



Как вредоносное ПО обходит песочницы

ЭВОЛЮЦИЯ МЕТОДОВ

ptsecurity.com

Ольга Зиненко

Старший аналитик информационной безопасности
ozinenko@ptsecurity.com

Алексей Вишняков

Руководитель отдела обнаружения вредоносного ПО, PT ESC
avishnyakov@ptsecurity.com

Сегодня поговорим о...

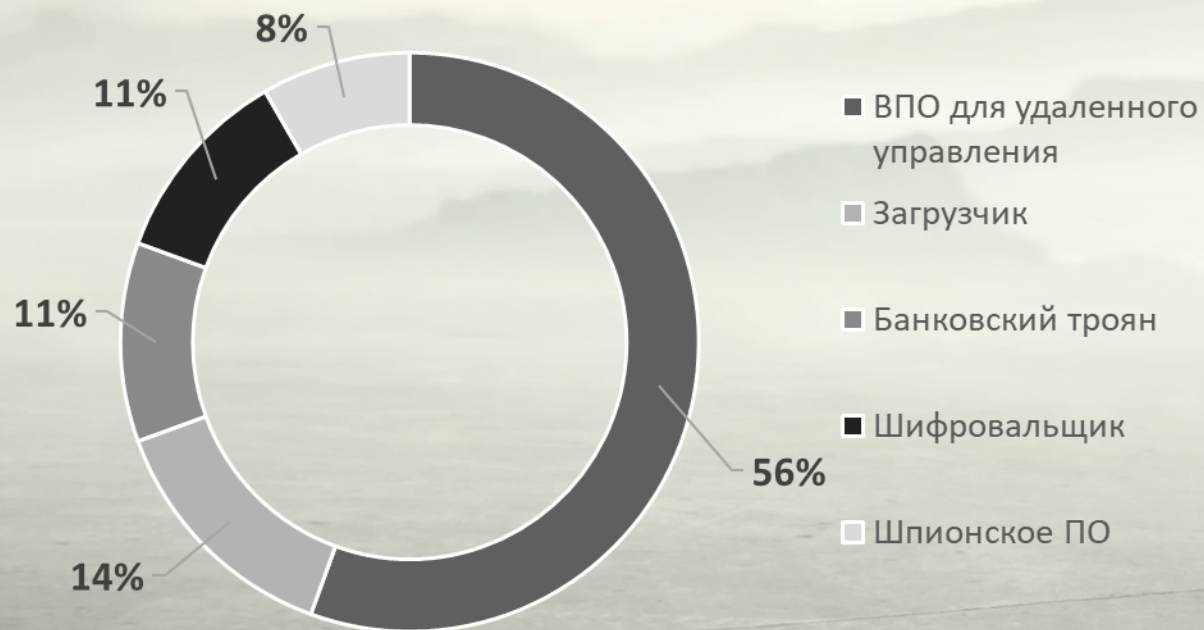


- Вредоносном ПО
 - Песочницах: агентных и безагентных
 - Эволюции популярных методов обхода песочниц
-
- Обнаружении техник обхода изолированных сред

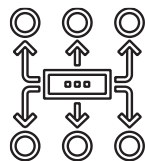
Злоумышленники собирают данные о системе для того, чтобы оценить, **какую пользу они смогут получить из этой атаки**



Вредоносное ПО, в котором чаще всего применяются техники обхода средств виртуализации и анализа:



Какие бывают песочницы



Агентные

Внутри VM присутствует вспомогательный агент, который получает и передает интересные события и артефакты на хостовый сервер

VS



Безагентные

Наблюдают за происходящим за пределами изолированной машины. Соответственно, ВПО, которое находится внутри, не может обнаружить факт наблюдения

Наиболее популярные техники обхода средств виртуализации и анализа

РТ



60% вредоносного ПО, обладающего данными техниками, используется для шпионажа

SELECT * FROM

MSAcpi_ThermalZoneTemperature

SELECT * FROM



MSAcpi_ThermalZoneTemperature

```
C:\WINDOWS\system32>wmic /namespace:\\root\WMI path MSAcpi_ThermalZoneTemperature get CurrentTemperature
CurrentTemperature
3062
2732
2911
2922
2922
2932
2921
```

**Результат выполнения команды
на физическом устройстве**

```
C:\Windows\system32>wmic /namespace:\\root\WMI path MSAcpi_ThermalZoneTemperature get CurrentTemperature
Node - DESKTOP-FU7K5UL
ERROR:
Description = Not supported
```

**Результат выполнения команды
в среде виртуализации**

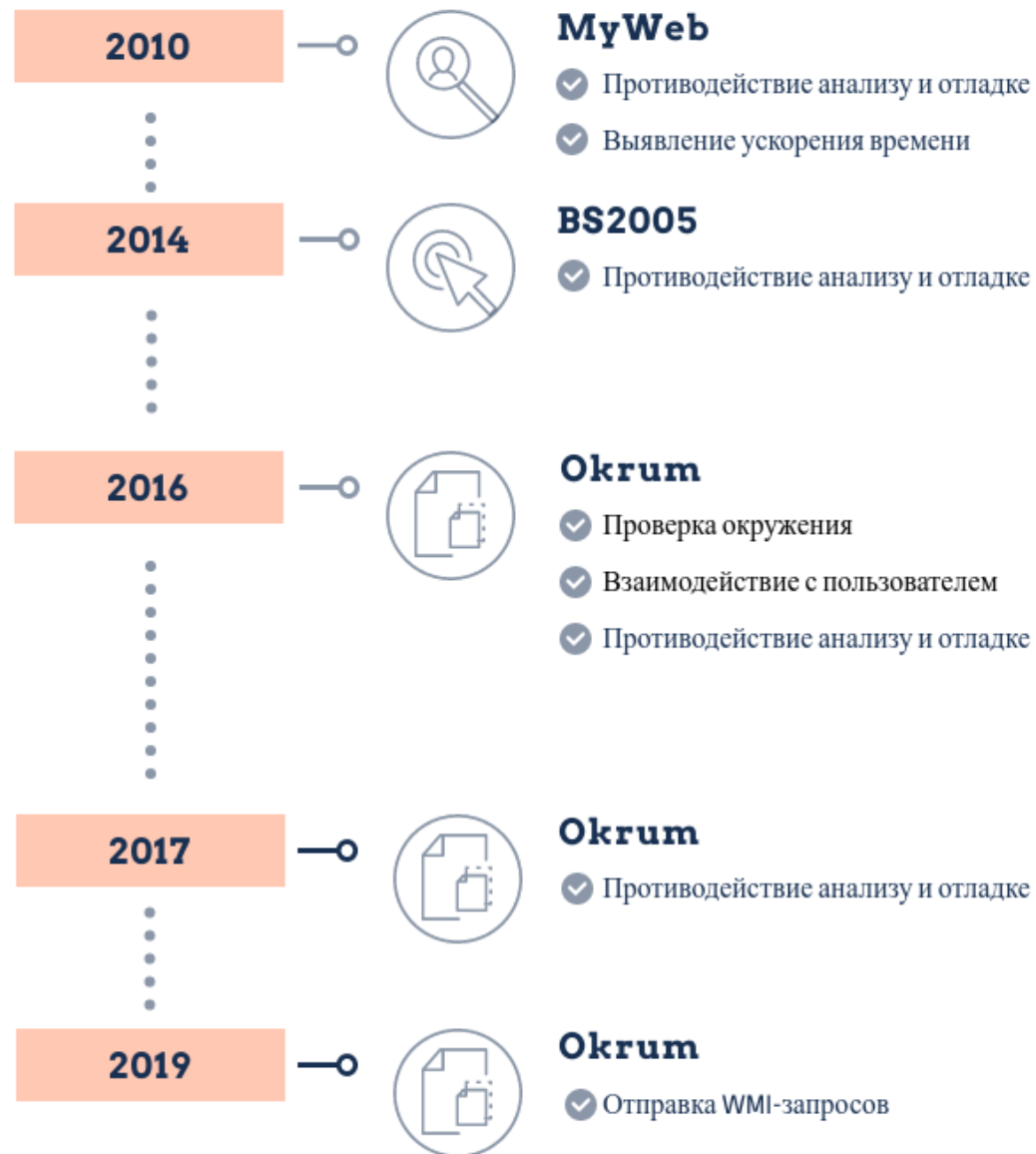
Наиболее популярные техники обхода средств виртуализации и анализа

РТ



60% вредоносного ПО, обладающего данными техниками, используется для шпионажа

Эволюция техник на примере APT-группировки Ke3chang (APT-15)



Эволюция техник на примере APT-группировки Ke3chang (APT-15)

РТ

MyWeb

2010

- Противодействие анализу и отладке • • • • • Двойной вызов функции GetTickCount
- Выявление ускорения времени • • • • • Сравнение значения в миллисекундах последнего вызова со значением, сохраненным при первом вызове

BS2005

2014

- Противодействие анализу и отладке • • • • • Двойной вызов функции GetTickCount

Oknum

2016

- Проверка окружения • • • • • Вызов функции GetGlobalMemoryStatusEx для проверки, что объем фактической физической памяти более 15Гб
- Взаимодействие с пользователем • • • • • Проверка, что левая кнопка мыши была нажата не менее трех раз
- Противодействие анализу и отладке • • • • • Двойной вызов функции GetTickCount

2017

- Противодействие анализу и отладке • • • • • Использование стеганографии для сокрытия загрузчика

2019

- Отправка WMI-запросов • • • • • Проверка объема физической памяти, используя WMI-запросы

РТ

Обнаружение техник обхода изолированных сред



PT Sandbox, подозрительные детекты



Поведенческий анализ

Скачать результаты анализа



Результат поведенческого анализа

Троян-вымогатель

Поведенческий анализ

Обнаруженное опасное ПО

Троян
Trojan.Win32.Generic.a

Троян-вымогатель
Trojan-Ransom.Win32.Generic.a

Потенциально опасное поведение

Create.Process.BCDEdit.BootSettings

Create.Process.DotNet.RestrictionBypass

Create.Process.Vssadmin.ShadowCopySettings

Create.Process.WMIC.WMIManagement

Read.File.Name.Enumeration

Read.Process.Info.AntiDebug

Read.Process.Name.Enumeration

Write.File.Data.Modify

Write.Registry.Key.Persistence

PT Sandbox, подозрительные детекты



<[Action].[Object].[Property].[Description]>

- Выполненное действие
- Объект, над которым выполнили действие
- Свойство объекта, представляющее интерес
- Описание цели выполненного действия

Например, **Write.Registry.Key.Persistence**

Read.Process.Name.Enumeration

Получение списка запущенных процессов в системе

На примере kernel32!CreateToolhelp32Snapshot

```
v15 = 0;
v24 = 0;
v25 = 0;
if ( !th32ProcessID )
    th32ProcessID = GetCurrentProcessId();
ms_exc.registration.TryLevel = 0;
Status = ThpCreateRawSnap(
    dwFlags,
    th32ProcessID,
    &v22,
    (int)&DebugBuffer,
    (int)&v27,
    (int)&v17,
    (int)&v21,
    (int)&v28,
    (int)&v19);
if ( Status < 0 )
{
    BaseSetLastNTErrror(Status);
    goto LABEL_50;
}
if ( v28 )
{
```

```
while ( 1 )
{
    ms_exc.registration.TryLevel = 0;
    RegionSize = ResultLength;
    v9 = NtAllocateVirtualMemory((HANDLE)0xFFFFFFFF, BaseAddress, 0, &RegionSize, 0x1000u, 4u);
    ms_exc.registration.TryLevel = -2;
    if ( v9 < 0 )
        break;
    ResultLength = RegionSize;
    v9 = NtQuerySystemInformation(SystemProcessInformation, *BaseAddress, RegionSize, &ResultLength);
    if ( v9 != -1073741820 )
        break;
    NtFreeVirtualMemory((HANDLE)0xFFFFFFFF, BaseAddress, &RegionSize, 0x8000u);
    *BaseAddress = 0;
    ResultLength = (ResultLength + 0x1FFF) & 0xFFFFE000;
}
v10 = v17;
}
if ( v10 & 0x18 && v9 >= 0 )
{
    v12 = RtlCreateQueryDebugBuffer(0, 0);
```


Read.Process.Name.Enumeration

Системный вызов (syscall) и heaven's gate в случае WOW64

```
public NtQuerySystemInformation
NtQuerySystemInformation proc near ; CODE XREF: RtlCreateHeap+CE3↑p
                                ; EtwpStartUmLogger+1B5↑p ...
                                ; NtQuerySystemInformation
                                ; RtlGetNativeSystemInformation
    mov     r10, rcx
    mov     eax, 36h ; '6'
    test    byte ptr ds:7FFE0308h, 1
    jnz     short loc_18009FE25
    syscall ; Low latency system call
    retn

; -----
loc_18009FE25: ; CODE XREF: NtQuerySystemInforma
               int     2Eh
               ; DOS 2+ internal - EXECUTE COMMAN
               ; DS:SI -> counted CR-terminated
               retn
NtQuerySystemInformation endp
```

```
; __stdcall NtQuerySystemInformation(x, x, x, x)
public _NtQuerySystemInformation@16
_NtQuerySystemInformation@16 proc near ; CODE XREF: TpInitializePackage()+43↑p
                                ; EtwpStartUmLogger(x,x,x,x)+16E↑p ...
                                ; NtQuerySystemInformation
    mov     eax, 36h ; '6'
    mov     edx, offset Wow64SystemServiceCall@0 ; Wow64SystemService
    call    edx ; Wow64SystemServiceCall() ; Wow64SystemServiceCall()
    retn     10h
_NtQuerySystemInformation@16 endp
```

Create.Query.WMI.CheckVM



В большинстве случаев проверка окружения

<https://github.com/LordNoteworthy/al-khaser/blob/master/al-khaser/AntiVM/Generic.cpp>

```
1  #include "pch.h"
2
3  #include "Generic.h"
4
5  /*
6   Check if the DLL is loaded in the context of the process
7   */
8  VOID loaded_dlls()
9  {
10     /* Some vars */
11     HMODULE hDll;
```

Search "select *" (24 hits in 1 file of 1 searched)

new 2 (24 hits)

```
Line 466:      bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_Processor"));
Line 534:      bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_LogicalDisk"));
Line 1007:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_BIOS"));
Line 1078:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_ComputerSystem"));
Line 1147:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_ComputerSystem"));
Line 1221:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM MSAcpi_ThermalZoneTemperature"));
Line 1280:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_Processor"));
Line 1365:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_Fan"));
Line 1416:     bStatus = ExecWMIQuery(&pSvc, &pLoc, &pEnumerator, T("SELECT * FROM Win32_VideoController"));
Line 1548:     int count = wmi_query_count(T("SELECT * FROM Win32_CacheMemory"));
Line 1561:     int count = wmi_query_count(T("SELECT * FROM Win32_PhysicalMemory"));
Line 1574:     int count = wmi_query_count(T("SELECT * FROM Win32_MemoryDevice"));
Line 1587:     int count = wmi_query_count(T("SELECT * FROM Win32_MemoryArray"));
Line 1600:     int count = wmi_query_count(T("SELECT * FROM Win32_VoltageProbe"));
Line 1613:     int count = wmi_query_count(T("SELECT * FROM Win32_PortConnector"));
Line 1626:     int count = wmi_query_count(T("SELECT * FROM Win32_SMBIOSMemory"));
Line 1639:     int count = wmi_query_count(T("SELECT * FROM Win32_PerfFormattedData_Counters_ThermalZoneInformation"));
Line 1652:     int count = wmi_query_count(T("SELECT * FROM CIM_Memory"));
Line 1665:     int count = wmi_query_count(T("SELECT * FROM CIM_NumericSensor"));
Line 1678:     int count = wmi_query_count(T("SELECT * FROM CIM_PhysicalConnector"));
Line 1691:     int count = wmi_query_count(T("SELECT * FROM CIM_Sensor"));
Line 1704:     int count = wmi_query_count(T("SELECT * FROM CIM_Slot"));
Line 1717:     int count = wmi_query_count(T("SELECT * FROM CIM_TemperatureSensor"));
Line 1730:     int count = wmi_query_count(T("SELECT * FROM CIM_VoltageSensor"));
```

Create.Query.WMI.CheckVM



Оффтоп: проверка защитных средств (Create.Query.WMI.CheckAntivirus)

<https://securelist.com/the-cozyduke-apt/69731/>

Anti-detection and trojan functionality

The file collects system information, and then invokes a WMI instance in the rootsecuritycenter namespace to identify security products installed on the system, meaning that this code was built for x86 systems, wql here:

```
SELECT * FROM AntiVirusProduct  
SELECT * FROM FireWallProduct
```

The code hunts for several security products to evade:

- CRYSTAL
- KASPERSKY
- SOPHOS
- DrWeb
- AVIRA
- COMODO Dragon

Create.Query.WMI.CheckVM



Для WMI вызовов используется COM интерфейс (CLSID_WbemLocator):

- **CoInitialize**
- **CoCreateInstance**
- **IWbemLocator::ConnectServer**
- **IWbemServices::ExecQuery**

<https://docs.microsoft.com/en-us/windows/win32/wmisdk/example--getting-wmi-data-from-the-local-computer>

Create.Query.WMI.CheckVM



Фрагмент кода из DRAKVUF

<https://github.com/tklengyel/drakvuf/blob/master/src/plugins/wmimon/wmimon.cpp>

```
try
{
    vtable vt(drakvuf, info, data->m_vtable, 26);

    search_breakpoint_by_addr bp(plugin, vt[20]);
    plugin->register_trap(info, ExecQuery_handler, bp, "ExecQuery");
    plugin->register_trap(info, GetObject_handler, bp.set_addr(vt[6]), "GetObject");
    plugin->register_trap(info, ExecMethod_handler, bp.set_addr(vt[24]), "ExecMethod");
}
catch (const std::exception& e)
{
    PRINT_DEBUG("[WMIMon] Failed to read a vtable of IWbemServices\n");
}

return VMI_EVENT_RESPONSE_NONE;
}
```

Read.Registry.Key.*

Получение информации о системе из реестра:

- **Read.Registry.Key.CheckBios**
 - **HKLM\HARDWARE\DESCRIPTION\System (SystemBiosDate)**
- **Read.Registry.Key.CheckCPU**
 - **HKLM\HARDWARE\DESCRIPTION\System\CentralProcessor\0 (ProcessorNameString)**
- **Read.Registry.Key.CheckInstalledSoft**
 - **HKLM\SOFTWARE(\WOW6432Node)\Microsoft\Windows\CurrentVersion\Uninstall**
- **Read.Registry.Key.DigitalProductId**
 - **HKLM\SOFTWARE(\WOW6432Node)\Microsoft\Windows NT\CurrentVersion (DigitalProductId)**

Ложные срабатывания

Следующие вызовы часто используются легитимным софтом (1/2):

- **kernel32!GlobalMemoryStatus**
- **advapi32!GetUserNameA**
- **ws2_32!gethostname**
- **ntdll!RtlGetVersion**

Ложные срабатывания

Следующие вызовы часто используются легитимным софтом (2/2):

- **kernel32!GetComputerNameA**
- **kernel32!GetProductInfo**
- **iphlpapi!GetAdaptersInfo**
- ...

Рекогносцировка ВПО

Загрузчики Lo2

Несмотря на классификацию, задачи у троянов различаются. Так, файл **Serviceflow.exe** выполняет сторожевую роль (watchdog). Он собирает информацию о системе:

- имя пользователя,
- имя компьютера,
- содержимое каталогов \Program F
- версию ОС,
- данные о процессоре —

3. Проверка наличия в списке процессов системы ант
сравнения CRC32 от имени процесса в нижнем реги
внутри кода.

4. Сбор информации о системе:

- дата установки системы,
- IP-адрес зараженной машины,
- тип системы (серверная или стационарная),
- версия Windows (от XP до 10).

Рисунок 24. Общая информация о полезной нагрузке

Создает отдельный поток, в котором происходят все полезные действия.

В начале работы выполняет разведку в системе и собирает пользовательскую информацию:

- имя компьютера;
- IP-адрес;
- кодовую страницу OEM;
- MAC-адрес (позднее от полученного значения вычисляется MD5-хеш-сумма, которая будет использоваться при взаимодействии с управляющим сервером);

ptsecurity.com/ru-ru/research/pt-esc-threat-intelligence/issleduem-aktivnost-kibergruppirovki-donot-team/

ptsecurity.com/ru-ru/research/pt-esc-threat-intelligence/cobalt-obnovlenie-tactic-i-instrumentov/

ptsecurity.com/ru-ru/research/pt-esc-threat-intelligence/shadowpad-novaya-aktivnost-gruppirovki-winnti/

Рекогносцировка ВПО

```
nSize = 30;
GetComputerNameA(g_Config.acsComputerName, &nSize);
pcbBuffer = 30;
GetUserNameA(g_Config.acsUserName, &pcbBuffer);
NewCore::MachineInfo::GetStringOS((int)&g_Config.acsOsVersion);
v2 = 0;
SystemInfo.dwOemId = 0;
SystemInfo.dwPageSize = 0;
SystemInfo.lpMinimumApplicationAddress = 0;
SystemInfo.lpMaximumApplicationAddress = 0;
SystemInfo.dwActiveProcessorMask = 0;
SystemInfo.dwNumberOfProcessors = 0;
SystemInfo.dwProcessorType = 0;
SystemInfo.dwAllocationGranularity = 0;
*(DWORD*)&SystemInfo.wProcessorLevel = 0;
GetSystemInfo(&SystemInfo);
g_Config.dwMHz = NewCore::MachineInfo::GetMhz();
g_Config.dwNumberOfProcessor = SystemInfo.dwNumberOfProcessors;
v3 = NewCore::MachineInfo::GetTotalPhysMemoryWCS();
g_Config.dwTotalPhysMemory = _wtoi(v3);
sub_10009A30(&g_Config.f_IsLocalAdminG, (int)&g_Config.field_7F4);
name = 0;
memset(v55, 0, sizeof(v55));
if (gethostname(&name, 512) || (v4 = gethostbyname(&name), (v5 = v4) == 0))
{
    qmemcpy(g_Config.acsIpAddressesList_Cpy, "N/A", 3);
}
else
```

BlueCore RAT, Goblin Panda/Cycldek APT (MD5: 600e14e4b0035c6f0c6a344d87b6c27f)

Рекогносцировка ВПО



Решение:

обнаруживать N легитимных действий за M секунд

Read.System.Info.Reconnaissance



Спасибо за внимание!

ptsecurity.com