广西科学院学报
Journal of Guangxi Academy of Sciences

2009,25(4):327~329,332
Vol. 25,No. 4 November 2009

游戏软件反外挂技术方案的设计和实现方法
Anti Illegal-Plug-Game Based on Software Protection Technology

高 翊[1],张旸旸[2]
GAO Yi[1],ZHANG Yang-yang[2]

(1.南宁市平方软件新技术有限责任公司,广西南宁 530003;2.中国电子技术标准化研究所,北京 100007)
(1. Naning Pingsoft New Technology Co., Ltd., Nanning, Guangxi, 530003, China; 2. China Electronics Standardization Institute, Beijing, 100007, China)

摘要:根据软件外挂和软件保护的技术特点,选择现有的几种适合反外挂的软件保护方法,设计游戏软件反外挂技术方案,并描述其实现的关键技术。

关键词:游戏软件 反外挂 软件保护
中图法分类号:TP311.52 **文献标识码**:A **文章编号**:1002-7378(2009)04-0327-03

Abstract: The research of the anti illegal-plug-game based on software protection technologies includs a series of methods and their characteristics is produced. After analyzing and selecting suitable technologies, a solution and realization of anti illegal-plug game is illustrated.
Key words: game, anti illegal-plug, software safeguard

目前,游戏软件厂商对游戏软件的画面质量、可玩性、市场广和占有,以及游戏软件为公司所能带来的利润问题很关注,对正在制作或者运营的游戏软件安全性重视程度却不够。他们对游戏软件安全的侧重点主要放在服务器端,例如资料数据库的安全性,以及游戏软件服务器的安全性和稳定性,对游戏软件客户端程序缺乏保护性的工作,使得一些游戏软件程序几乎完全暴露于黑客之手。黑客拿到游戏软件运营商发布的客户端后,可以利用反编译或反汇编等工具得到反汇编代码,任意地进行研究、调试和修改,严重地威胁到游戏软件的安全性。

许多种游戏软件外挂就是因为程序安全防范性能不高,导致被破解工具破解制作出来的。控制/模拟式外挂与监听/修改式外挂就属于对网络游戏软件软件安全的侵犯范畴。对于由于网络游戏软件客户端程序的安全性能问题而出现的这两种外挂,现在的大多数防范措施是从网络通讯协议加密的角度出发来做相应的安全处理,如利用 MD5、SSL 等加密算法,或者研究和利用更新、更有效率的加密算法来防范。但是这些措施只是从提高通讯协议加密效率的方法入手,显得十分被动,黑客在得到最新的客户端程序后,还是利用各种工具对代码进行分析、跟踪和调试,对网络协议加密进行破解只是时间问题。有的黑客高手还可以绕过加密算法等更加巧妙的方法来达到破解目的。尽管提高加密算法性能的方法在短时期内可以阻止游戏软件外挂出现,却还是没有解决实质问题,不能有效地解决外挂问题。所以,针对控制/模拟式与监听/修改式这两类游戏软件外挂,作者在了解现有各种软件保护方法[1~6]各自的技术性能和优缺点的基础上,选择其中一些最佳软件保护技术,重新组合设计游戏软件反外挂技术方案,主动和有效地从源头上杜绝游戏软件外挂的制作和使用。

1 游戏软件反外挂技术方案的设计思路

游戏软件反外挂技术方案的主要功能是防止黑客利用工具了解游戏程序内部信息,所以需要针对外挂的特点,综合各方面因素,从现有的软件保护方法中选择合适的保护方法,优化客户端游戏软件的代码,或者是作为一个反外挂的功能模块加到新游

收稿日期:2009-09-10
作者简介:高 翊(1974-),男,硕士,主要从事网络游戏、软件保护研究工作。

戏软件开发的需求中，使反外挂技术的性能最佳化，既达到防止外挂的目的，又保证游戏软件原有的各项性能。

从外挂制作的特点来看，黑客所使用的破解工具很全面，类型很多、功能也很强大，主要的方式是获取静态代码信息，进行动态跟踪，监视文件读写情况和对程序篡改。所以，选择反外挂的软件保护方法时，一方面需要考虑选择一些具备代码加密、代码迷惑、反调试、反跟踪、反逆向工程、反篡改能力的软件保护方法，另一方面需要考虑被选择的多种方法能相互结合，共同形成一套反外挂解决方案。此外，每增加一种软件保护的方法都会付出一定的代价，例如消耗CPU、增加代码量、增加内存的占用、影响软件的可靠性和稳定性，最突出的还是影响程序的运行速度，所以在选择多种方法结合的同时，还需要对投入成本、方法对游戏软件产品本身的影响和对反外挂所能起到的效果等多方面因素进行权衡，找到比较折中的解决方案。

由于游戏软件的客户端光盘可以任由用户复制，所以对客户端代码保护一般来说就不必使用认证和授权等方法。同时在方案设计中还有些经验可以借鉴，例如：使用汇编语言实现软件保护的方法，比使用高级语言实现要安全一些；把保护语言直接放到要保护的代码中，而不是放在分离的DLL库中；只需要在关键的代码上使用保护代码，而不是用在整个代码段，这样程序的稳定性和效率会相对高一些；程序开始和运行过程中，不断检测是否有调试器存在，如果存在则终止程序。

这样，设计游戏软件反外挂技术方案就可以采用几种软件保护方法相结合进行深度防护，其基本思想就是采用一系列、分散防护的策略，如果一层防护不行，还有另外一层防护可以抵挡；如果一种保护被突破，还有其它的保护措施。采用的措施越多，黑客的破解过程就越是复杂和困难，程序就越安全，反外挂的效果越明显[7]。

2 游戏软件反外挂技术方案的设计和实现方法

2.1 反外挂技术方案设计

游戏软件反外挂技术方案就是由加密、反破解、代码迷惑、调试探测、反调试、反篡改等方法组合的技术，从静态、动态和反篡改等多个方面共同防止黑客利用工具了解游戏软件程序内部信息，同时又不影响游戏软件正常工作。其中，加密是防止客户端代码被分解或者分析，并且直到许可认证成功前，保护软件的代码不被解密[8]。反破解是抗反汇编和反编译，使黑客无法通过工具得到客户端的静态代码，无法分析代码结构。代码迷惑是使黑客破解得到代码是经过技术处理的，而无法了解代码的内容。调试探测是在软件被尝试调试时禁止被保护软件运行，或者直接退出程序。反调试是禁止黑客设置断点绕过调试探测，使得调试过程十分麻烦，或者根本无法绕过调试探测，也就不能了解程序执行流程以及各状态的信息。反篡改是前面的防护工作全部都被破解，黑客也制作出外挂程序，但是正常游戏软件程序不受外挂程序篡改的影响，一样能起到防外挂的作用[9]。

由于各种保护方法可以被看成一个单独的模块，模块的功能很明确也很简单，过程流程比较明显，所以使用结构化的方法来设计。首先总结出系统应有的功能并将系统功能细分，从功能完成的过程考虑，将各个功能的过程列出，标识出过程转向和传递的数据；分析各过程之间的耦合关系，合理地进行模块划分以提高它们之间的内聚性，也使模块具有信息内聚性[9]。这就可以完成方案设计。

2.2 反外挂技术方案实现的关键技术

2.2.1 加密技术

黑客制作外挂时最希望获得的游戏代码是关于网络通讯模块和设备接口的部分，考虑到游戏的执行速度，所以只针对这两部分关键代码进行加密。这里所说的加密指的是软件保护方面的。实现加密技术时，首先在程序中关键代码位置调用解密函数，使编译完成的程序不能执行；其次是在调试时先记录调用解密函数的地址，再找到编译完成后程序的相应地址，对该函数进行加密。

//对协议包进行加密。加密算法可以根据实际情况进行优化和定制。

```
void encrypt(void * addr,int bytes){
  unsigned char *s;
  unsigned char c= 0x45;
  unsigned char i;
  s = ( unsigned char * )addr;
  for(i = 0;i<bytes;i++){
    s[i]^ =c;
    c+=s[i]; } }
//对上述加密协议包进行解码
void decrypt(void * addr,int bytes){
  unsigned char *s;
  unsigned char next_c;
  unsigned char c= 0x45;
```

```
unsigned char i;
s = (unsigned char *)addr;
for(i = 0;i<bytes;i++){
  next_c = c+s[i];
  s[i]^ =c;
  c = next_c; } }。
```

2.2.2 反破解技术

反破解的实现是在代码中加入部分汇编语言,创建简单的宏作为迷惑反汇编工具。

```
#define paste(a,b) a##b
#define pastesymbols(a,b) paste(a,b)
#define OBFUSCATE() \ //加入干扰汇编语言的宏定义
_asm { mov eax,_LINE_ * 0x635186f1 };
_asm { cmp eax,_LINE_ * 0x9cb16d48 };
_asm { je pastesymbols(Junk,_LINE_) };
_asm { mov eax,pastesymbols(After,_LINE_) };
_asm { jmp eax };
_asm { pastesymbols(Junk,_LINE_): };
_asm { _emit (0xd8 + _LINE_ % 8) };
_asm { pastesymbols(After,_LINE_): }。
```

2.2.3 调试探测

调试探测是在游戏代码中插入检测代码,检查进程中是否有动态分析代码的工具出现,一旦发现就做异常处理。

```
_try{
_asm int 1;} //检测是否有动态分析代码工具出现
_except(TestSingleStepException
(GetExceptionInformation())){ }
//捕获异常
 int TestSingleStepException (LPEXCEPTION
—POINTERS pExceptionInfo){
DWORD ExceptionCode = pExceptionInfo—>
ExceptionRecord—>ExceptionCode;
if (ExceptionCode ! = STATUS_ACCESS_
VIOLATION)
printf ("SoftICE is present!");
return EXCEPTION_EXECUTE_
HANDLER;}。
```

2.2.4 反调试技术

反调试是在游戏程序正常执行过程中不断检测,是否被调试器工具用断点打断,如果有断点出现,就说明有调试器在调试程序,一旦发现就做异常处理,及时阻止黑客了解程序执行情况。

```
_try{
_asm{
pushfd
or dword ptr [esp],0x100 // Set the Trap Flag
popfd
// Load value into EFLAGS register
nop } }//设置检测是否有断点调试的标志
_except(EXCEPTION_EXECUTE_
HANDLER){
bExceptionHit = TRUE; // An exception has
been raised
// there is no debugger. }
//出现异常
if (bExceptionHit == FALSE)
printf ("A debugger is present! \n")。
```

2.2.5 反篡改技术

反篡改是在检查到有鼠标或键盘的修改操作时,就立即对硬件进行中断检查,如果没有中断则说明修改是来自外挂,需要丢弃。

```
_try{
_asm int 33;}
//如果有鼠标或键盘的操作,检测是否有中断
_except(TestTamperException
(GetExceptionInformation())){ }
//捕获异常
 int TestTamperException (LPEXCEPTION_
POINTERS pExceptionInfo){
DWORD ExceptionCode = pExceptionInfo—>
ExceptionRecord—>ExceptionCode;
if (ExceptionCode ! = STATUS_ACCESS_
VIOLATION)
printf ("Your code is tampered!");
return EXCEPTION_EXECUTE_
HANDLER;}。
```

3 结束语

很多软件保护方法可以用于网络游戏软件反外挂。本文根据外挂和软件保护的技术特点,选择其中几种适合反外挂的方法,设计一套采用分散风险策略和多种技术的深度防护方法,并描述了实现的关键技术。本文的研究从过去反外挂时所使用的仅仅通过提高网络协议安全程度的一味防守的方向,转换到向外挂程序主动攻击,通过软件保护方法保护程序不能被反编译、跟踪和篡改,从根源上铲除网络游戏软件外挂的制作,无法制作出游戏软件外挂当然也无法使用外挂。另外,对游戏软件程序实行深度防护,采取多层防护方法共同保护游戏软件程序,提高黑客破解程序的难度,消耗他们更多的精力和时间,提高了破解成本,这样,不但延长了游戏软件的安全运行时间,而且也会在一定程度上达到迫使黑客放弃尝试的目的。

值得注意的是,每一层防护被破解是不可避免

(下转第 332 页)

比，将对比的结果记录下来。比如，为了证明A数据是准确的，找出一个已经被证明是准确的数据样本B，通过设置A、B的各种限制条件 R，对A和B进行横向或纵向的比较。在对比过程中，先设置一个阀值 $f \in R$，当对比的差值大于阀值 f 后，就认为数据是可疑数据，然后对数据进行标记。同时，还可以衍生地对A数据和B数据的样本进行分割 $A=[A_1, A_2, A_3, \cdots, A_n]$，$B=[B_1, B_2, B_3, \cdots, B_n]$，设置对比规则 $R=[R_1, R_2, R_3, \cdots, R_n]$，对比A和B的样本记录异常数据。这样采用数据对比检测法就可以直接定位数据问题的准确位置，有效地缩小问题的范围，为后续数据检测工作创造良好的开端。

2.2.2 检查出异常或错误数据

对于BSS帐务系统数据检测来说，人工抽样核查存在很多的局限。比如，某月发现当月部分规则可能发生变动，可能是规则修改，也可能是程序变动导致数据出错。为了将问题核查出来，采用数据对比检测法将异常的数据范围确定下来。首先根据数据 $A=[a_1, a_2, \cdots, a_n]$ 的内容和数据检测的目的，建立数据抽样检测规则 $R=[r_1, r_2, \cdots, r_n]$（$R$ 的主要作用是有效地选取到抽样的样本，又由于 R 规定了如何对数据A进行分割，所以按照用户地域或资费标准对数据A进行分割）；其次，确定 R 抽样检测的样本规模，根据概率原理，使得抽样样本满足：表征全部、表征特殊、表征个体等特点。按照规则 R 进行抽样后，取到满足数据检测要求的样本；最后根据抽样样本和业务规则对数据样本进行检查（图3）。

异常数据范围确定下来以后，根据确定好的数据进行分类，再确定需要提取检测的样本。其中，提取抽样检测的数据样本是极为重要的事情，如果提取的样本不全或不够，都会导致检测的失败。一般是按照资费或地域来进行样本分类，每一个点都要抽取100个以上的用户进行抽样检测，整个样本量一般接近1万个。接下来的工作是根据系统的规则进行检查，实现业务检测的程序一般是按单个条件对样本进行准确性检查，而对于满足多个同规则的样本，可以循环调用不同的程序进行不断检测，然后将检测结果存入数据库或文件中。

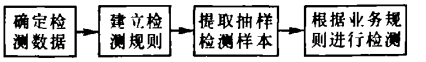

图3 抽样检测过程

3 结束语

本文不是从学术角度研究新的数据检测技术，而是探讨在具体的应用环境下，如何采用适当的数据检测技术更好地完成系统维护工作。我们综合应用BSS帐务系统的几种常用的数据检测技术，提出了一种实现BSS帐务系统日常数据监控和月结出帐数据检测的技术方案。该技术方案在中国联通广西分公司的在线BSS帐务系统已经投入实际应用，能够有效地解决BSS系统帐务数据的准确性和可靠性问题。

参考文献：

[1] 王雷，陈松林，顾学道. 电信行业收入保障体系模型研究[J]. 电信科学，2005，21(11)：2-5.

[2] QB/CU 022-2009. 中国联通公司企业标准，中国联通IT系统BSS系统域综合计费帐务系统业务规范[S]. 2009：81-85.

[3] 傅湘玲，宋茂强. 我国电信运营商收入流失问题分析[J]. 电信科学，2005，21(5)：16-18.

（责任编辑：邓大玉）

（上接第329页）

的。在黑客有足够时间的情况下所有的安全保护都可以被破解，因此，需要不间断研究层次保护中的最薄弱环节和失败安全，不断更新防护方法提升防破解的能力，为规避风险提前做好规划来预防可能发生的破解。

参考文献：

[1] Eric Cronin，Burton Filstrup，Anthony R，et al. An efficient synchronization mechanism for mirrored game architectures：proceedings of the 1st workshop on Network and system support for games[C]. 2002.

[2] Kris Kaspersky. 黑客反汇编揭密[M]. 北京：电子工业出版社，2004.

[3] 白雪梅. 软件保护技术概述[J]. 现代计算机，2009(7)：4-7，15.

[4] Eldad Eilam. Secrets of reverse engineering[C]. Wiley Publishing Inc，2005.

[5] Peter Szor. The art of cinoyter virus research and defense[C]. Addison Wesley Professional，2005.

[6] 周立国，熊小兵，孙洁. 基于自封闭代码块的软件保护技术[J]. 计算机应用，2009(3)：817-822.

[7] John Viega，Gary McGraw. 构建安全的软件——避免产生软件安全问题的正确方法[M]. 北京：清华大学出版社，2003.

[8] Dr Mikhail J Atallah，Eric D Bryant，Dr Martin R Stytz. A survey of anti-tamper technologies[C]. Arxan Technologies Inc，2004.

[9] Michael E Whitman，Herbert J Mattord. 信息安全原理[M]//徐焱，译. 北京：清华大学出版社，2004.

（责任编辑：邓大玉）