yoursunny.P2008.VirusScanner技术报告

IS217《计算机病毒原理》　F0503602 石君霄 5050369043　2008-06-11

1 开发思路

1.1 背景

Windows是一款相当流行的操作系统。Windows的图形界面易于上手，因此很多计算机用户接触的第一个操作系统往往就是Windows。正是由于初学者广泛使用Windows，导致攻击者针对Windows平台编写了各类计算机病毒，影响了系统的正常使用。Windows用户往往会购买各类防病毒软件，以求“解决”病毒对Windows的威胁。

然而，随着计算机病毒编制技术的发展，很多病毒启动后第一件事就是关闭并破坏系统中存在的防病毒软件——方法包括结束进程、篡改文件、修改系统日期至防杀病毒软件注册有效期以外等。防病毒软件被破坏后，病毒就可长驱直入、入驻Windows系统，而此时初学用户就很难再将系统恢复原状了。对于计算机维护人员来说，防病毒软件被破坏后，如果安全模式等也同时被破坏，在使用故障恢复控制台无效的情况下，大概也就只能重装Windows了。

在这种情况下，我提出了一个新思路——交叉杀毒！

1.2 规划

我们知道，由于Linux操作系统在商业市场和中国的家庭并不流行，并且Linux默认的权限管理，针对Linux的病毒相对较少。因此，可以开发一种杀毒工具，它本身运行于Linux操作系统中，而可以对Windows病毒进行查杀。这就是所谓的“交叉杀毒”。

虽然“针对Linux的病毒相对较少”，但是“较少”并不是“没有”，即Linux也会被病毒感染。但是，病毒感染后必须具有持久化的存储，才能够在下一次启动时发挥作用；持久化的存储必须有硬件作支撑，例如存储在硬盘、U盘或引导系统的tftp服务器上。试想，从光盘引导Linux，光盘一旦制作成功，硬件特性就决定了光盘上的Linux可以完全避免被病毒感染。

这个想法不是我第一个提出的。事实上，卡巴斯基杀毒软件已经做出了这样的一个产品——Kaspersky Rescue Disk，它基于Gentoo，可以用光盘启动Linux，并且查杀病毒。正如cnBeta网友的评论那样，它的最大弱点是无法自动更新病毒库，要使用新的病毒库就必须重新刻录光盘；虽然CD-RW可以多次写入，但是每次写入仍然需要消耗很多时间。

我的设计是：光盘启动后，可以通过网络更新病毒库。一次交互过程举例如下：

```
yoursunny.P2008.VirusScanner
Starting up......

Configuring network with DHCP.....Failed
IP Address: 192.168.1.50
```

```
Subnet Mask: 255.255.255.0
Gateway: 192.168.1.1
DNS Server: 208.67.222.222

Looking for harddisk partitions.....
/dev/sda1 label=WINDOWS fs=ntfs size=8GB mount=/media/WINDOWS
/dev/sda5 label=SOFTWARE fs=fat32 size=15GB mount=/media/SOFTWARE
/dev/sdb1 label=MYUSB fs=fat size=64MB mount=/media/MYUSB

Resolving [signature.virusscanner.yoursunny.cn] - 10.0.7.50
Downloading latest virus signatures.....600KB
Uncompressing virus signatures.....35MB

Mounted partitions:
[1] WINDOWS
[2] SOFTWARE
[3] MYUSB
Which partitions do you want to scan? 13

Scanning WINDOWS..... 100%, ETA 00:00:00
Scanning MYUSB..... 3%, ETA 00:01:12

Found 2 virus:
WINDOWS:\windows\svch0st.exe Anomaly-277
MYUSB:\my.exe Scribble
Do you want to clean them [yn]? y

Username: test03
Password:
Resolving [member.virusscanner.yoursunny.cn] - 10.0.7.35
Login to server.....
Account balance: $10.50

Available plans:
[1] 24-hour plan, $1.0
[2] 72-hour plan, $2.0
[3] 1-week plan, $3.0
[4] 3-month plan, $8.0
[5] 1-year plan, $20.0
[6] 10-virus plan, $2.0
[7] 50-virus plan, $6.0
[8] 100-virus plan, $10.0
[9] 1-virus plan, FREE!
Choose a plan: 2

New balance: $8.50
Now you are: 72-hour plan member, until 2008-06.13 20:25

Resolving [cleaner.virusscanner.yoursunny.cn] - 10.0.7.65
Downloading cleaner for Anomaly-277.....
Cleaning virus on WINDOWS:\windows\svch0st.exe
Downloading cleaner for Scribble.....
To clean Scibble on MYUSB:\my.exe, it's required to submit it to the server,
it is okay [yn]? y
Resolving [c4.virusscanner.yoursunny.cn] - 10.0.7.24
Uploading MYUSB:\my.exe.....
Downloading clean version of MYUSB:\my.exe.....
```

```
All work is done, press any key to reboot
```

这个交互过程比较长，主要步骤是：

1. 通过DHCP获取网络参数，如果获取失败则请用户输入
2. 从服务器下载最新的病毒库，解压缩
3. 检测系统中存在的所有硬盘分区和U盘分区，让用户选择要扫描哪些分区
4. 依次扫描选择的各分区，显示估计剩余时间
5. 列出发现的病毒列表，询问是否清除病毒；如果用户选择不清除病毒，结束并重启系统
6. 输入杀毒服务用户名和密码
7. 让用户付费购买杀毒服务，杀毒服务可以根据时间或者病毒种数计费
8. 下载每种病毒的杀毒器，杀掉病毒；部分病毒需要提交到服务器进行杀毒，提交前要提示用户
9. 结束并重启系统

上述过程可以做成gnome图形界面，已熟悉Windows的用户更加容易上手。

2 技术要点

刚才已经介绍了这个程序的整体设计。下面来看点更加实际的东西：演示中实现了什么？我已经编程实现了这个程序中最核心的部分——查毒模块。具体的说，我实现了1个算法和4个程序。

2.1 SMA多模式匹配算法

这个是最重要的部分，用C语言完成。

查毒（不是杀毒）的最基本方法就是模式匹配。模式匹配的本质，就是判定一个大字符串（称为haystack）中是否某个子串（这个子串称为模式pattern）；例如"A quick brown fox jumps over a lazy dog."中存在"brown"而不存在"love"。如果你使用过高级编程语言，你一定已经接触过模式匹配：

- JavaScript：if (haystack.indexOf(pattern)>=0) {...} else {...}
- PHP：if (strpos(haystack,pattern)>=0) {...} else {...}

模式匹配的基本方法是：从大字符串开头开始，逐字符比较与子串是否相等，一旦发现不相等，就把子串后移一位，从头比较。高级点的方法是使用KMP算法：对子串进行一定的分析，以便发现不相等的时候可以多后移一点，效率就提高了。

然而，基本方法和KMP算法，它们都只能对一个子串进行处理，也就是“单模式匹配”。病毒有成千上万种，查毒不能只查一种。你一定会想到另一个方法：foreach (pattern in virus_array) {if (KMP(haystack,pattern)){...}else{...}}。首先，这个循环的方法是可以完成功能的；然后，这个循环的方法是效率极低的——它消耗的时间与病毒种数成正比。

于是，我们要用一定的方法，只扫描haystack一次，而要找出其中包含的所有pattern——这就是“多模式匹配”要完成的。

如果你了解编译器词法分析的原理，你可以把多模式匹配看成一个词法分析器，所有的

病毒特征码都是这个词法分析器接受的单词。这个方法正确，但是实践起来并不容易。词法分析器可以使用GNU flex自动生成，或者用NFA->DFA->规约DFA的方法手工完成。不论采用哪一种途径，都有一个巨大的问题：不是所有的文件都是病毒，而且带毒文件也仅仅有一部分是特征码。例如"A quick brown fox jumps over a lazy dog."，用"brown"和"jumps"两个pattern来匹配，那么词法分析器是否要接受"fox"等单词呢？如果不接受，词法分析会失败；如果接受，你需要接受多少单词呢？因此用词法分析器解决多模式匹配问题并不容易。

还有一种常用的方法是使用树，从树顶开始，根据字符分支，逐级深入。如果匹配失败，则根据一定算法找到下一个节点：匹配到"brown"的w，后面却不是n，则转到owe的w上；如果后面不是e，再转到was的w上，如果仍然不行，只能回到树根上。

学术论文《基于有序二叉树的多模式匹配算法》提出了另一种方法：使用二叉树结构代替树结构，大大降低内存占用，而匹配速度基本一致；这个方法的原理较为复杂，请参见论文。论文中给出了一些伪代码，遗憾的是，这些伪代码存在诸多错误，直接写成程序是无法运行的。论文作者在某光盘上给出了一份C++语言的代码，但是需要Visual C++开发环境，而VirusScanner是运行于Linux的程序，因此也无法使用；不过这份C++代码虽然几乎没有注释，但是其中错误较少，可以参考。

经过本人的不懈努力，成功用C语言实现了SMA二叉树多模式匹配算法。该算法的接口是sma.h，接口主要内容有：

节点数据结构

```
//SMA算法二叉树节点
typedef struct smaNode smaNode;
typedef smaNode* smaState;
struct smaNode {
    unsigned int id;
    unsigned int Lchar;
    smaState Lchild;
    unsigned int Rchar;
    smaState Rchild;
  smaState Parent;//SMA意义上的父节点
    smaState Fail;
  smaUserData value;//用户数据
};

typedef smaNode* smaTree;
```

节点内存管理操作函数

- 创建新的SMA有序二叉树节点：smaNode* smaNode_make();
- 释放SMA有序二叉树节点及其所有后代内存：void smaNode_destroy(smaNode* n);

SMA算法基本计算函数

- 根据访问规则从状态p出发到子状态child时，若栈中有且只有ch，则返回child：smaState sma_goto(smaState p, unsigned int ch);

- 计算状态深度：int sma_StateDepth(smaState p);

SMA二叉树构建函数
- 向有序二叉树添加一个pattern：void sma_AddPattern(smaTree tree, const char* pattern, int len, smaUserData value);
- 创建节点编号：void sma_SetId(smaTree tree);
- 创建失败指针：void sma_BuildFail(smaTree tree);

持久化函数
- 将有序二叉树输出为XML流：void sma_dumpXML(smaTree tree, FILE* fd);
- 注：XML输出目前只能用于查看，因为没有实现从XML读取二叉树的函数
- 将有序二叉树输出为文本流：void sma_dumpText(smaTree tree, FILE* fd);
- 从文本流读取有序二叉树：smaTree sma_loadText(FILE* fd);

匹配查找函数
- 查找回调函数：typedef void (*sma_callback)(int index, smaUserData value, void* data);
- 从输入字符串中查找有序二叉树中的模式：void sma_search(smaTree tree, const char* haystack, int len, sma_callback cb, void* data);
- 从输入文件中查找有序二叉树中的模式：void sma_search_file(smaTree tree, FILE* haystack, sma_callback cb, void* data);

2.2 credo

credo并不是查毒工具的必要组成部分。它的目的是将开源查毒软件OpenAntiVirus的病毒库clamav.strings转换成smaconv可以识别的格式，同时创建一份“模式id=>病毒名称”的列表。这个程序用C语言完成。

目前只能转换clamav.strings，也就是单模式病毒库。OpenAntiVirus的另一个病毒库clamav2.strings定义了一个病毒需要依次匹配多个模式，但是credo并不支持这种病毒库。

credo的使用方法：下载VirusSignatures-latest.zip，解压出其中的clamav.strings，然后在解压后所在目录运行credo程序，当前目录中即会生成pattern-list.txt和virus-name.txt两个文件。
- pattern-list.txt：作为smaconv的输入，依此创建SMA二叉树
- virus-name.txt：“模式id=>病毒名称”的列表，若要显示病毒名称是肯定需要它

OpenAntiVirus的病毒库，现在看到的更新日期是在2004年，也就是已经好长时间没有更新了。查毒工具的重中之重就是病毒库，但是作为技术研究，用一个较老的病毒库同样可以达到演示目的。

2.3 smaconv

smaconv的作用是把形如pattern-list.txt的文件转换成SMA二叉树，用C语言完成。

pattern-list.txt的格式是：每行一条记录，模式id=模式十六进制串，例如：

```
fcb=58354F2150254041505B345C505A58353428505E2937434329377D2445494341522D53544
14E4441
```

smaconv读入这样的文件，然后根据其中的记录构建SMA二叉树，最后输出成XML和文本文件。

smaconv的使用方法：smaconv pattern-list.txt sma.txt sma.xml

2.4 scanner

scanner是唯一需要发布给最终用户的程序。它读入sma.txt，然后根据其中的信息还原SMA二叉树，然后对各个文件作匹配。这个程序用C语言完成。

scanner有三种使用方法：

- 在命令行参数指定被扫描文件：scanner sma.txt file1.ext file2.ext ...
- 从文件指定被扫描文件：scanner sma.txt @list，list文件中包含被扫描文件的路径、每行一个
- 从标准输入流指定被扫描文件：scanner sma.txt @-，标准输入流包含被扫描文件的路径、每行一个

推荐使用第三种方法。例如要扫描/media/MYUSB的所有文件，可以用命令：

```
find /media/MYUSB -xtype f | tee list | scanner sma.txt @- > result
```

scanner会向标准输出流输出发现的所有特征信息，格式为：每行一条记录，“文件序号(从0开始),发现位置,特征id”。

这个程序对于查毒工具的性能至关重要。这个程序需要从文件读入数据，而系统调用引起的映像切换会消耗大量时间。读文件有三种方式：

- 使用fgetc逐字节读取：每次读一个字节就需要一次系统调用，效率很低
- 使用fread一次性读取一块：系统调用次数减少，但是管理缓冲区并不容易，而且需要对SMA算法库进行修改
- 使用mmap将文件映射到进程地址空间：我选择了这种方法，程序只需从“内存”取数据，操作系统会自动在缺页中断时加载所需的文件块；不但效率高，而且不需要程序管理缓冲区。这种方法的缺点是无法扫描大文件，因为我是将整个文件映射到地址空间，而地址空间的大小是有限的

2.5 result.sh

result.sh是一个shell脚本，读入scanner输出的结果，然后在被扫描文件列表、病毒名称列表中提取出病毒名称。

由于这是使用shell脚本实现的，它只能适合发现了很少几个病毒的情况，如果发现了大量病毒、scanner输出了一份很长很长的结果，那么它需要耗费大量的时间。

这个脚本仅仅用于测试。如果要制作杀毒光盘，应该使用更为高效的方法重写这个脚本，例如Python。

result.sh的使用方法：result.sh < result > result.txt

3 程序测试

测试环境：Ubuntu 8.04 hardy，kernel 2.6.24-16-generic，gnome 2.22.1；奔腾M 1.70GHz，512MB内存。

先编译上面三个程序，可执行文件复制到同一目录，并准备好clamav.strings病毒库。准备一些病毒样本和正常文件在sample/目录中，我准备了3827个文件，总共13.0MB，处于ext3文件系统中。

第一步，转换OpenAntiVirus病毒库

clamav.strings大小为2.6MB，有21096行。

运行命令：./credo

计时结果：real　0m0.089s，user　0m0.052s，sys　0m0.024s（用time命令计时）

产生文件：

pattern-list.txt，2.2MB、21096行

virus-name.txt，467.8KB、21096行

第二步，创建SMA二叉树

运行命令：./smaconv pattern-list.txt sma.txt sma.xml

计时结果：real　0m7.256s，user　0m5.672s，sys　0m0.312s

产生文件：

sma.txt，32.9MB、1028296个状态

sma.xml，74.6MB

注：sma.txt文件比pattern-list.txt大得多，但是可以对病毒库保密；尝试对sma.txt进行bz2压缩，9.1MB

第三步，扫描病毒

运行命令：find ./sample -xtype f | tee list | ./scanner sma.txt @- > result

计时结果：real　0m22.107s，user　0m11.869s，sys　0m0.524s

产生文件：

list，123.3k、3766行，被扫描的文件列表

result，118.5k、9013行，发现了9013个特征码

第四步，整理扫描结果

运行命令：./result.sh virus-name.txt list < result > result.txt
计时结果：real　3m31.299s，user　1m51.827s，sys　1m3.812s
产生文件：
result.txt，519.1k、9013行，包含result文件的所有内容，加上带毒文件名称、病毒名称
这个文件是可以人工阅读的，节选如下：

```
1845,./sample/atozvirus/k/KILLER1.COM,1,3cc1,VGEN.153.0 (Clam)
1846,./sample/atozvirus/k/KIT.COM,586,1b6a,Kit.1
1846,./sample/atozvirus/k/KIT.COM,1181,1b88,KitG
1846,./sample/atozvirus/k/KIT.COM,2254,1b6b,Kit.2384 (Clam)
1847,./sample/atozvirus/k/KLAEREN.COM,7,2b34,Small.130-gen (Clam)
1847,./sample/atozvirus/k/KLAEREN.COM,5946,1b97,Klaeren
1848,./sample/atozvirus/k/KHIZ751.COM,2013,184b,Infector-624.726
1849,./sample/atozvirus/k/KENNEDY.COM,63,1ae8,Kennedy.4
1849,./sample/atozvirus/k/KENNEDY.COM,97,1ae7,Kennedy.3
1849,./sample/atozvirus/k/KENNEDY.COM,120,1ae6,Kennedy.1
1850,./sample/atozvirus/k/KUKAC-NA.COM,68,3894,Turbo-448.A
1850,./sample/atozvirus/k/KUKAC-NA.COM,365,3895,Turbo-448.B
1851,./sample/atozvirus/k/KING1424.EXE,1428,280a,Satan 3.00
1852,./sample/atozvirus/k/KUKAC448.COM,68,3894,Turbo-448.A
1852,./sample/atozvirus/k/KUKAC448.COM,365,3895,Turbo-448.B
1853,./sample/atozvirus/i/IRONMAID.COM,7,2b34,Small.130-gen (Clam)
1853,./sample/atozvirus/i/IRONMAID.COM,636,18ca,Iron Maiden.1
1853,./sample/atozvirus/i/IRONMAID.COM,636,18cb,Iron Maiden
1854,./sample/atozvirus/i/ICELANDC.EXE,127,201f,Mix-664
1854,./sample/atozvirus/i/ICELANDC.EXE,169,1800,Icelandic-Saratoga
1854,./sample/atozvirus/i/ICELANDC.EXE,249,17fc,Icelandic I
1855,./sample/atozvirus/i/ICEMIX1B.EXE,8929,2021,Mixer-1B
1855,./sample/atozvirus/i/ICEMIX1B.EXE,9227,2023,Mix-I
1856,./sample/atozvirus/i/INT1319C.EXE,814,189b,Intruder.1
1856,./sample/atozvirus/i/INT1319C.EXE,987,189e,Intruder.4
1857,./sample/atozvirus/i/IV-APRIL.EXE,2571,18db,Italian.8 (Clam)
1858,./sample/atozvirus/i/INCOM.COM,50,1824,IKV-512
1859,./sample/atozvirus/i/ITTI-MAL.COM,4,18ed,Itti-Malmsey
1860,./sample/atozvirus/i/INTN1381.COM,230,188e,Internal.2
1860,./sample/atozvirus/i/INTN1381.COM,636,188b,Internal.1
1860,./sample/atozvirus/i/INTN1381.COM,653,188c,Internal-1381
```

可见，查毒工具对病毒种类的识别基本正确；sample目录里还有一份Nachos源代码作为正常文件，它没有被识别为病毒。

最大的缺点是，这个脚本太慢了！不过这次扫描的目标包括病毒样本包从而有大量病毒，而平时的扫描不会是这样的情况。

效率的一个对比

连接上我的移动硬盘（USB2.0、NTFS分区），使用上面的音乐目录作为样本，并且放入一个eicar测试病毒。总共88个文件、305.8MB。

运行命令：find /media/M1/music -xtype f | tee list.m1 | ./scanner sma.txt @- > result.m1

计时结果：real　10m36.176s，user　8m47.537s，sys　0m1.864s

运行命令：./result.sh virus-name.txt list.m1 < result.m1 > result.m1.txt

计时结果：real　0m0.073s，user　0m0.024s，sys　0m0.016s

产生文件：

result.m1.txt，内容只有一行：

```
0,/media/M1/music/eicar,2,fcb,Eicar-Test-Signature
```

查毒的正确性得到了验证：音乐文件没有病毒，只有一个eicar是病毒。效率方面，19分钟显得有点慢，但作为自己写的一个原型，就不要苛求了。

4 结论

yoursunny.P2008.VirusScanner使用SMA多模式匹配算法正确实现了特征码查病毒功能。