privacy-preserving data publishing [C] // Proc of the 12th ACM SIGMOD Int'l Conf. New York, 2006: 754-759.
- [5] Truta T M, Vinay B. Privacy protection: p-sensitive k-anonymity property[C]//Proc of 22nd IEEE Int'l Conf. on Data Engineering Workshops. Washington DC: IEEE computer Society, 2006: 94-103.
- [6] Nergiz M E, Clifton C. MultiRelational k-Anonymity [C]//Proc of the IEEE 23rd Int'l Conf. Turkey: Istanbul, 2007: 1417-1421.
- [7] Machanavajjhala A, Kifer D, Gehrke J, et al. ℓ -diversity: privacy beyond k-anonymity[C]//Proc of the 22nd Int'l Conf. on Data Engineering. New York: ACM, 2006: 24-35.
- [8] Li Ninghui, Li Tiancheng. T-closeness: privacy beyond k-anonymity and ℓ -diversity [C]//Proc of 23rd Int'l Conf. on Data Engineering. Turkey, 2007: 106-115.
- [9] Xiao Xiaokui, Tao Yufei. Personalized privacy preservation[C]//Proc of ACM SIGMOD Conf. on Management of Data. USA, Chicago, 2006: 229-240.
- [10] Bugra Gedik, Liu Ling. Protecting location privacy with personalized k-anonymity: architecture and algorithms[J]. Mobile Computing, 2008(7): 1-18.
- [11] Xiao Xiaokui, Tao Yufei. M-invariance: towards privacy preserving republication of dynamic datasets[C]//Proc of the ACM SIGMOD Int'l Conf. on Management of Data. China: Beijing, 2007: 689-700.

作者简介:

刘英华 女, (1975-), 博士研究生, 讲师. 研究方向为数据挖掘、知识发现、隐私保护.

(上接第 3 页)

- [6] Shi, Y, Casey, K, Ertl, M and Gregg, D. Virtual machine showdown: stack versus registers [J]. ACM Transactions on Architecture and Code Optimization (TACO), 2008, V4(4): 1-36.
- [7] 于森, 孙强. 对超粒度混杂技术的改进: 基于瘦虚拟机的指令集交替技术[J]. 计算机应用, 2005, 25(12): 2808-2810.

作者简介:

汤战勇 男, (1979-), 博士生. 研究方向为网络与信息安全和软件安全与保护.

王怀军 男, (1981-), 博士生. 研究方向为软件安全与保护.

房鼎益 男, (1959-), 教授, 博士生导师. 研究方向为网络与信息安全、软件安全与保护、无线传感器网络.

陈晓江 男, (1973-), 博士, 副教授. 研究方向为传感器网络、信息安全.