逆向工程在分析恶意代码通信地址中的应用

吕少阳

（东南大学计算机科学与工程学院 ，南京， 210000 ）

摘　要：为了分析僵尸网络控制器的地址，本文提出了利用逆向工程分析恶意代码通信地址的方法。本文针对现有利用沙盒分析恶意代码通信地址的方法存在的不足提出了逆向工程分析。首先介绍了利用逆向工程分析恶意代码通信地址的优势及原理和所使用工具。随后提出了逆向工程分析恶意代码通信地址的两种方法，介绍了两种方法的特点、适用情况以及技术难点和实现方法。最终利用两种方法分别对恶意代码样本进行了实验分析，结果表明两种方法都成功分析出了恶意代码样本通信地址。

关键词：逆向工程；恶意代码；僵尸网络；静态分析；动态分析

Analysis of Communication Address of Malicious Software Using Resverse Engineering

Lv Shaoyang

(School of Computer Science and Engineering, Southeast University, Nanjing, 210000)

Abstract: To get the addresses of botnet controllers, the method of analysis of communication address of malicious software using reverse engineering is given in this paper.For making up the insufficiency of using Sandbox to analyse the communication address of malware ,this paper suggest using reverse engineering to do that.First,the advantage and the theory of using reverse engineering to Analyse the communication address of malware and the the tools of doing that are introduced.Second,two method of analysis of communication address of malicious software using reverse engineering are given.The feature and the applicable case and the difficulty of two method are also given .Based on the two method,a malware sample is analysed experimentally .Experimental results manifest that the two methods are useful.

Key words: Reverse Engineering; Malware; Botnet; Static Analysis; Dynamic Analysis

CNCERT / CC 对僵尸网络的定义是僵尸网络是攻击者利用互联网秘密建立的可以集中控制的计算机群．其组成通常包括被植入僵尸程序的计算机群、一个或多个控制器及控制者的控制终端等。恶意代码一方面通过各种途径感染网络上的大量主机，另一方面主动连接控制器并接受黑客控制者的各种命令。对抗僵尸网络最有效的方法是封杀控制器的地址使得僵尸主机无法连接到控制器从而杜绝控制者向僵尸网络发送命令的渠道。目前我们的网络安全实验室已经可以通过“蜜罐”（ honeypot ）的手段捕获黑客向僵尸网络投送的恶意代码样本，通过分析这些代码的通信地址就可以得到僵尸网络控制器的地址，这就为防止僵尸网络被黑客控制者利用提供了可能性。

目前得到恶意代码通信地址最直接的方法是对其进行沙盒（ sandbox ）分析。主要过程是将需要分析的恶意代码程序放入沙盒环境运行，监视并记录其在沙盒环境中的行为，从而得到恶意代码试图连接的通信地址。该方法实现原理较简单，但是由于程序在沙盒中运行过程所展示的功能的局限性，因此通过沙盒分析不一定能得到的恶意代码的所有通信地址。另外，目前的沙盒分析多依靠第三方分析平台实现，而第三方分析平台分析水平参差不齐，无法保证分析的准确性。

逆向工程最初被用于对硬件的分析，它的定义为：分析他人设计的硬件产品，生成产品规格说明的活动，通常用于修正原产品中的缺陷或增强原产品的功能。软件逆向工程就是在没有程序源代码的前提下对软件进行破解、分析，从而得到软件的功能、运行流程和作者的思想和设计理

念。

逆向工程在对恶意代码的分析中有着先天优势。由于逆向工程从分析恶意代码的二进制代码出发，研究恶意代码的运行机理，因此得到的信息最多，避免了通过沙盒分析恶意代码通信地址时可能会遇到的结果不完整的情况。

1 逆向工程分析方法

1.1 逆向工程方法分类

1.1.1离线代码分析（静态反汇编）

利用反汇编工具对二进制代码本身进行分析，从而得到原机器代码中有用信息。优点：可提供程序整体框架结构，易于查找感兴趣的代码和函数。缺点：无法看到程序处理的数据流，因此需更好的理解代码并猜测代码的执行情况。

1.1.2现场代码分析（动态跟踪调试）

利用动态调试工具对恶意代码程序进行动态跟踪调试，根据跟踪程序本身的执行过程中的各种现场信息分析得到有用数据。优点：可观察到程序内部数据及数据对代码流的影响，因此可以提供更多信息。另外一些运行时才能脱壳的或者代码被加密、压缩的程序只有使用现场代码分析。缺点：需要实现通过离线代码分析进行配合来理解程序的结构。

1.2 逆向工程分析恶意代码通信地址原理

恶意代码通信地址由域名和IP地址组成。通常情况下，域名以明文或加密的形式存储在恶意代码程序中。在与控制器通信前，恶意代码通过对域名的解析得到控制器IP地址，使用该IP地址与僵尸网络控制器进行通信。逆向工程在分析通信地址中主要使用两种手段：在程序代码中查找所使用的域名；跟踪程序的执行过程中的网络通信过程从而得到通信IP地址。对应的方法分别是静态分析和动态跟踪调试分析。

逆向工程分析恶意代码的一般流程为：查壳、脱壳 >> 静态分析 >> 动态跟踪调试分析。对恶意代码的查壳与脱壳不在本文的讨论范围内，本文只讨论静态分析与动态跟踪调试分析部分。

1.3 软件逆向工程使用的工具

（1 ）反汇编器：以程序的可执行二进制代码为输入，生成包含整个或部分程序汇编语言代码的文本文件，并带有一定的辅助分析功能。反汇编器是静态分析使用的主要工具，并且在动态分析的前期也起到基础分析功能。

（2 ）调试器：允许用户在程序运行的同时观察程序，并允许用户选择程序中任何位置的某个函数或代码行设置断点并进行跟踪。

（3 ）系统监视工具：辅助逆向分析，用于寻找、监控、研究以及剖析待被逆向的程序。

2静态分析

通常情况恶意代码会将通信域名存储在二进制文件中，而通信IP地址只有在通信前才会通过解析域名的方式获得。静态分析针对恶意代码的二进制文件进行分析，因此静态分析技术通常用于分析恶意代码的通信域名。首先利用反汇编器对恶意代码进行反汇编得到汇编代码，其次汇编代码中以人工识别的方法搜寻可能为通信地址的字符串，最后对可能为通信域名的字符串在代码中的使用情况展开分析，最终挑选出用途为通信域名的字符串。该方法实现原理较为简单，但是由于需要人工识别可能为通信域名的字符串，因此该方法只能用于处理通信地址以明文形式存储的情况。对于通信地址加密存储或者分段存储的情况，静态分析的人工识别难度较大，这种情况只能用动态分析法来解决。

2.1 技术难点

静态分析恶意代码通信地址时遇到的技术难点主要有两个方面：

(1) 人工识别代码中的字符串

恶意代码中含有大量的字符串，包括程序中定义的变量名、函数名以及字符串都是以字符串的形式存储在二进制代码中，数量有可能较大。如果逐一对这些字符串进行人工识别，工作强度会非常大。

各种反汇编器的辅助分析功能参差不齐，在选择反汇编器时可以有针对性选择字符串分析功能较强大的反汇编器。利用反汇编器自动挑选出所有存储在恶意代码程序文件中的字符串，并且过滤掉变量名、函数名等干扰字符串。与此同时

还可以利用反汇编器的字符串显示长度限制功能限制反汇编器显示出来的字符串长度。设定反汇编器仅显示长度大于一定值的字符串。由于大部分域名的长度都大于 8 个字节，因此本文的例子中限制反汇编器显示的字符串长度大于 8 个字节。经过前面两方面的限制后，需要人工识别的字符串已经大为减少。

(2) 验证通信域名字符串

通过人工识别的方法找到的域名字符串并不一定就是恶意代码使用的通信域名，因此如何对其进行验证就是一个重要问题。针对这个问题，解决的方法主要是查找字符串在程序代码中的使用情况。一方面通过反汇编器的辅助分析功能查询字符串是否被程序代码调用，另外对被程序所调用的字符串进一步验证其是非被用于当做域名解析使用。

一般情况下恶意代码都是通过 API 函数 struct hostent* FAR gethostbyname(__in const char *name) 来进行域名解析，因此如果字符串被当做 gethostbyname （）的参数“ name”使用，就可以判断该字符串为通信域名。

2.2 工具介绍

目前常用的反汇编器主要有 W32Dasm 、 IDE Pro 、 C32asm 、 DUMPBIN 等。其中 IDA Pro 作为目前功能最强的反汇编器被众多反汇编爱好者使用，它除了具有一般反汇编器所具有的反汇编功能外还提供了非常强大的辅助分析功能。 IDA Pro 可以自动识别大部分库函数和局部变量以及静态分析通信地址所关心的字符串数据，并将整个程序的流程以非常直观明了的流程图的形式表示出来。 IDA Pro 还可以自动对函数调用时的参数进行分析，很有利于验证通信域名字符串。由于 IDA Pro 分析功能过于强大，造成其分析速度较慢，对于大小达到数 MB 的程序，分析时间要达到若干分钟。不过对于一般大小只有几十 KB 或者几百 KB 的恶意代码来说， IDA Pro 的分析速度劣势可以忽略。因此 IDA Pro 在静态分析恶意代码通信地址上具有明显优势。

2.3 实例分析

待分析恶意代码： Backdoor.Hacarmy.D ，该恶意代码为 PE 结构的 win32 可执行程序，功能为木马被控端。

分析工具：IDA Pro 5.5。

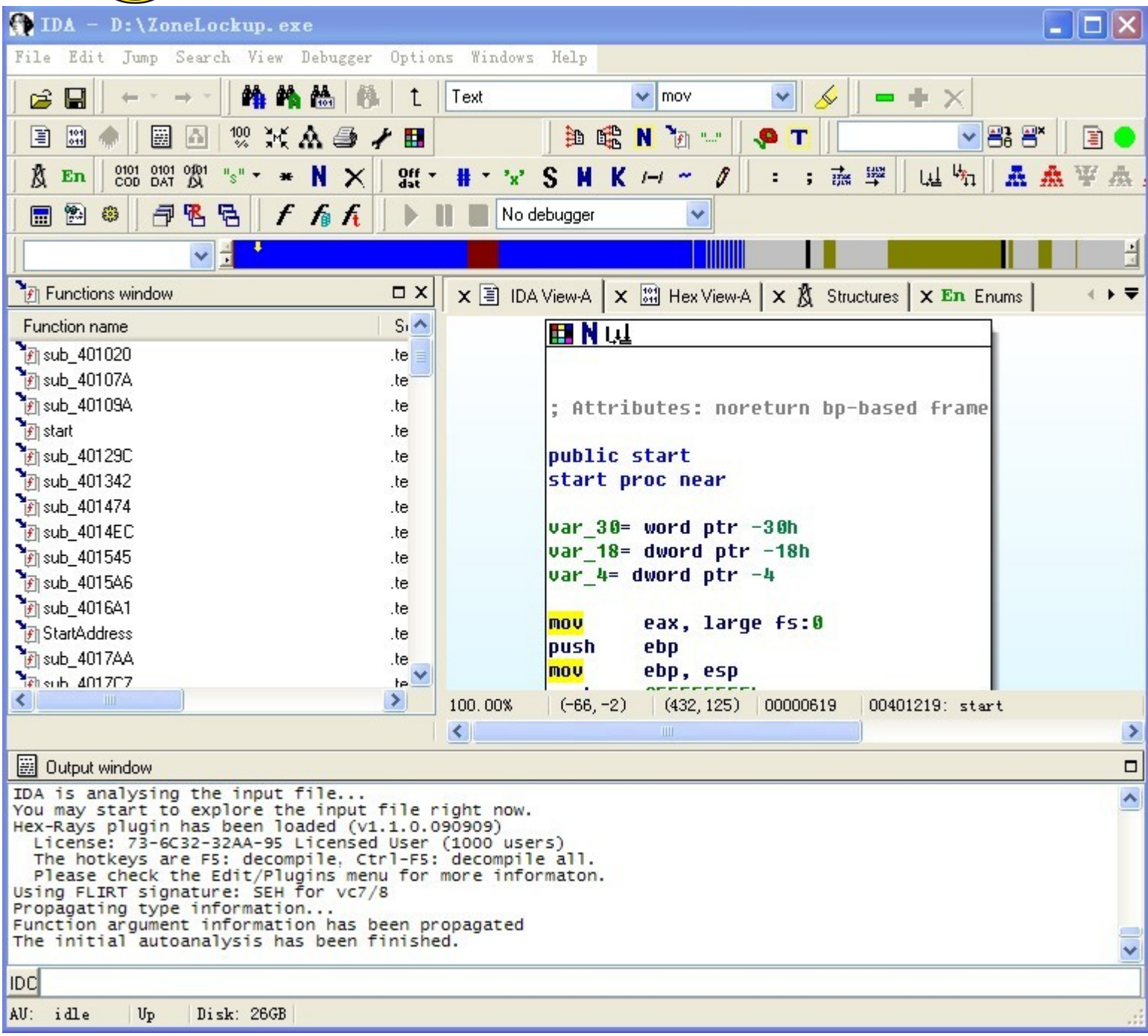

图 1 IDA Pro 预分析结果界面

本例的分析中，在 IDA Pro 中的Strings长度限制选项中设置字符串的最小长度为 8 字节。图 2 为 IDA Pro 的字符串分析结果，经过人工识别分析，只有字符串“ g.hackarmy.tk”符合通信域名特征。

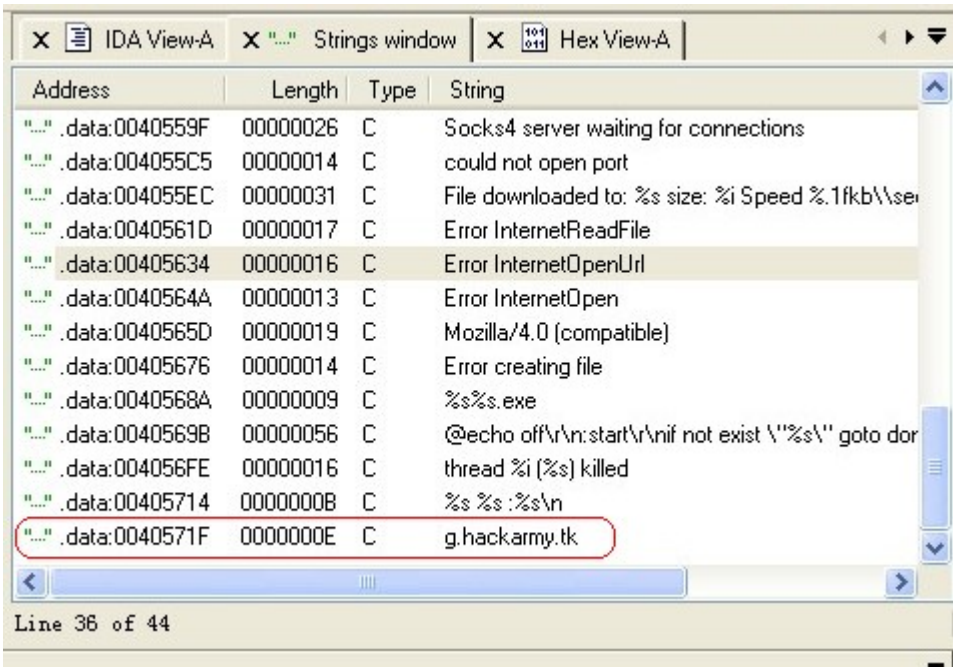

图 2 Strings 分析结果

图 3 所示为字符串“ g.hackarmy.tk”的使用情况。可以看出，该字符串共被使用两次，而且都是作为 API 函数 struct hostent* FAR gethostbyname(__in const char *name) 的参数“ name”使用。就此可以判断出域名“ g.hackarmy.tk”为被分析恶意代码对象的域名通信地址。

```
.data:00405104 name            dd offset aG_hackarmy_tk ; DATA XREF: sub_402621:loc_402954↑r
.data:00405104                                          ; sub_402621+346↑r
.data:00405104                                          ; "g.hackarmy.tk"
.data:00405108                 db    0
.data:00405109                 db    0
.data:0040510A                 db    0
.data:0040510B                 db    0
.data:0040510C ; __int16 dword_40510C
.data:0040510C dword_40510C    dd 1A0Bh
.data:00405110 aMorris         db 'morris',0
.data:00405117 ; char aRaw[]
.data:00405117 aRaw            db 'raw',0
.data:0040511B ; char a?dontuseme[]

loc_402954:                                             ; CODE XREF: sub_402621+325↑j
                       cmp     name[edi*4], 0
                       jnz     short loc_402960
                       xor     edi, edi

loc_402960:                                             ; CODE XREF: sub_402621+33B↑j
                       push    dword_40510C[edi*4] ; __int16
                       push    name[edi*4]     ; name
                       call    sub_4029B1
```

图3 字符串“ g.hackarmy.tk”调用情况

3动态分析

动态分析技术建立在使用静态分析技术的基础上，需要分析人员首先通过静态分析对分析对象的代码有大致的了解，找到恶意代码通信部分的关键代码，跟踪关键代码的运行。通过观察堆栈、寄存器、函数变量及返回值等关键数据的变

化而得到有用信息。具体在分析恶意代码通信地址时，需要找到通信函数并跟踪函数的调用使用的参数，通过对参数的分析得到通信地址。

该技术实现原理较为复杂，但是由于直接在代码运行过程中进行分析，如果找到关键代码或函数，则只需关注这部分代码的运行情况，分析效率较高。

3.1 技术难点

静态分析恶意代码通信地址时遇到的技术难点主要有两个方面：

（1 ）关键函数的识别及定位

分析恶意代码的侧重点不同，所需要关注的函数就不同。对于动态分析恶意代码的通信地址来说，需要关注的函数是域名解析函数和网络主动连接函数。通过这两种函数的跟踪可以分别得到通信域名和通信IP地址。一般恶意代码所使用的域名解析函数是 WS2_32.dll 中的 gethostbyname （），该函数已在之前说明。恶意代码使用WS2_32.dll中的 connect （）与控制器建立连接。在知道了需要关注的函数后，就需要在程序中找到这些函数的调用点，在调用点上设置断点，以便下一步分析其运行参数。这部分工作需要反汇编器的辅助分析功能来做，通过反汇编器将程序调用到的说有库函数列出，找到 gethostbyname （）与 connect （），并进一步找到这两个函数的所有调用点设置断点。

（2 ）函数运行参数的分析

在找到关键函数并设置好断点后，便遇到了动态分析最重要也是最复杂的一步，即分析函数的运行参数得到通信域名与通信IP地址。函数在运行时，参数的传递方式多种多样，对于 gethostbyname （）与 connect （）来说，参数是通过从右至左的顺序以压栈的方式传入函数的。

对于struct hostent* FAR gethostbyname(__in const char *name) 函数来说，只有一个参数，因此栈内从栈顶开始前 4 字节的内存空间存储的就是参数“ const char *name”的值，而变量 name 存储的是域名字符串的地址，进一步查询 name 的值指向的内存块中的数据即可得到通信域名字符串。

对于 int connect(int sockfd, const struct sockaddr* server_addr, socklen_t addrlen)函数。参数 sockaddr 在IPv4下的结构如下：

```
struct sockaddr {
    ushort  sa_family;
    char    sa_data[14];
};
```

其中 sockaddr 结构的内存数据中前两个字节存储变量sa_family值，第 3 、 4 字节即sa_data的前两字节存储端口号，第 5-8 字节存储IP地址，该IP地址就是恶意代码通信IP地址。因此该函数运行前，栈内从栈顶开始 5-8 字节内存储的是结构体 sockaddr 的地址，进一步通过该地址可以再内存中定位到结构体 sockaddr 的内容，再进一步查找结构体 sockaddr 内 5-8 字节数据就可以得到所需恶意代码通信IP地址。

3.2 工具介绍

动态分析技术使用的主要工具是调试器，其作用是在程序运行的同时观察程序。调试器的基本功能是设置断点与代码跟踪，同时带有部分较基础的代码辅助分析能力。目前主要的调试器有 SoftICE 、 OllyDbg 、 TRW2000 等。 TRW2000 由于不支持Windows2000以后的操作系统，目前已经很少有人使用。 SoftICE 的调试级别为Ring 0级，属于内核级调试工具，功能非常强大，很适合用于驱动程序的开法。但由于其属于内核级调试器，所以容易在调试过程中造成系统出错，而且操作较为复杂，不适合普通用户级程序调试。 OllyDbg 属于用户级调试工具，功能不如 SoftICE 强大，但是使用简单明了并且对库函数的识别率较高，设置断点和对函数参数的分析都很方便，非常适合用于对恶意代码通信地址的分析。

3.3 实例分析

待分析恶意代码：同样实用恶意代码 Backdoor.Hacarmy.D 作为分析目标。

分析工具： OllyICE 。 OllyDbg 的一个版本，增加了部分实用插件。

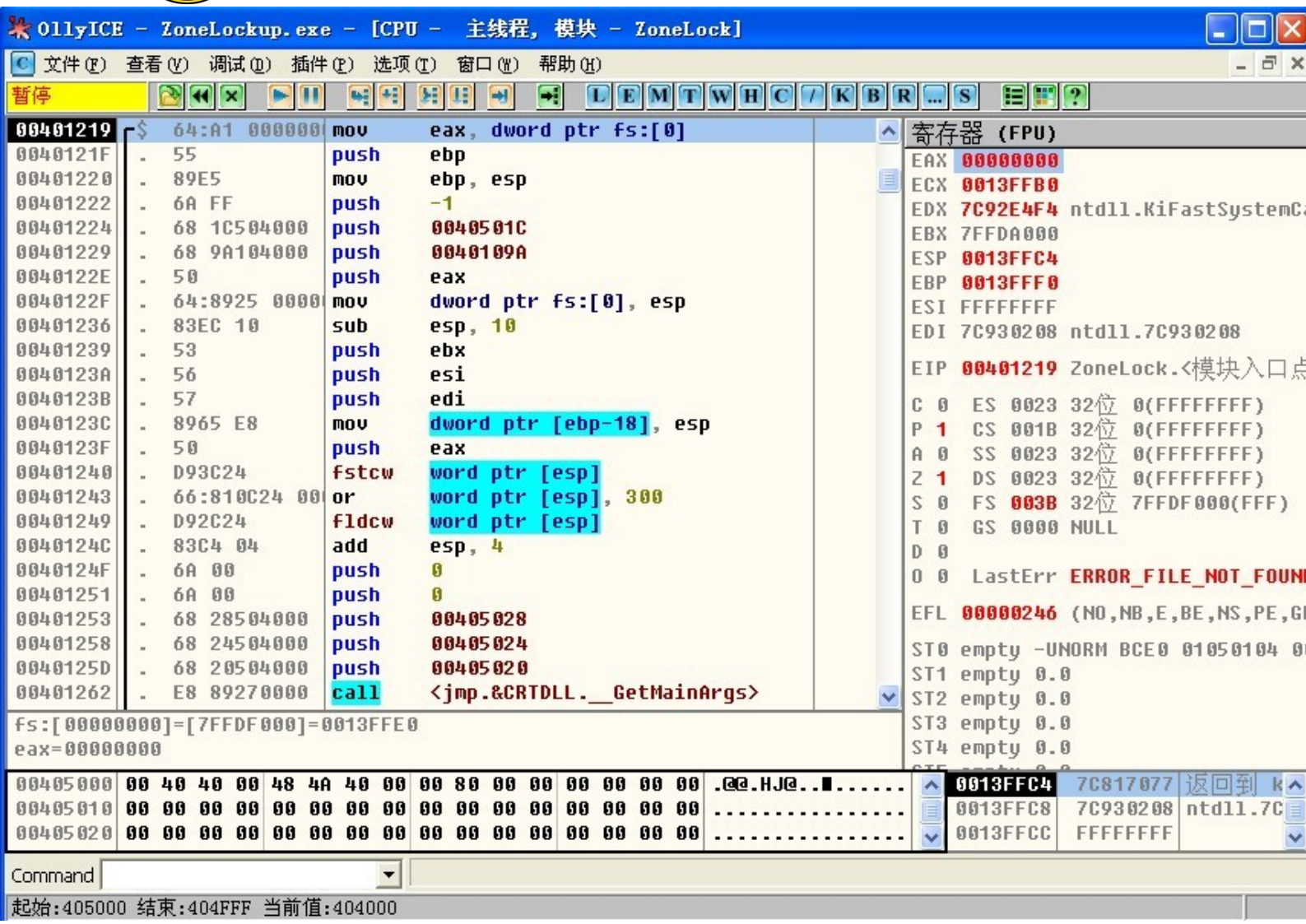

图 4　　程序入口点

对于本例而言， gethostbyname （）函数的调用点地址为004012F7H， connect （）的调用点地址为0040132CH。利用 OllyICE 可以对这两个函数的调用点分别设置断点。

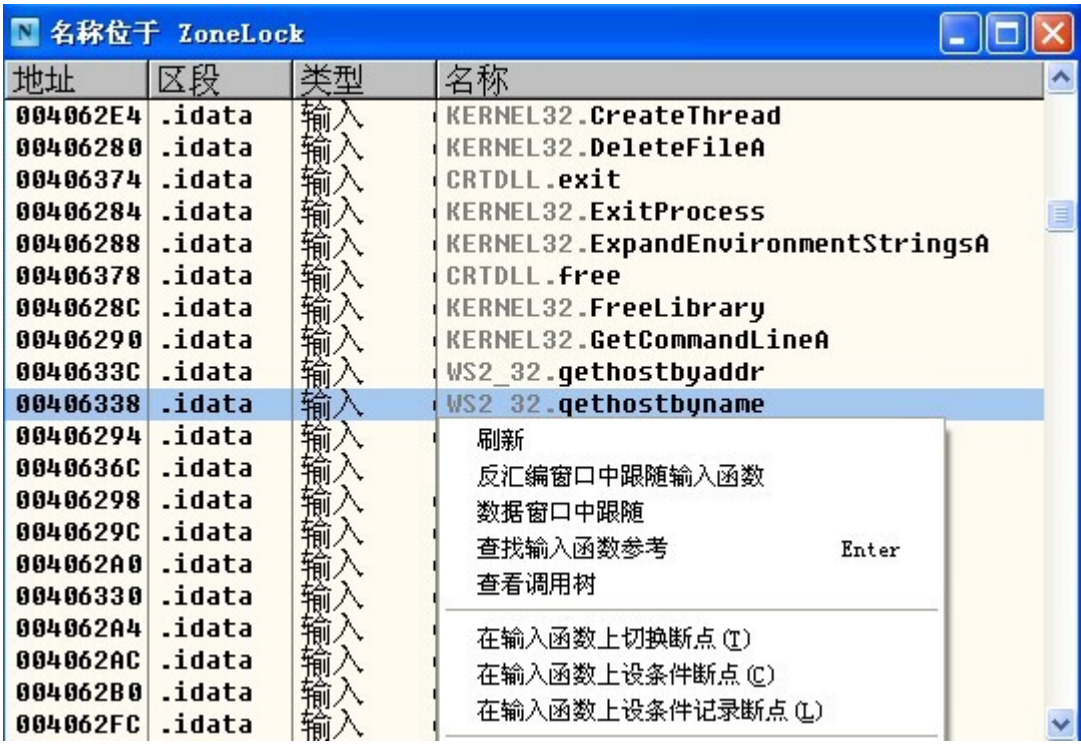

图5 主模块函数调用表

图 6 断点设置

在本例中， gethostbyname （）调用点的断点处如图 7 所示，栈顶地址为 0013F594H，存储的数据为 0040571FH，因此指针参数“ name”的值为 0040571FH，而地址为 0040571FH 的内存单元中的数据内容为“ g.hackarmy.tk”。该域名很有可能即为恶意代码所用的通信域名。

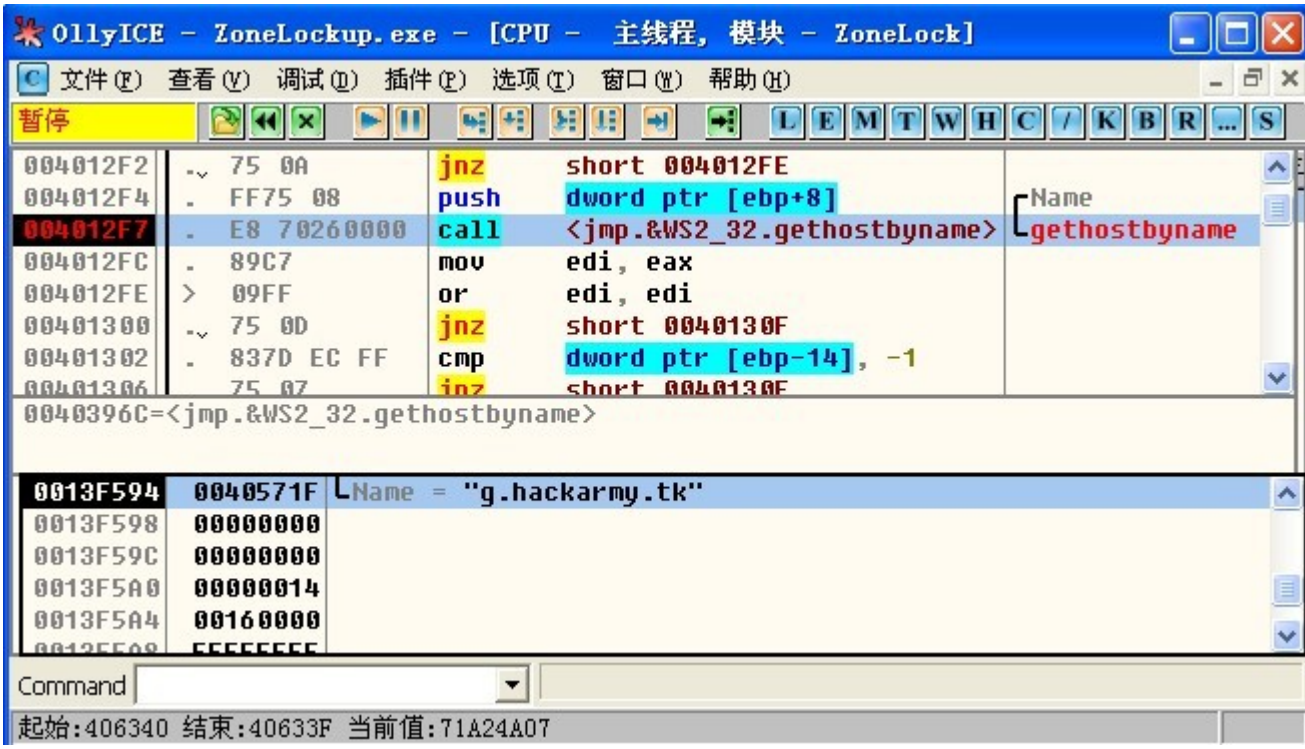

图7 gethostbyname （）调用断点信息

connect （）调用点的断点处如图 8 所示，栈中第 5-8 字节存储的 sockaddr 结构体地址参数为 0013F5ACH 。通过对地址 0013F5ACH 的跟踪得到 0013F5AEH-0013F5AFH 内的数据为 1A0BH ，对应10进制数为6667。 0013F5B0H-0013F5B3H内的数据为2EH、89H、63H、38H,对应十进制数为46、 137 、99、56。因此 Connect （）在本例的恶意代码的调用中使用的地址参数即IP地址为 46.137.99.56 ， 端口为 6667。

为了验证 46.137.99.56 是否是域名 g.hackarmy.tk 对应的IP地址，本例中用Ping命令对 g.hackarmy.tk 域名进行了解析，结果为 46.137.99.56 。因此可以确定域名 g.hackarmy.tk 与IP地址 46.137.99.56 为该恶意代码通信地址。

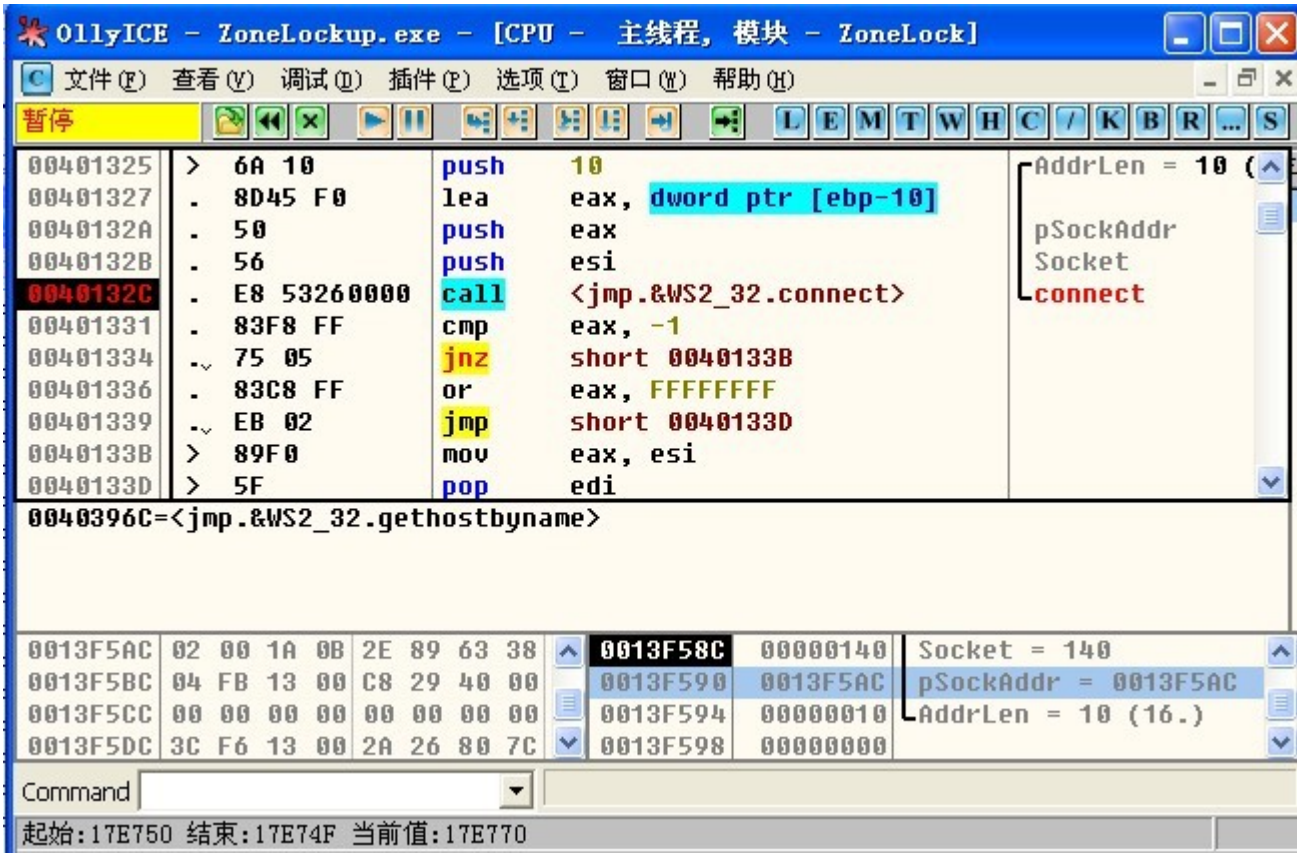

图8 connect （）调用断点信息

4结论

逆向工程在分析恶意代码通信地址方面具有先天优势。相对沙盒分析而言，逆向工程从程序的汇编代码出发，分析的更底层、更透彻。静态分析技术适用于反逆向手段不高的恶意代码，通过简单的字符串搜索可以方便的查找到通信所用的域名，但对于通信地址经过加密存储的恶意代码来说则不适用。动态分析技术绕过了程序本身对通信地址部分的加扰部分，直接跟踪到通信地址的最终使用点上进行分析。动态分析需要找到关键的函数调用点，相对静态分析而言，实现难度更高，但是分析结果准确率也更高。

参考文献：

[1]

,Eldad Eilam. 逆向工程揭密 [M]. 北京，电子工业出版社， 2007.

[2]
段钢 . 加密与解密（第二版） [M]. 北京，电子工业出版社， 2004.

[3]
Kris kaspersky. 黑客反汇编揭秘 [M]. 北京，电子工业出版社， 2004.

[4]
段冬燕 . 基于主机的恶意代码防范方法 [J] , SCIENCE & TECHNOLOGY INFORMATION， 2010， 24 .

[5]
冉宏敏，柴胜，冯铁，张家晨 .P2P 僵尸网络研究 [J]. 计算机应用研究， 2010， Vol.27 ， No.10.

[6]
梁晓 . 恶意代码行为自动化分析技术探析 [J] ， 2010， Vol.17 ， No.8.

[7]
刘颖 .Windows 环境恶意代码检测技术研究 [D]. 电子科技大学， 2006.