Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский Политехнический Университет Петра Великого
—
Институт прикладной математики и механики
Кафедра «Информационная безопасность компьютерных систем»

ЛАБОРАТОРНАЯ РАБОТА № 4

«Изучение подходов к автоматизированному анализу вредоносного программного обеспечения, обладающего механизмами самозащиты»

по дисциплине «Безопасность операционных систем»

Выполнил
студент гр. 43609/1 Куликов Д.А.
<подпись>

Преподаватель
Жуковский Е.В.
<подпись>

Санкт-Петербург
2019

1 Цель работы

Изучение подходов к автоматизированному анализу вредоносного программного обеспечения, обладающего механизмами самозащиты. Изучение технологий динамического и статического анализа исполняемых файлов. Получение навыков анализа вредоносного программного обеспечения.

2 Формулировка задания

В ходе выполнения лабораторной работы необходимо выполнить следующие действия:

1. Изучить представленные в базе знаний MITRE ATT&CK [1] техники, используемые в целевых атаках в соответствии с указанным в таблице 1 вариантом задания. Представить описание изученных техник в виде таблицы 2. Составить перечень признаков (функций, имен объектов), применимых для возможности выявления данных техник.

2. Изучить существующие механизмы самозащиты, используемые во вредоносном программном обеспечение (антиотладочные методы, методы выявления виртуальных машин, выявление средств защиты и анализа и т.п.). [2 - 9]. Составить перечень (таблица 3) известных механизмов с указанием используемых характерных инструкций, функций или аргументов функций, которые могут быть использованы для их выявления во время динамического анализа.

3. Разработать программу, осуществляющую динамическое исследование анализируемого исполняемого файла с использованием средства бинарной инструментации (DBI) в соответствии с вариантом задания (Intel Pin / DynamoRIO). Программа должна принимать на вход список искомых инструкций, функций, функций:значений аргументов. При вызове искомой функции с аргументами искать сигнатурные аргументы (строки) можно путем анализа (включая разыменование адресов) стека и регистров. Выявленные сигнатурные вызовы функций следует сохранять в текстовый файл с указанием адреса вызова и другой требуемой информации.

4. Разработанная программа должна на основе анализа исполняемых инструкций, вызываемых функций и значений их аргументов осуществлять выявление и обход распространенных механизмом самозащиты (описанных в п. 2 задания) и потенциальных вредоносных действий (описанных в п. 1 задания). Также отслеживать вызовы передачи управления в сторонние модули и процессы (создание потоков, процессов, запуск сторонних исполняемых файлов и т.п.)

5. Повести динамический анализ с использованием разработанной программы исполняемых файлов APT, в соответствии с вариантом задания к лабораторной работе №3, а также файлов APT, указанных в таблице 1. Изучить участки кода, содержащие выявленные в ходе анализа вызовы функций, предположительно относящиеся к механизмам самозащиты или вредоносным техникам. Результаты анализа привести в виде таблицы 4.

6. Привести фрагменты ассемблерного / псевдокода (восстановленного кода на языке C) выявленных механизмов самозащиты и используемых техник атаки, выявленных в соответствие с п. 7 задания.

3 Ход работы

В соответствии с полученным вариантом задания требуется изучить следующие представленные в базе знаний MITRE ATT&CK техники, используемые в целевых атаках:

Execution through API

Process Hollowing

Input Capture

Domain Trust Discovery

Service Stop

Disabling Security Tools

Execution Guardrails

File Permissions Modification

LLMNR/NBT-NS Poisoning and Relay

Credential Dumping

Описание техник представлено в приложении в таблице 1. Изученные существующие механизмы самозащиты, используемые во вредоносном программном обеспечение, представлены в приложении в таблице 2.

Была разработана программа, осуществляющая динамическое исследование исполняемых файлов APT DPRK и GreyEnergyAPT с использованием средства бинарной инструментария Intel PIN.

APT DPRK был изучен в лабораторной работе №3.

При реализации инструмента анализа, сначала был установлен формат входных данных (в файле FileListFunctions.txt) следующего вида:

<Имя функции>;<Количество аргументов анализа (первые n штук)>;<Тип аргумента 1>;<Тип аргумента 2>;…;

Текущая реализация различает только string и wstring, остальные выводятся в формате PTR.

Входные данные сохраняются в структуры

```
struct Function
{
	string		functionName;
	uint32_t	numArgs;
	vector<string>	argsTypes;
};
```

Далее был установлен call back, на загрузку образа в память процесса:

IMG_AddInstrumentFunction(ImageLoad, 0);

Текущая реализация поддерживает обработку не более трех аргументов, но это легко расширяемо. Следующий участок кода ищет в образе функции из списка и в зависимости от количества аргументов этой функции устанавливает вызов кастомной функции перед выполнением истинной.

```
RTN funcRtn;
for (auto it = functions.begin(); it != functions.end(); ++it)
{
	funcRtn = RTN_FindByName(Image, it->functionName.c_str());
	if (RTN_Valid(funcRtn)) {
		RTN_Open(funcRtn);
		switch (it->numArgs)
		{
		case 0:
			RTN_InsertCall(funcRtn, IPOINT_BEFORE, (AFUNPTR)zeroArgsFunc,
				IARG_ADDRINT, it->functionName.c_str(),
				IARG_INST_PTR,
				IARG_END);
			break;
```

```
                        case 1:
                                RTN_InsertCall(funcRtn, IPOINT_BEFORE, (AFUNPTR)oneArgsFunc,
                                        IARG_ADDRINT, it->functionName.c_str(),
                                        IARG_INST_PTR,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 0,
                                        IARG_END);
                                break;
                        case 2:
                                RTN_InsertCall(funcRtn, IPOINT_BEFORE, (AFUNPTR)twoArgsFunc,
                                        IARG_ADDRINT, it->functionName.c_str(),
                                        IARG_INST_PTR,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 0,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 1,
                                        IARG_END);
                                break;
                        case 3:
                                RTN_InsertCall(funcRtn, IPOINT_BEFORE, (AFUNPTR)threeArgsFunc,
                                        IARG_ADDRINT, it->functionName.c_str(),
                                        IARG_INST_PTR,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 0,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 1,
                                        IARG_FUNCARG_ENTRYPOINT_VALUE, 2,
                                        IARG_END);
                                break;
                        }
                        RTN_Close(funcRtn);
                }
                }
```

Функция обработки вызова функции с одним аргументов (анализируемым) выглядит следующим образом (остальные аналогично):

```
VOID oneArgsFunc(char* funcName, ADDRINT address, ADDRINT arg1)
{
        fileFunctionsLog << "Function: " << funcName << " Address: " << address << " Args:
";
        for (auto it = functions.begin(); it != functions.end(); ++it)
        {
                if (it->functionName == funcName)
                {
                        if (it->argsTypes[0] == string("string"))
                        {
                                fileFunctionsLog << (char*)arg1;
                        }else if (it->argsTypes[0] == string("wstring"))
                        {
                                wstring tempWString = wstring((wchar_t*)arg1);
                                string  tempString = string(tempWString.begin(),
tempWString.end());
                                fileFunctionsLog << tempString;
                        }
                        else
                        {
                                fileFunctionsLog << arg1;
                        }
                }
        }
        fileFunctionsLog <<'\n';
}
```

Аналогичным образом заменяются функции защиты от дебага и анализа:

```
	funcRtn = RTN_FindByName(Image, "GetTickCount");
	if (RTN_Valid(funcRtn)) {
		fileFunctionsLog << "GetTickCount is found.\n";
		RTN_Open(funcRtn);
		RTN_Replace(funcRtn, (AFUNPTR)GetTickCount);
		RTN_Close(funcRtn);
	}
	funcRtn = RTN_FindByName(Image, "IsDebuggerPresent");
	if (RTN_Valid(funcRtn)) {
		fileFunctionsLog << "IsDebuggerPresent is found.\n";
		RTN_Open(funcRtn);
		RTN_Replace(funcRtn, (AFUNPTR)IsDebuggerPresent);
		RTN_Close(funcRtn);
		}
UINT __stdcall GetTickCount() {
	fileFunctionsLog << "Returned zero.\n";
	return (UINT)0;
}

bool __stdcall IsDebuggerPresent() {
	fileFunctionsLog << "Change IsDebuggerPresent value to false: compelete.\n";
	return false;
}
```

Аналогичным образом можно реализовать остальные.

Таким образом, был реализован анализ функций, используемых для обнаружения факта отладки или работы в виртуальной среде. Все их вызовы логируются.

Далее была составлена таблица 1 и изучены используемые функции. Из них составлены входные данные анализатора, плюс некоторые не входящие в таблицу:

CreateProcessA;2;string;string;
CreateProcessW;2;wstring;wstring;
CreateProcessAsUserA;3;int;string;string;
CreateProcessAsUserW;3;int;wstring;wstring;
CreateProcessWithLogonA;3;string;string;string;
CreateProcessWithLogonW;3;wstring;wstring;wstrin
g;
CreateProcessWithTokenW;3;int;int;wstring;
LoadLibraryA;1;string;
LoadLibraryW;1;wstring;
LoadLibraryExA;1;string;
LoadLibraryExW;1;wstring;
LoadModule;2;string;string;
LoadPackagedLibrary;1;string;
WinExec;1;string;
ShellExecutedA;3;int;string;string;
ShellExecutedW;3;int;wstring;wstring;
ShellExecutedExA;3;int;string;string;
ShellExecutedExW;3;int;wstring;wstring;
WriteProcessMemory;1;int;
ZwUnmapViewOfSection;1;int;
NtUnmapViewOfSection;1;int;
SetWindowsHookExA;1;int;
SetWindowsHookExW;1;int;
GetKeyState;1;int;
GetAsyncKeyState;1;int;
RegOpenKeyExA;2;int;string;
RegOpenKeyExW;2;int;wstring;
RegOpenKeyA;2;int;string;
RegCreateKeyExA;2;int;string;
RegCreateKeyExW;2;int;wstring;
RegCreateKeyA;2;int;string;
RegSetValueA;2;int;string;
RegSetValueW;2;int;wstring;

RegSetValueExA;2;int;string;
RegSetValueExW;2;int;wstring;
RegDeleteKeyA;2;int;string;
RegDeleteKeyW;2;int;wstring;
RegDeleteKeyExA;2;int;string;
RegDeleteKeyExW;2;int;wstring;
socket;3;int;int;int;
listen;2;int;int;
accept;1;int;
connect;1;int;
CreateServiceA;3;int;string;string;
OpenSCManagerA;3;string;string;int;
CreateServiceW;3;int;wstring;wstring;
OpenSCManagerW;3;wstring;wstring;int;
OpenService;3;int;string;int;
ControlService;3;int;int;int;
QueryServiceStatus;2;int;int;
GetProcAddress;2;int;string;
CreateFileA;3;string;int;int;
CreateFileW;3;wstring;int;int;
WriteFile;1;int;
ReadFile;1;int;
DeleteFileA;1;string;
DeleteFileW;1;wstring;
DsEnumerateDomainTrusts;2;string;int;
ChangeServiceConfigA;3;int;int;int;
ChangeServiceConfigW;3;int;int;int;
GetEnvironmentVariable;1;string;

Во время отслеживания деятельности группы GreyEnergy исследователи чаще всего видели, как злоумышленники использовали два первичных вектора инфекции. первый актуален для организаций с автономными веб-сервисами.

Если такой общедоступный веб-сервис работает на сервере, который подключен к внутренней сети, злоумышленники постараются скомпрометировать его, а затем пробраться внутрь сети.

Вторым вектором инфекции является использование писем с вредоносными вложениями.

Исследователи заметили, что вредоносные документы сбрасывают «GreyEnergy mini», облегченный бэкдор первого этапа, который не требует административных привилегий. После компрометации компьютера с помощью GreyEnergy mini злоумышленники наносят на карту сеть и собирают пароли для получения в домене привилегий администратора. С этими привилегиями злоумышленники могут контролировать всю сеть.

Группа GreyEnergy использует для этих задач довольно стандартные инструменты: Nmap и Mimikatz.

Как только злоумышленники завершат начальное сопоставление сети, они смогут развернуть свой флагманский бэкдор - основная вредоносная программа GreyEnergy. Эта вредоносная программа требует прав

администратора, которые уже должны были быть получены до того, как эта стадия достигнута. Согласно нашему исследованию, создатели GreyEnergy разворачивают этот бэкдор в основном на двух типах конечных точек: серверы с высокой продолжительностью работы и рабочие станции, используемые для управления среды ICS.

Чтобы сделать связь с командными и управляющими (C & C) серверами более надежной, злоумышленники могут развернуть дополнительное программное обеспечение на внутренних серверах в скомпрометированной сети, чтобы каждый сервер работал в качестве прокси. Такой прокси-сервер C & C перенаправляет запросы с зараженных узлов в сети на внешний C & C сервер в интернете. Таким образом, это может быть менее подозрительно для защиты, которая замечает, что несколько компьютеров «общаются» с внутренним сервером, а не с удаленным сервером. Эта техника может также использоваться злоумышленниками для контроля вредоносных программ в различных сегментах скомпрометированной сети. Похожая техника, использующая внутренние серверы в качестве прокси-серверов C & C, была использована Duqu 2.0 APT.

Если затронутая организация имеет общедоступные веб-серверы, подключенные к внутренней сети, злоумышленники могут развернуть «резервные» бэкдоры на этих серверах. Эти черные ходы используются для восстановления доступа в сеть в случае обнаружения и удаления основных бэкдоров.

Все C & C-серверы, которые использовались вредоносным ПО GreyEnergy, являются ретрансляторами Tor.

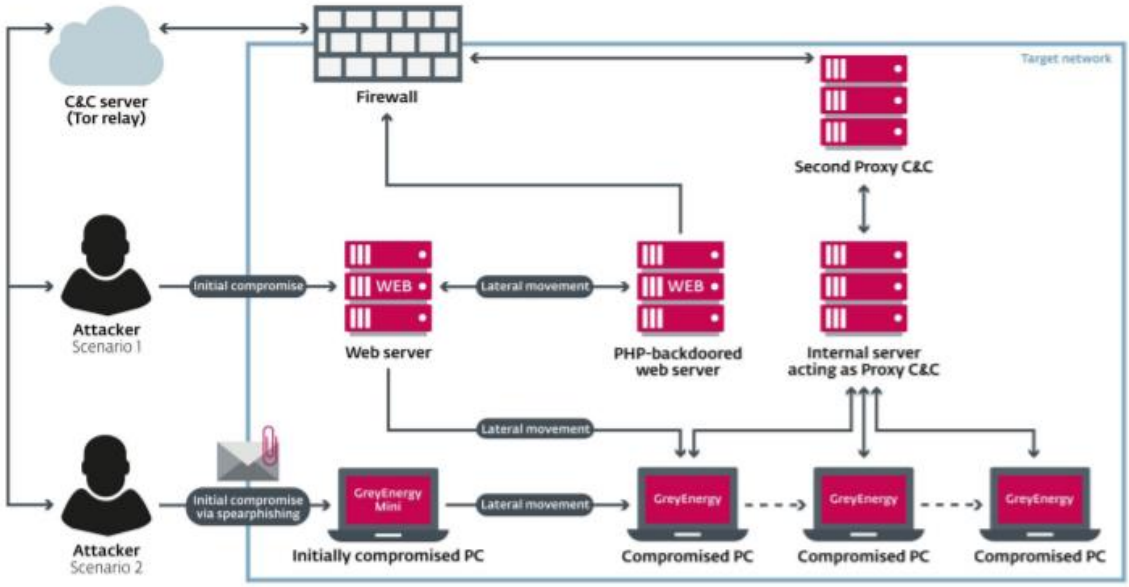

Figure 1. // Simplified scheme of the two network compromise scenarios used by the GreyEnergy group

GreyEnergy Mini

GreyEnergy mini - это легковесный бэкдор первого уровня, который используется злоумышленниками для оценки взломать компьютер и получить первоначальную точку опоры в сети. Обычно вредоносная программа GreyEnergy mini загружается вредоносным документом, который был доставлен с использованием электронной почты. GreyEnergy mini – это также известный как FELIXROOT.

В сентябре 2017 года ESET обнаружил поддельный документ Microsoft Word на украинском языке, вредоносный макрос. Документ-приманка был разработан, чтобы выглядеть как интерактивная форма, подсказывающая жертве, что необходимо включить макросы, чтобы заполнить его.

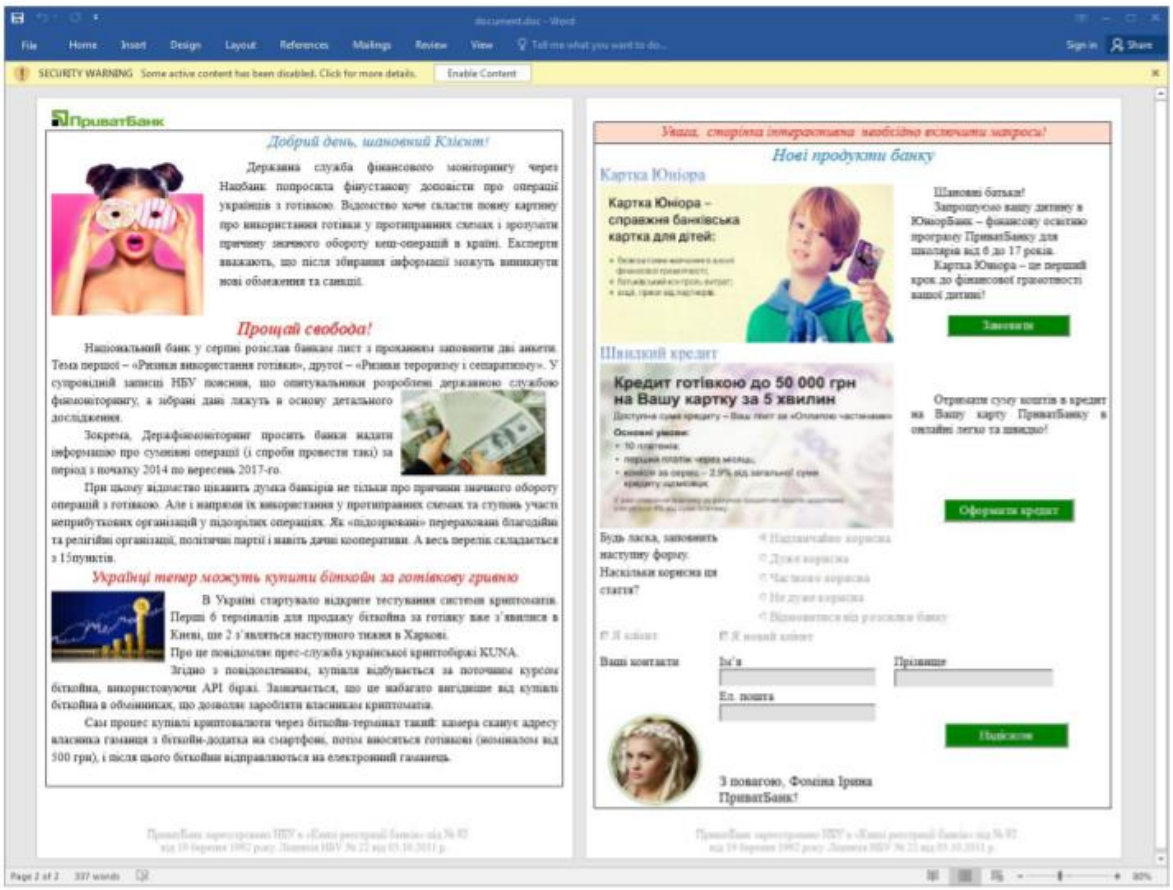

Рисунок 2 – GreyEnergy mini

Как только макрос включен, его код пытается загрузить и выполнить двоичный файл с удаленного сервера.

Чтобы оценить ценность скомпрометированного компьютера, вредоносная программа собирает столько информации, сколько возможно и отправляет собранные данные в C & C. Сбор информации осуществляется с использованием WMI

Язык запросов (WQL) и запросы к реестру Windows. Следующая информация собирается:

• Имя компьютера

• Версия операционной системы, включая версию пакета обновления

• Язык по умолчанию

• Имя пользователя

• Текущие привилегии пользователя Windows, повышение прав, уровень контроля учетных записей

• Настройки прокси

• Информация о компьютере (производитель, модель, тип системы)

• Часовой пояс

• Установленное программное обеспечение безопасности (антивирус и брандмауэр)

• Список пользователей и доменов

• Список установленного программного обеспечения, полученный из реестра

• Информация о сети (IP-адреса, DHCP-сервер и т. Д.)

• Список запущенных процессов

Вредоносная программа получает команды от C & C - сервера. Поддерживаются следующие команды:

Command ID	Meaning
1	Collect information about computer
2	Download and run executable file from temporary directory
3	Run shell command
4	Uninstall itself from compromised computer
5	Download and run .BAT file from temporary directory
6	Download file to local drive
7	Upload file

Рисунок 3 – Поддерживаемые команды

4 Результаты

Были исследованы APT DPRK и GreyEnergyAPT. GreyEnergyAPT имеет doc формат, поэтому к нему этот метод неприменим. DPRK не реализует никаких антиотладочных приемов:

```
Function: OpenSCManagerA Address: 1986487536
Function: LoadLibraryExW Address: 1964948119 Args: rpcrt4.dll
Function: RegOpenKeyExA Address: 1996567566 Args: 2147483650 Software\Microsoft\Rpc
Function: RegOpenKeyExW Address: 1996538249 Args: 2147483650 Software\Policies\Microsoft\Windows NT\Rpc
Function: RegOpenKeyExW Address: 1996538249 Args: 2147483650 System\CurrentControlSet\Control\SQMServiceList
Function: DeleteFileA Address: 1996507083 Args: C:\Windows\system32\scardprv.dll
Function: DeleteFileA Address: 1964951713 Args: C:\Windows\system32\scardprv.dll
Function: DeleteFileW Address: 1964968373 Args: C:\Windows\system32\scardprv.dll
Function: CreateFileA Address: 1996541672 Args: C:\Windows\system32\scardprv.dll 1073741824 0
Function: CreateFileW Address: 1996541014 Args: C:\Windows\system32\scardprv.dll 1073741824 0
Function: CreateFileW Address: 1964943440 Args: C:\Windows\system32\scardprv.dll 1073741824 0
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
```

Рисунок 4 – DPRK - создание и запись dll, создание сервиса

```
Function: DeleteFileA Address: 1996507083 Args: C:\Windows\system32\Wmmvsvc.dll
Function: DeleteFileA Address: 1964951713 Args: C:\Windows\system32\Wmmvsvc.dll
Function: DeleteFileW Address: 1964968373 Args: C:\Windows\system32\Wmmvsvc.dll
Function: CreateFileA Address: 1996541672 Args: C:\Windows\system32\Wmmvsvc.dll 1073741824 0
Function: CreateFileW Address: 1996541014 Args: C:\Windows\system32\Wmmvsvc.dll 1073741824 0
Function: CreateFileW Address: 1964943440 Args: C:\Windows\system32\Wmmvsvc.dll 1073741824 0
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
```

Рисунок 5 – DPRK – создание и запись dll второго сервиса

```
Function: CreateFileA Address: 1996541672 Args: C:\Windows\system32\mssscardprv.ax 2147483648 3
Function: CreateFileW Address: 1996541014 Args: C:\Windows\system32\mssscardprv.ax 2147483648 3
Function: CreateFileW Address: 1964943440 Args: C:\Windows\system32\mssscardprv.ax 2147483648 3
Function: DeleteFileA Address: 1996507083 Args: C:\Windows\system32\mssscardprv.ax
Function: DeleteFileA Address: 1964951713 Args: C:\Windows\system32\mssscardprv.ax
Function: DeleteFileW Address: 1964968373 Args: C:\Windows\system32\mssscardprv.ax
Function: CreateFileA Address: 1996541672 Args: C:\Windows\system32\mssscardprv.ax 1073741824 0
Function: CreateFileW Address: 1996541014 Args: C:\Windows\system32\mssscardprv.ax 1073741824 0
Function: CreateFileW Address: 1964943440 Args: C:\Windows\system32\mssscardprv.ax 1073741824 0
Function: WriteFile Address: 1996559360 Args: 236
```

Рисунок 6 – DPRK- создание файла для записи данных шпионажа

```
Function: CreateFileW Address: 1964943440 Args: C:\Windows\system32\mssscardprv.ax 1073741824 0
Function: WriteFile Address: 1996559360 Args: 236
Function: WriteFile Address: 1964930341 Args: 236
Function: RegOpenKeyExA Address: 1968785671 Args: 2147483650 SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost
Function: RegOpenKeyExA Address: 1996567566 Args: 2147483650 SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost
Function: RegSetValueExA Address: 1968772275 Args: 236 SCardPrv
Function: RegSetValueExA Address: 1996502771 Args: 236 SCardPrv
Function: OpenSCManagerA Address: 1968778200
Function: OpenSCManagerA Address: 1986487536
Function: CreateServiceA Address: 1968976216 Args: 2567592 SCardPrv SmartCard Protector
Function: CreateServiceA Address: 1986483836 Args: 2567592 SCardPrv SmartCard Protector
Function: RegOpenKeyExA Address: 1968785671 Args: 2147483650 SYSTEM\CurrentControlSet\Services\SCardPrv
Function: RegOpenKeyExA Address: 1996567566 Args: 2147483650 SYSTEM\CurrentControlSet\Services\SCardPrv
Function: RegCreateKeyA Address: 1968753921 Args: 236 Parameters
Function: RegCreateKeyExA Address: 1996500348 Args: 236 Parameters
Function: RegSetValueExA Address: 1968772275 Args: 240 ServiceDll
Function: RegSetValueExA Address: 1996502771 Args: 240 ServiceDll
Function: OpenSCManagerA Address: 1968778200
Function: OpenSCManagerA Address: 1986487536
Function: RegOpenKeyExA Address: 1968785671 Args: 2147483650 SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost
Function: RegOpenKeyExA Address: 1996567566 Args: 2147483650 SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost
Function: RegSetValueExA Address: 1968772275 Args: 236 Wmmvsvc
Function: RegSetValueExA Address: 1996502771 Args: 236 Wmmvsvc
Function: OpenSCManagerA Address: 1968778200
Function: OpenSCManagerA Address: 1986487536
Function: CreateServiceA Address: 1968976216 Args: 2567592 Wmmvsvc Windows Media Management Driver Extensions
Function: CreateServiceA Address: 1986483836 Args: 2567592 Wmmvsvc Windows Media Management Driver Extensions
Function: RegOpenKeyExA Address: 1968785671 Args: 2147483650 SYSTEM\CurrentControlSet\Services\Wmmvsvc
Function: RegOpenKeyExA Address: 1996567566 Args: 2147483650 SYSTEM\CurrentControlSet\Services\Wmmvsvc
Function: RegCreateKeyA Address: 1968753921 Args: 236 Parameters
Function: RegCreateKeyExA Address: 1996500348 Args: 236 Parameters
Function: RegSetValueExA Address: 1968772275 Args: 240 ServiceDll
Function: RegSetValueExA Address: 1996502771 Args: 240 ServiceDll
Function: OpenSCManagerA Address: 1968778200
Function: OpenSCManagerA Address: 1986487536
Function: LoadLibraryExW Address: 1964948119 Args: kernel32.dll
Function: GetProcAddress Address: 1964928041 Args: 1996226560 SortGetHandle
Function: GetProcAddress Address: 1964928041 Args: 1996226560 SortCloseHandle
```

Рисунок 7 – Создание и выставление параметров сервисов

Средство анализа было дополнительно запущено на DarkTequila.exe

```
Function: LoadLibraryA Address: 2008627548 Args: Shell32.dll
Function: LoadLibraryExA Address: 1977781606 Args: Shell32.dll
Function: LoadLibraryExW Address: 1977793175 Args: Shell32.dll
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
```

Рисунок 2 – Логи DarkTequila.exe NtUnmapViewOfSection может быть признаком использования техники обхода защиты Process Hollowing

```
Function: GetProcAddress Address: 2008626131 Args: 1972043776 CryptAcquireContextA
Function: GetProcAddress Address: 1977773097 Args: 1972043776 CryptAcquireContextA
Function: RegOpenKeyExA Address: 2008626190 Args: 2147483650 SOFTWARE\Microsoft\Cryptography\Defaults\Provider Types\Type 001
Function: RegOpenKeyExA Address: 2008626190 Args: 2147483650 SOFTWARE\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong C
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: CreateFileW Address: 2008599638 Args: C:\Windows\system32\rsaenh.dll 2147483648 1
Function: CreateFileW Address: 1977788496 Args: C:\Windows\system32\rsaenh.dll 2147483648 1
Function: ZwUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: NtUnmapViewOfSection Address: 2009491896 Args: 4294967295
Function: LoadLibraryExA Address: 1977781606 Args: C:\Windows\system32\rsaenh.dll
Function: LoadLibraryExW Address: 1977793175 Args: C:\Windows\system32\rsaenh.dll
```

Рисунок 3 – Создание файла и загрузка созданной dll, работа с ключами реестра

```
Function: socket Address: 1982807736 Args: 23 1 6
Function: LoadLibraryExW Address: 1977793175 Args: C:\Windows\system32\mswsock.dll
Function: GetProcAddress Address: 1977773097 Args: 1971781632 WSPStartup
Function: RegOpenKeyExW Address: 2008596873 Args: 2147483650 SYSTEM\CurrentControlSet\Services\Winsock\Parameters
Function: RegOpenKeyExW Address: 2008596873 Args: 2147483650 System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock
Function: RegOpenKeyExW Address: 2008596873 Args: 2147483650 SYSTEM\CurrentControlSet\Services\Winsock\Setup Migration\Providers
Function: RegOpenKeyExW Address: 2008596873 Args: 228 Tcpip6
Function: RegOpenKeyExW Address: 2008596873 Args: 2147483650 System\CurrentControlSet\Services\Tcpip6\Parameters\Winsock
Function: LoadLibraryExW Address: 1977793175 Args: C:\Windows\System32\wship6.dll
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHOpenSocket
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHOpenSocket2
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHJoinLeaf
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHNotify
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHGetSocketInformation
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHSetSocketInformation
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHGetSockaddrType
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHGetWildcardSockaddr
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHGetBroadcastSockaddr
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHAddressToString
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHStringToAddress
Function: GetProcAddress Address: 1977773097 Args: 1969618944 WSHIoctl
Function: GetProcAddress Address: 2008626131 Args: 1982791680 setsockopt
Function: GetProcAddress Address: 1977773097 Args: 1982791680 setsockopt
Function: GetProcAddress Address: 2008626131 Args: 1982791680 bind
Function: GetProcAddress Address: 1977773097 Args: 1982791680 bind
```

Рисунок 4 – Фрагментов логов, после применения к APT EnergyBear, видим открытие порта, работу с ключами реестра, загрузку функций работы с сетевым обменом.

5 Вывод

В результате выполнения лабораторной работы были изучены подходы к автоматизированному анализу вредоносного программного обеспечения, обладающего механизмами самозащиты, и получены навыки динамического анализа вредоносного программного обеспечения.

Были изучены и описаны техники, применяемые в целевых атаках на основе классификации MITRE ATT&CK и механизмы защиты от динамического анализа. Была разработана программа, осуществляющая динамическое исследование анализируемого исполняемого файла с использованием средства бинарной инструментации Intel PIN и проведён динамический анализ исполняемых файлов APT.

Полученные по варианту задания APT реализовывают мало техник из рассматриваемых по варианту. Антиотладочные техники, можно сказать, вовсе не реализованы. Но выполнение лабораторной работы было полезным и интересным с точки зрения изучения работы с Intel PIN.

Приложение

Таблица 1 – Формат описания техник, применяемых в целевых атаках на основе классификации MITRE ATT&CK

Название техники	**Класс действий (Tactics)**	**Описание и используемые функции**	**Выявление (Detection) и противодействие (Mitigation)**	**Примеры APT**
Execution through API	Выполнение	Средства злоумышленника могут напрямую использовать интерфейс Windows (API) для выполнения двоичных файлов. Такие функции, как Windows API CreateProcess, позволяют программам и сценариям запускать другие процессы с собственными параметрами пути и аргументов: • CreateProcessA() and CreateProcessW(), • CreateProcessAsUserA() and CreateProcessAsUserW(), • CreateProcessInternalA() and CreateProcessInternalW(), • CreateProcessWithLogonW(), CreateProcessWithTokenW(), • LoadLibraryA() and LoadLibraryW(), • LoadLibraryExA() and LoadLibraryExW(), • LoadModule(), • LoadPackagedLibrary(), • WinExec(), • ShellExecuteA() and ShellExecuteW(),	Мониторинг вызовов API может генерировать значительный объем данных и может оказаться бесполезным для защиты, если только он не собран при определенных обстоятельствах, поскольку безопасное использование функций Windows API, таких как CreateProcess, является распространенным явлением и его трудно отличить от злонамеренного поведения. Корреляция других событий с поведением, связанным с вызовами функций API с использованием мониторинга API, предоставит дополнительный контекст событию, который может помочь определить, связано ли это со злонамеренным поведением. Корреляция активности по линии процесса с идентификатором процесса может быть достаточной. Для защиты необходимо идентифицировать и блокировать потенциально вредоносное программное обеспечение, которое	ADVSTORESHELL, APT37, BADNEWS, BANKSHOT,Cobalt Strike, Empire, Gorgon Group, HAWKBALL, HyperBro, InnaputRAT, LightNeuron, Mosquito, Plugx, Silence, SyncAck, TrickBot, Turla, Ursnif, Volgmer, XAgentOSX

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
		• ShellExecuteExA() and ShellExecuteExW()	может быть выполнено с помощью этого метода, с помощью таких инструментов, как белый список приложений, например, Контроль приложений Защитника Windows, AppLocker или Политики ограниченного использования программ.	
Process Hollowing	Обход средств защиты	Приостановка процесса, когда он создан в приостановленном состоянии, а его память не отображена и заменяется вредоносным кодом. Подобно технике внедрения процесса, выполнение вредоносного кода маскируется настоящим процессом и может обойти защиту и обнаруживающий анализ. • WriteProcessMemory • ZwUnmapViewOfSection • NtUnmapViewOfSection	Мониторинг API может генерировать значительный объем данных и может оказаться бесполезным для защиты, если только он не собран при определенных обстоятельствах для известных опасных последовательностей вызовов, поскольку безопасное использование функций API может быть обычным явлением и его трудно отличить от злонамеренного поведения. Следует отслеживать вызовы API, которые отменяют отображение памяти процесса, такие как ZwUnmapViewOfSection, и те,	Astaroth, Azorult, BADNEWS, Bandook, BBSRAT, Cobalt Strike, Duqu, Gorgon Group, ISMInjector, menuPass, Orz, Patchwork, Smoke Loader, Ursnif

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
			которые можно использовать для изменения памяти в другом процессе, такие как WriteProcessMemory. Отслеживать действия процесса на наличие необычных (открытие сетевых соединений, чтение файлов и другие, не характерные для этого процесса).	
Input Capture	Сбор учетных данных	Злоумышленники могут использовать методы сбора пользовательского ввода для получения учетных данных для учетных записей и сбора информации, включающей в себя журнал ключей и ввод пользователя. Чаще всего это логирование нажатия клавишь, с множеством видов перехвата. • SetWindowsHook • GetKeyState • GetAsyncKeyState	Кейлоггеры могут принимать различные формы, включая изменение реестра и установку драйвера, установку перехвата или опрос для перехвата нажатий клавиш. Обычно используемые вызовы API включают SetWindowsHook, GetKeyState и GetAsyncKeyState. [1] Мониторинг реестра и файловой системы на наличие таких изменений и обнаружение установки драйверов, а также поиск общих вызовов API-интерфейсов ведения блога. Одни только вызовы API не являются индикатором регистрации ключей, но могут	ADVSTORESHELL, Agent Tesla, APT28, APT3, APT38, Astaroth, BADNEWS, Cardinal RAT, Cobalt Strike, DOGCALL, FIN4, jRAT, Matroyshka, OilRig, PowerSploit, Remsec, ROKRAT, Stolen Pencil,

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
			предоставлять поведенческие данные, которые полезны в сочетании с другой информацией, такой как новые файлы, записанные на диск, и необычные процессы. Мониторинг реестра для добавления поставщика пользовательских учетных данных. [2] Обнаружение скомпрометированных действительных учетных записей, используемых злоумышленниками, может помочь отследить результат перехвата ввода пользователем, если используются новые методы. Этот тип техники атаки не может быть легко смягчен профилактическими мерами, поскольку он основан на злоупотреблении системными функциями.	Threat Group-3390, Zeus Panda
Domain Trust Discovery	Обнаружение	Злоумышленники могут попытаться собрать информацию о доверительных отношениях в домене, которая может использоваться для определения возможностей бокового перемещения в многодоменных / лесных средах Windows. Доменные доверительные отношения обеспечивают механизм для домена, чтобы позволить доступ к ресурсам на основе	Методы обнаружения системы и сети обычно происходят во время операции, когда злоумышленник изучает окружающую среду. Данные и события следует рассматривать не изолированно, а как часть цепочки поведения, которая может привести	Dsquery, Empire, Nltest, PoshC2, PowerSpoit, TrickBot.

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
		процедур аутентификации другого домена. [1] Доверие к домену позволяет пользователям доверенного домена получать доступ к ресурсам в доверяющем домене. Обнаруженная информация может помочь противнику провести SID-History Injection, Pass the Ticket и Kerberoasting. [2] [3] Доменные доверительные отношения могут быть перечислены с помощью DSEnumerateDomainTrusts () вызова Win32 API, методов .NET и LDAP. [3] Известно, что утилита Windows Nltest используется злоумышленниками для перечисления доверительных отношений домена. [4] GetAllTrustRelationships ()	к другим действиям, основанным на полученной информации. Мониторинг процессов и аргументов командной строки для действий, которые могут быть предприняты для сбора системной и сетевой информации, такой как nltest / domain_trusts. Инструменты удаленного доступа со встроенными функциями могут напрямую взаимодействовать с Windows API для сбора информации. Ищите вызов Win32 API DSEnumerateDomainTrusts () для определения активности, связанной с обнаружением доверия домену. [3] Информация также может быть получена с помощью инструментов управления системой Windows, таких как PowerShell. Метод .NET GetAllTrustRelationships () может быть индикатором обнаружения доверия доменов. [11] Меры противодействия: Сопоставить пути в существующих доменах / лесах и свести к минимуму доверительные отношения.	

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
			Использовать сегментацию сети для важных доменов.	
Service Stop	Влияние на систему	Злоумышленники могут останавливать или отключать службы в системе, чтобы сделать эти службы недоступными для законных пользователей. Остановка критически важных служб может помешать или остановить реагирование на инцидент или помочь в достижении общих целей противника, нанося ущерб окружающей среде. [1] [2] Злоумышленники могут достичь этого, отключив отдельные службы, имеющие большое значение для организации, такие как MSExchangeIS, что сделает контент Exchange недоступным [2]. В некоторых случаях злоумышленники могут остановить или отключить многие или все службы, чтобы сделать системы непригодными для использования. [1] Сервисы могут не разрешать изменение своих хранилищ данных во время работы. Злоумышленники могут остановить службы для проведения уничтожения данных или данных, зашифрованных для воздействия на	Следите за процессами и аргументами командной строки, чтобы увидеть, завершены ли критические процессы или остановлены. Мониторинг изменений в реестре для модификаций служб и программ запуска, которые соответствуют службам высокой важности. Ищите изменения в записях реестра службы, которые не связаны с известным программным обеспечением, циклами исправлений и т. Д. Информация о службе хранится в реестре по адресу HKLM \ SYSTEM \ CurrentControlSet \ Services. Изменения двоичного пути службы или типа запуска службы, измененного на отключенный, могут быть подозрительными. Инструменты удаленного доступа со встроенными функциями могут напрямую взаимодействовать с API Windows для выполнения этих	Lazarus Group, Olympic Destroyer, WannaCry

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
		хранилища данных таких служб, как Exchange и SQL Server. [3] ChangeServiceConfigW	функций вне обычных системных утилит. Например, ChangeServiceConfigW может использоваться злоумышленником для предотвращения запуска служб. [1] Методы противодействия: • Network Segmentation • Restrict File and Directory permissions • Restrict registry Permissions • User Account Management	
Disabling Security Tools	Обход средств защиты	Злоумышленники могут отключить средства безопасности, чтобы избежать возможного обнаружения их инструментов и действий. Это может принимать форму уничтожения программного обеспечения безопасности или процессов регистрации событий, удаления разделов реестра, чтобы инструменты не запускались во время выполнения, или других методов, мешающих сканированию безопасности или созданию отчетов о событиях.	Следите за процессами и аргументами командной строки, чтобы увидеть, убиты ли средства безопасности или они перестают работать. Мониторинг изменений в реестре на наличие изменений в службах и программах запуска, соответствующих средствам безопасности. Отсутствие отчетов журнала или файла событий может быть подозрительным. Методы защиты: Restrict File and Directory Permissions User Account Management	Agenta Tesla, BACKSPACE, BADCALL, Brave Prience, Lazarus Group, LockerGoga, NanHaiShu, NanoCore, Putter Panda, Remsec, RunningRAT, Turla, Unknown Logger

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
Execution Guardrails	Обход средств защиты	Данный тип ограничивает выполнение или действия, основываясь на специфических условиях среды, предоставленной противником, которые, как ожидается, будут присутствовать на цели. Ограничения гарантируют, что полезная нагрузка выполняется только против намеченной цели и уменьшает побочный ущерб кампании. Такие ограничения могут включать специфичные имена сетей, присоединенные физические устройства, файлы, домены AD, и IP адреса. GetEnvironmentVariable	Обнаружение маниполяции окружением окружения может быть затруднено в зависимости от реализации. Мониторинг для выявления подозрительных процессов, которые собирают различную системную информацию или выполняют другие формы обнаружения, особенно в течение короткого периода времени, может помочь в обнаружении.	PowerSploit; Reg; TrickBot
File Permissions Modification	Обход средств защиты	Отключение средств защиты. Злоумышленники могут отключать различные средства безопасности, уничтожать процессы журналирования событий, ключи реестра, чтобы средства безопасности не запускались во время вредоносной активности, или применять иные способы вмешательства в работу сканеров безопасности или отчеты о событиях. RegOpenKeyExA RegOpenKeyExW RegOpenKeyA RegCreateKeyExA RegCreateKeyExW RegCreateKeyA	Обеспечение корректной настройки прав доступа к процессам, реестру и файлам, для предотвращения несанкционированного отключение или вмешательства в работу средств безопасности. Перехват попыток отключить средства защиты при помощи утилиты sc и доступа к средствам, содержащих слова: defender, antivirus, protect, security.	APT33, Equation

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
LLMNR/NBT-NS Poisoning and Relay	Получение учётных данных	Link-Local Multicast Name Resolution (LLMNR) и служба имен NetBIOS (NBT-NS) - это компоненты Microsoft Windows, которые служат альтернативными методами идентификации хоста. LLMNR основан на формате системы доменных имен (DNS) и позволяет узлам на той же локальной ссылке выполнять разрешение имен для других узлов. NBT-NS идентифицирует системы в локальной сети по их имени NetBIOS. Злоумышленники могут подделать авторитетный источник для разрешения имен в сети жертвы, отвечая на трафик LLMNR (UDP 5355) / NBT-NS (UDP 137), как если бы они знали личность запрошенного хоста, эффективно отравляя службу, чтобы жертвы могли общаться с системой, контролируемой противником. socket; listen; accept; connect;	Проверьте HKLM \ Software \ Policies \ Microsoft \ Windows NT \ DNSClient на наличие изменений в значении DWORD «EnableMulticast». Значение «0» указывает, что LLMNR отключен. [13] Отслеживайте трафик через порты UDP 5355 и UDP 137, если LLMNR / NetBIOS отключен политикой безопасности. Разверните LLMNR / NBT-NS инструмент обнаружения спуфинга. [14] Мониторинг журналов событий Windows для идентификаторов событий 4697 и 7045 может помочь в обнаружении успешных методов ретрансляции. [4] Методы противодействия: Disable or Remove Feature or Program (disable LLMNR, NetBIOS) Filter Network Traffic (block LLMNR and NetBIOS traffic)	Empire, impacket, PoshC2, Pupy, Responder
Credential Dumping	Получение учётных данных	Сброс учетных данных - это процесс получения информации об имени пользователя и пароле учетной записи, обычно в форме хеш-кода или открытого	Обычные дамперы учетных данных, такие как Mimikatz, получают доступ к процессу службы	APT1, APT28, APT3, APT32, APT33, APT37, APT39, Astarotb, Cleaver, Cobalt Strike,

Название техники	Класс действий (Tactics)	Описание и используемые функции	Выявление (Detection) и противодействие (Mitigation)	Примеры APT
		текстового пароля, из операционной системы и программного обеспечения. Затем можно использовать учетные данные для выполнения бокового перемещения и доступа к ограниченной информации. Некоторые из инструментов, упомянутых в этом методе, могут использоваться как противниками, так и профессиональными тестерами безопасности. Дополнительные пользовательские инструменты, вероятно, также существуют. SAM, Cached Credentials, LSA, NTDS, Group Policy Preferences, SPNs, DCSync. WriteFile ReadFile	подсистемы LSA (LSASS), открывая процесс, находя секретный ключ LSA и расшифровывая разделы в памяти, где хранятся данные учетных данных. Дамперы учетных данных могут также использовать методы для отражения процесса внедрения, чтобы уменьшить потенциальные признаки вредоносной активности. • Active Directory Configuration • Credential Access Protection • Operating System Configuration • Password Policies • Privileged Account Management • Privileged Process Integrity • User Training	Dragonfly 2.0, GreyEnergy, Lazarus Group, Matroyshka, menuPass, OilRig, Olympic Destroyer, Patchwork, PoshC2, PowerSploit, Pupy, Soft Cell, Strider, Unknown Logger

Таблица 2 – Известные механизмы защиты от динамического анализа

Название техники	Описание техники	Используемые функции и инструкции	Используемые аргументы и имена объектов	Способ выявления и противодействия
Антиотладочные техники				
Анализ PEB структуры	PEB — это закрытая структура, используемая внутри операционной системы. Создается при создании процесса и содержит всю необходимую информацию для работы процесса.	IsDebuggerPresent __readgsqword __readfsdword NtGlobalFlag heapForceFlags TrapFlag CheckRemoteDebuggerPresent NtQueryInformationProcess	 0x0C * 8 0x0C * 4 0x70, 0x14, 0x40, 0x0C, 0x74, 0x18, 0x44, 0x10 Pushfd, 0x100 PID of current process PID of current process, 0x07, 0x1E, 0x1F, 0x00	Вернуть 0 Устновить бит NtGlobalFlag в 0 Установить TF в 0 Вернуть 0 Вернуть значение, отличное от 0
Использование точек остановки	Анализ поведения процесса при использовании точек остановки и возникновении исключений.	Int GetThreadContext	1 (TF), 3 ctx->Dr0 != 0 ctx->Dr1 != 0 ctx->Dr2 != 0 ctx->Dr3 != 0	Изменить содержимое контекста
Скрытие потока от отладчика	Скрытие потока от отладчика	NtSetInformationThread NtCreateThreadEx	PID текущего процесса, 0x11 0x04	Не передавать управление оригинальной функции NtSetInformationThread
Использование отладочных сообщений	Начиная с Windows 10 вызов функции OutputDebugString был	OutputDebugString	0x40010006, 0x4001000A	Обработать функцию без генерации исключения

Название техники	Описание техники	Используемые функции и инструкции	Используемые аргументы и имена объектов	Способ выявления и противодействия
	изменен на вызов RaiseException с определенными аргументами.			
Анализ запущенных в системе процессов	Поиск списка запущенных процессов и их сравнение имён с сигнатурами отладчиков.	Process32Next EnumProcesses	Название отладчика	Скрыть процесс
Анализ загруженных в процесс модулей	Выполняется поиск отладочных библиотек.	GetModuleFileNameEx	Название библиотеки	Скрыть библиотеку
Анализ родительского процесса.	Проверяется родительский процесс на причастность к отладчику.	GetWindowThreadProcessId NtQueryInfoProcess GetWindowTextA	Сигнатуры	Скрытие информации
Выявление виртуальных машин и песочниц				
Обнаружение артефактов	Поиск следов запуска в виртуальной среде или отладчике.	FindNextFile GetMotherBoardSerialNumber __cpuid RegOpenKeyEx GetFileAttributes GetAdaptersAddresses ExecWMIQuery	VMtools.exe, Vmwareuser.exe, vboxservice.exe 7, 3 1, 40000000h Сигнатуры	Скрытие информации Вернуть число, отличное от сигнатурных Скрытие информации
Использование невалидных опкодов	Гипервизоры часто поддерживают специальные инструкции, которые недоступны на реальных физических машинах.	0F 3F 07 OB (detect VirtualPC)		Изменение гипервизором
Наличие функций для работы с IDT и GDT	Анализ реакции системы на выполнение специфичных ассемблерных инструкций. Большинство из этих методов уже не	SIDT/SGDT/SLDT		Изменение гипервизором

Название техники	**Описание техники**	**Используемые функции и инструкции**	**Используемые аргументы и имена объектов**	**Способ выявления и противодействия**
	работают (SIDT/SGDT/SLDT, STR, SMSW).			
Сбор информации о «возрасте» системы	Получение информации о количестве файлов, папок, скачанных файлов, куков, записи в реестре, свободном месте на ЖД в системе и т.д.	WlanEnumInterfaces GetDiskFreeSpace RegEnumKeyEx GetSystemInfo GlobalMemoryStatusEx		Скрытие информации
Выявление средств защиты и анализа				
Выявление с помощью WMI	Сбор информации об установленных средствах защиты с помощью WMI-запросов.	ExecQuery	AntiVirusProduct, FirewallProduct. AntiSpywareProduct	Изменение результата
Выявление на основе следов в системе	Анализ следов, оставляемых в системе: наличие директорий и файлов, ключей реестра, запущенных процессов.	RegOpenKeyEx FindNextFile Process32Next	Сигнатуры	Изменение результата
Выявление перехватчиков	Анализ опкодов инструкций адресов критических функций для выявления перехватчиков.	BYTE *b = (BYTE *) { DeleteFile, ShellExecuteEx, CreateProcess …}	(*b == 0x8b) && (*(b+1) == 0xff) ? FALSE : TRUE;	Перехват доступа к памяти или задание для страницы опции PAGE_EXECUTE

Таблица 3 – Исследование атак

Название APT / операции	**Название компонента**	**Обнаруженные действия**
DPRK	WormSMB2.0.exe	Из исследуемых направлений использовался только Execution through api, в плане создания и запуска своих сервисов.
GreyEnergyAPT	Greyenergydropper.doc	Это приложение выполняет Credential dumping, Execution through api, и domain discovery.