1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт кибербезопасности и защиты информации

**ЛАБОРАТОРНАЯ РАБОТА № 1**

**Изучение сценариев целенаправленных атак и используемого в них программного обеспечения**

по дисциплине «Методы и средства противодействия целенаправленным атакам»

1. Выполнила
2. студентка гр. 5151003/90801 Кулеева А.Г.

1. Руководитель
2. доцент, к.т.н. Жуковский Е.В.
3. Санкт-Петербург
4. 2023

# **Цель**

Изучение сценариев известных целенаправленных атак (APT, Advanced persistent threats), рассмотрение способов их обнаружения и предотвращения. Изучение вредоносного программного обеспечения (ВПО), используемого в целенаправленных атаках.

# Формулировка задания

В ходе выполнения лабораторной работы необходимо выполнить следующие действия:

1. Получить у преподавателя вариант задания (таблица 1), состоящий из названия группы, реализующей целенаправленные атаки (APT). Получить ссылку на исследуемые файлы, связанные с указанной атакой (при наличии пароля на архиве: infected/malware). В ходе выполнения лабораторной работы необходимо изучить не менее 3 **исполняемых** **файлов** (.exe, .dll, .sys и т.п.), используемых в указанной APT компании. Вместо указанных по ссылке, по согласованию с преподавателем, могут исследоваться другие вредоносные файлы [4-15], относящиеся к указанной APT-группе.
2. При проведении исследования ОБЯЗАТЕЛЬНО (!) все взаимодействия с изучаемыми файлами проводить в виртуальной машине (ВМ) с отключенными средствами интеграции с хостовой системой, включая разделяемые директории [1]. Не подключайте к ВМ никакие USB-накопители. В случае, если не исследуется сетевое взаимодействие экземпляра следует также отключить сетевой интерфейс в ВМ. При наличии сетевого интерфейса рекомендуется не использовать соединение типа мост (bridge), чтобы исследуемое ПО не имело доступ напрямую в вашу локальную сеть. Особо следует быть осторожным с файлами Microsoft Office, архивами и lnk-файлами. Так как эти средства эксплуатации уязвимостей, которые могут быть запущены случайно. Рекомендуется создать снимок (snapshot) чистой системы с установленными средствами анализа до начала исследования.
3. Исследовать существующие публикации, описывающие указанную целенаправленнуюатаку и инструменты, используемые в ней. Представить в отчете описание рассматриваемых атак.
4. Провести предварительный динамический анализ с использованием изолированных сред динамического анализа ВПО (malware sandbox), например, Cuckoo Sandbox [2] / CAPE2 [3], либо их аналогов в виде онлайн-сервисов [4]. Получить начальную информацию о поведении и назначении исполняемых файлов.
5. Провести углубленный динамический и статический анализ полученных экземпляров ВПО с целью определения наиболее опасных функциональных возможностей в нем. С помощью утилит мониторинга изменений в системе (например, SysTracer, Total Uninstall, Sandboxie, RegShot) отследить изменения в системе после запуска исследуемых файлов (драйверы, службы, реестр, файлы). Используя утилиты ProcessMonitor и WinAPIOverride/WinAPI Monitor отследить действия, выполняемые исследуемым ПО. Используя утилиту ProcessHacker изучить создаваемые ВПО процессы и их ресурсы (загруженные библиотеки, открытые файлы, объекты синхронизации, сетевые порты). Для определения назначения исполняемых файлов и их алгоритмов работы стоит использовать дизассемблеры (IDA Pro/Ghidra) и отладчики (x64dbg/Immunity Debugger/WinDBG) и WinAPI-мониторы (WinAPIOverride, WinAPI Monitor).
6. Описать принцип работы исследованных компонентов. Определить и изучить код, связанный с осуществлением следующих действий, соответствующих классификации MITRE ATT&CK [3]:
   1. закрепление в системе (добавление в автозагрузку, заражение существующих файлов, установка своих компонентов, установка служб);
   2. повышение привилегий (эксплуатация уязвимостей, запуск специальных инструментов);
   3. сбор информации о системе (пользователи, установленное ПО, недавно используемые программы и файлы и т.д.);
   4. поиск средств защиты в системе (анализ процессов, файлов/директорий, реестра);
   5. сетевая активность (сбор информации об устройстве сети, сетевые атаки);
   6. реализация сетевых атак (MITM);
   7. эксплуатация уязвимостей (CVE, описание уязвимости);
   8. связь с управляющим (C&C) сервером (получение / отправка информации);
   9. поиск целевой информации в системе (название и содержание файлов);
   10. доступ к критической информации (кража аутентификационной информации: пароли, история браузера и других приложений);
   11. реализация вредоносных действий (майнинг, разрушающее воздействие, шифрование);
   12. взаимодействие с информацией, хранимой другими приложениями (браузеры, почта);
   13. изменение конфигурации системных и прикладных приложений (RDP, межсетевой экран, Windows Defender и т.д.).
7. Привести фрагменты ассемблерного / псевдокода (восстановленного кода на языке С) наиболее важных функций ВПО, выявленных в соответствие с п. 5 задания.
8. Представить результаты изучения, используемого в APT программного обеспечения в формате таблицы 2.
9. Описать в отчете также следующую информацию касательно исследуемой APT-атаки:
   1. цели атак;
   2. типовые сценарии атаки;
   3. описание используемых уязвимостей;
   4. описание используемых сторонних средств (например, mimicatz, PSExec и т.д.);
   5. техники, используемые для обхода средств защиты и закрепления в системе;
   6. другая важная информация.

# Ход работы

## Описание АРТ группировки

В соответствии с вариантом был получен файл Winnti [1]. Среди множества имеющихся вредоносов случайным образом был выбран 4af18b2314609db33db030b963aa1d9a43cbb627472ed8f9235ef476236ad7b6.

Для понимания потенциальных угроз, которые может нести в себе файл изучим информацию об APT группировке. На просторах интернета был найден отчет Лаборатории Касперского [2]. Хакерская группировка Winnti имеет китайское происхождение. Она действует уже на протяжении нескольких лет и специализируется на проведении атак на компании, занимающиеся компьютерными онлайн-играми. Основной целью атакующих является кража исходных кодов игровых проектов этих компаний и цифровых сертификатов. Кроме того, их интересуют данные об организации сетевых ресурсов (включая игровые серверы) и новых разработках: концептах, дизайне и т.п.

Осенью 2011 года на компьютерах огромного числа игроков одной из популярных онлайн-игр обнаружился троянец. Как выяснилось, эта вредоносная программа попала на компьютеры пользователей вместе с обновлениями игры, пришедшими с официального сервера обновлений. Многие даже заподозрили игровую компанию в ведении слежки за игроками. Однако вредоносная программа попала на компьютеры пользователей по ошибке — мишенью злоумышленников были не игроки, а компании, занимающиеся разработкой и выпуском компьютерных игр.

При анализе вредоносной программы, которую сотрудники компании обнаружили на сервере обновлений, оказалось, что это библиотека dll, скомпилированная под 64-битную систему Windows. И она содержала корректно подписанный вредоносный драйвер! Сам по себе вредоносный модуль оказался первым троянцем с легальной цифровой подписью для 64-битной версии Windows. Все предыдущие аналогичные инциденты с использованием подписанных файлов, относились к 32-битным приложениям.

Обнаруженная DLL имела функционал бэкдора, точнее полноценного RAT (Remote Administration Tool — средства удаленного управления), который скрытно от пользователя предоставлял возможность злоумышленникам управлять зараженным компьютером.

Дополнительную пикантность ситуации придавал тот факт, что была использована цифровая подпись другой игровой компании — KOG, расположенной в Южной Корее. После связи с компанией KOG, чей сертификат был использован для подписи файлов, а также компанией Verisign, выдавшую ей сертификат, и уведомлении обеих об инциденте сертификат был отозван.

Как выяснилось позже кража сертификатов у атакованных компаний и последующее их использование для новых атак, является фирменным стилем этой группы. За полтора года после инцидента удалось обнаружить около десятка таких краденых сертификатов. Кроме того, были обнаружены примеры использования этих сертификатов в атаках, проводимых другими группами китайских хакеров.

Были выделили три основные схемы монетизации, которые могли быть использованы группой Winnti.

* Нечестное накопление игровой валюты в онлайн-играх и перевод виртуальных средств в реальные деньги.
* Кража исходников серверной части онлайн-игр для поиска уязвимостей в играх — далее смотри пункт 1.
* Кража исходников серверной части популярных онлайн-игр для последующего развертывания пиратских серверов.

Приведем пример. В ходе расследования заражения одной из игровых компаний выяснилось, что зловред был создан для конкретной службы, запущенной на сервере этой компании. Вредоносная программа искала на сервере определенный работающий процесс, проникала в его код, искала по маске в коде процесса два определенных места и вставляла туда вызовы своих функций-перехватчиков. В функциях-перехватчиках вредоносная программа изменяла данные процесса, которые обрабатывались в этих двух местах, и возвращала управление обратно. Таким образом, злоумышленники влияли на серверный процесс, изменяя нормальный ход его исполнения.

Участники группы Winnti терпеливы и осторожны. Злоумышленники годами влияют на ход игр зараженных компаний, выкачивают оттуда деньги, но делают это так, чтобы не привлекать к себе внимание.

Рассматриваемая хакерская группа имеет особые черты, позволяющие выделить ее из всех прочих:

* широкое использование цифровых сертификатов: атакующие использовали сертификаты пострадавших от них компаний для атак на новые организации и кражи еще большего количества цифровых сертификатов;
* использование 64-битного драйвера руткита, подписанного легальной цифровой подписью;
* использование огромного количества публичных интернет-сервисов для размещения зашифрованных команд для управления зараженными системами;
* передача/продажа краденых сертификатов другим китайским группам хакеров, которые специализируются на других целях;
* кража исходных кодов и других видов интеллектуальной собственности компаний-разработчиков, занимающихся онлайн-играми.

## Предварительный динамический анализ (VirusTotal)

С помощью веб-ресурса VirusTotal был проведен предварительный анализ особенностей ВПО. Информация о файле была найдена по хэшу, 78% антивирусов определили файл как вредоносный, среди них McAfee, Kaspersky, Yandex, DrWeb, Avast (рисунок 1). По типу малваре файл был определен как троян или бэкдор, также некторые вендоры определили файл как часть семейства Winnti.

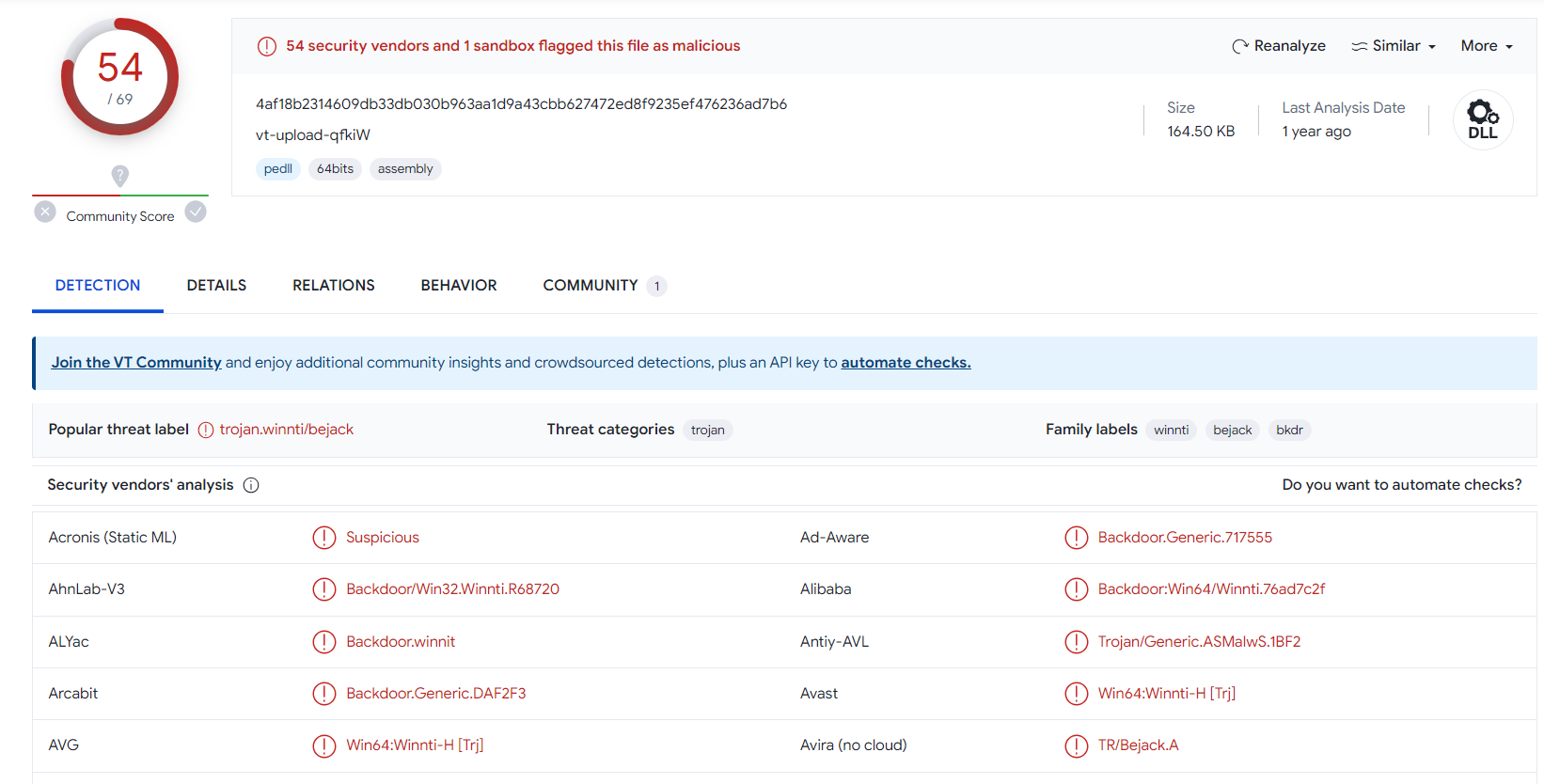


Рисунок ― Результаты анализа

На вкладке Details получим более подробную информацию о файле. Например, на рисунке 2 видим тип файла: executable windows file. Также имеется размер файла.

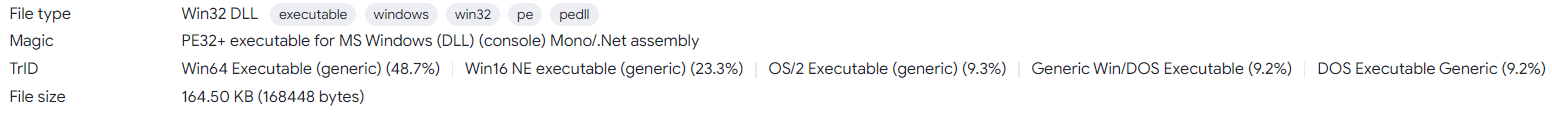


Рисунок ― Определение типа файла

Также на этой вкладке можно увидеть некоторые параметры компиляции, описание содержимых секций и импорты библиотек (рисунок 3). Импорты из Kernel32.dll позволяют использовать функции WinAPI, в частности создание процессов, работа с файлами, работа с переменными среды, манипуляции с кучей, инициализация критической секции, проверка наличия отладчика IsDebuggerPresent и другие.

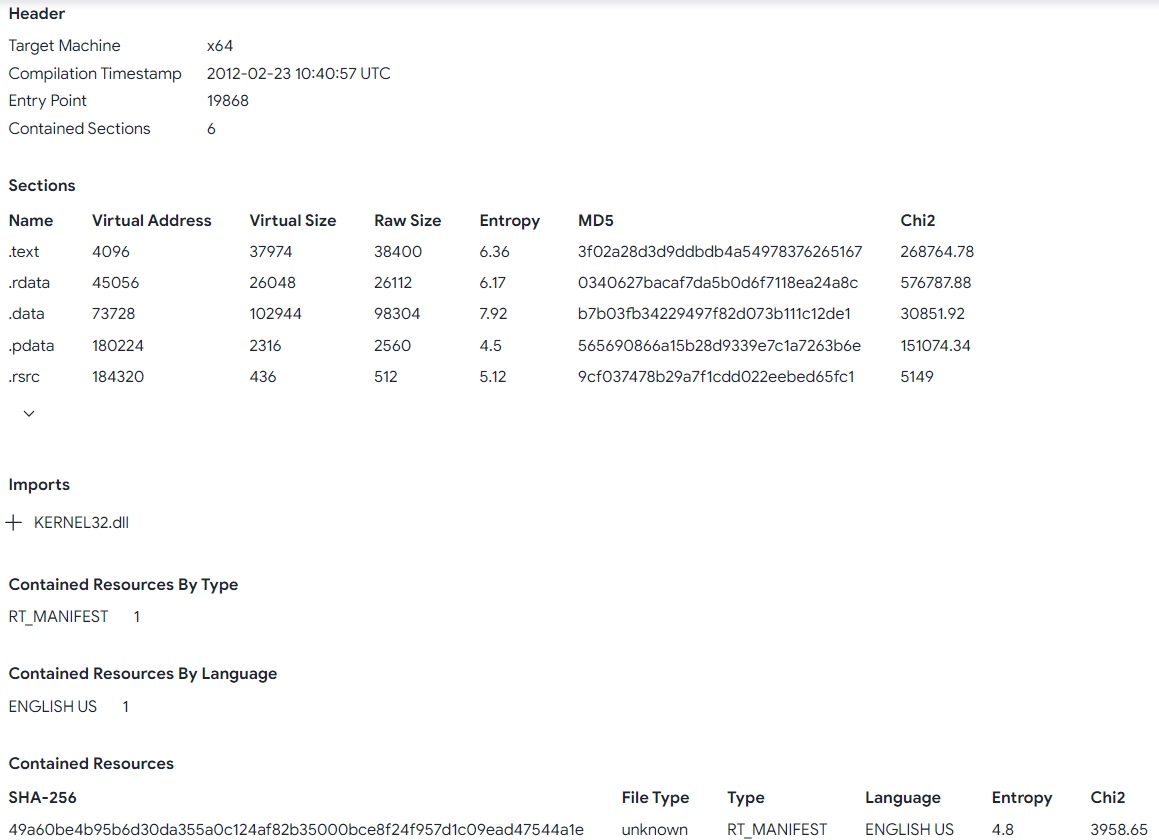


Рисунок ― Содержание вкладки Details

Также интересно рассмотреть вкладку Behavior, которая позволяет проанализировать поведение файла. На рисунке 4 можно увидеть файлы, ключи реестра и домены, с которыми взаимодействует ВПО. На рисунке 5 видим процессы, которыми манипулирует вредонос и загруженные модули.

DNS сервер в данном случае может быть игровым сервером, с которым установлена связь. Драйвер spliter.sys является компонентом режима ядра Microsoft Windows Driver Model (WDM) аудио компонентов.

Процесс rundll32.exe необходим для подключения в нужный момент нужных библиотек dll. Он как бы отыскивает и предоставляет доступ к нужным функциям тем программам, которым они требуются.

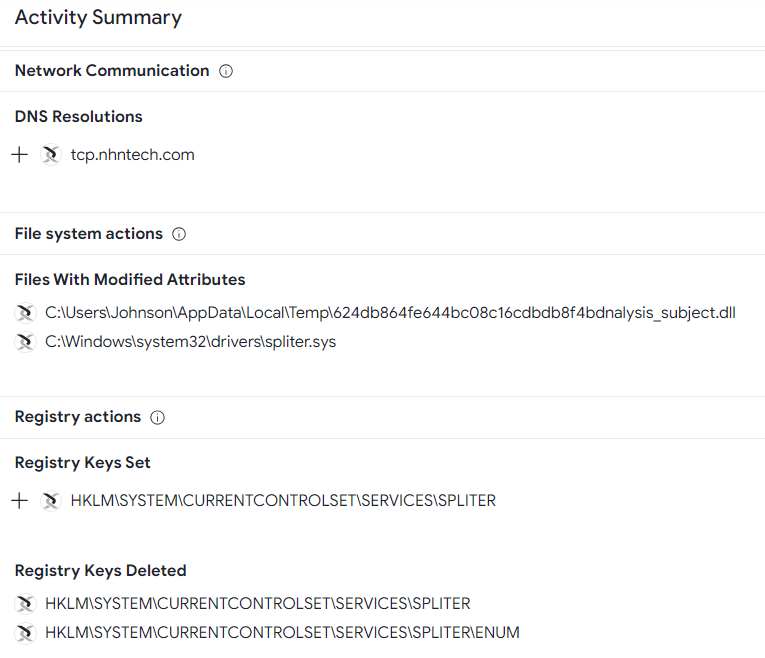


Рисунок ― Результат выполнения запроса

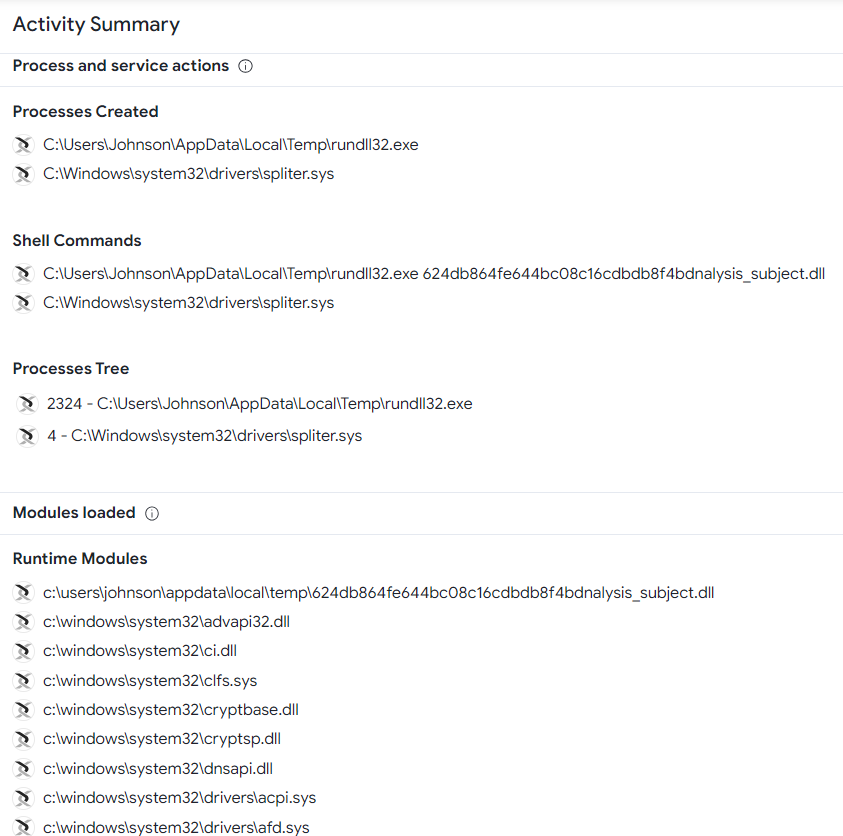


Рисунок ― Результат выполнения запроса

Зная историю и цели АРТ группировки и учитывая особенности малваре при анализе в песочнице, автор может предположить возможные действия трояна. ВПО вносит изменения в реестр, за счет чего изменяет поведение драйвера spliter.sys и судя по всему создает файл, чье название заканчивается как subject.dll, после чего этот файл «скармливается» процессу rundll32.exe и затем вызывается измененный драйвер spliter.

## Предварительный динамический анализ (Triage)

Далее файл был запущен в динамике в песочнице Triage. Это онлайн-песочница, с максимальным временем анализа до 30 минут и несколькими вариантами ОС на выбор. На рисунке 6 представлены настройки, которые были заданы перед запуском файла. Анализ на windows 7 дал больше информации, поскольку вирус достаточно старый, скорей всего он разрабатывался для xp или 7. Однако 7 версия очень сильно «тормозит», что затрудняет анализ. На 10 работать удобнее, однако там результаты несколько отличаются.

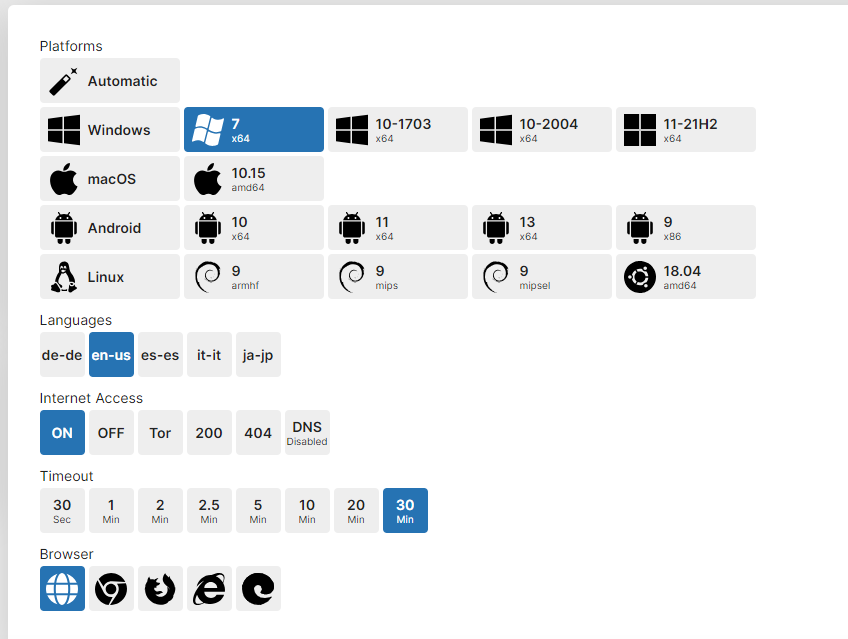


Рисунок — Настройки песочницы

При запуске в вм windows 10 задача сама завершилась через 2-3 минуты и сервис подготовил отчет (рисунок 7).

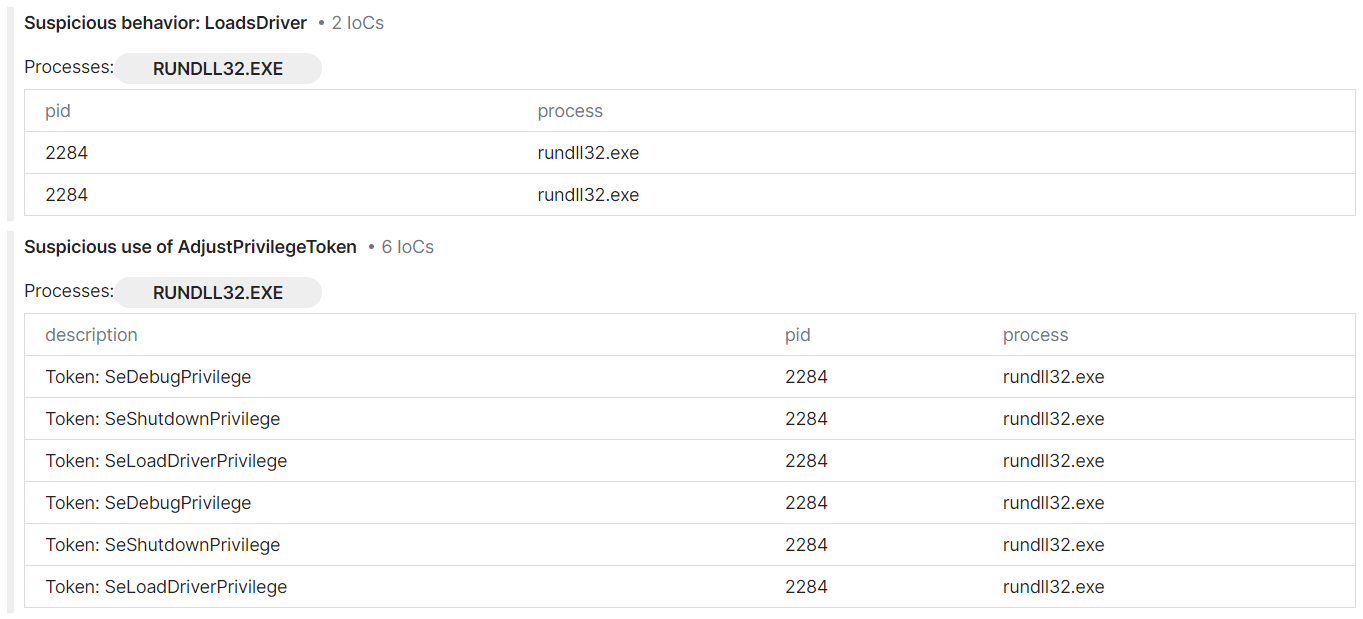
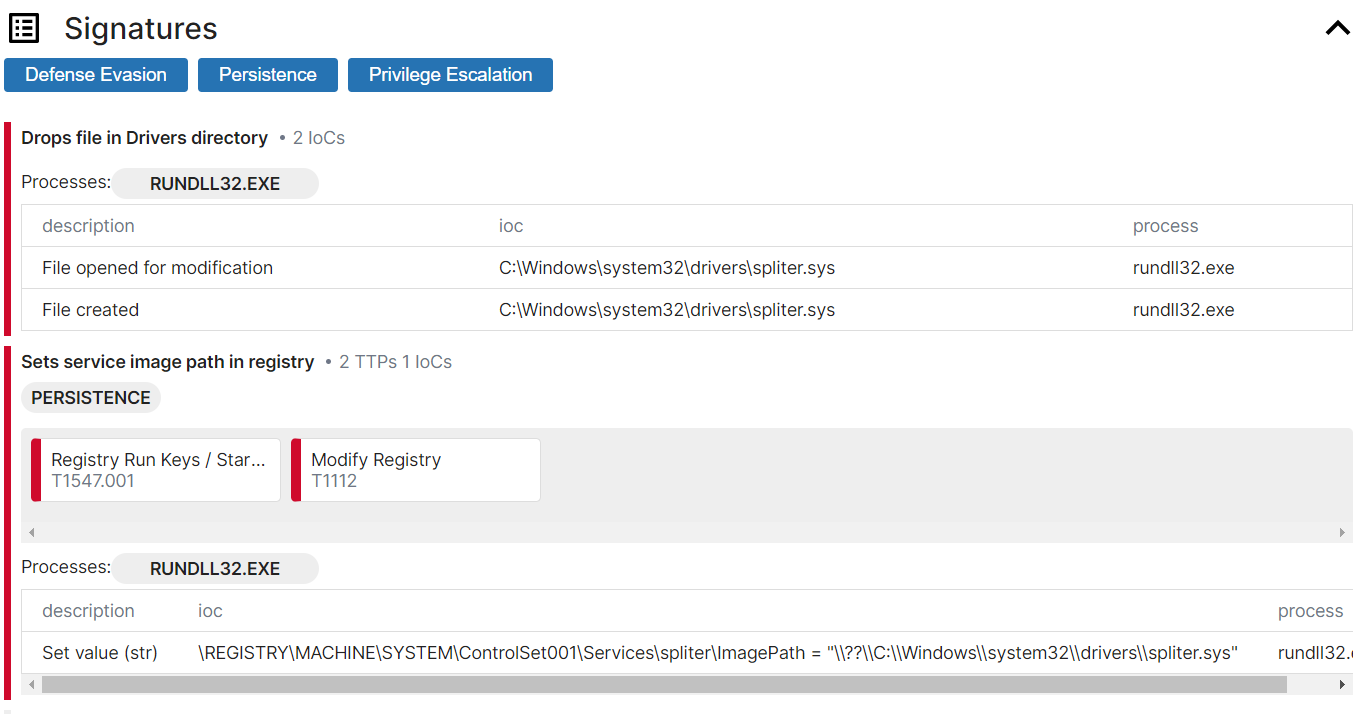


Рисунок — Анализ поведения (часть 1)

Рассмотрим рисунок 7. Здесь поведение совпадает с полученным в VirusTotal, однако уже есть более подробная информация. Вирус действительно модифицирует драйвер spliter.sys и прописывает в реестре путь до этого файла. Также описывается подозрительное поведение функции AdjustTokenPrivileges, которая включает или отключает привилегии в указанном [маркере доступа](https://learn.microsoft.com/en-us/windows/desktop/SecGloss/a-gly). Токен содержит информацию о безопасности для сеанса входа в систему. Система создает маркер доступа при входе пользователя в систему, и каждый процесс, выполняемый от имени пользователя, имеет копию токена. Он идентифицирует пользователя, группы пользователей и привилегии пользователя. Система использует токен для управления доступом к защищаемым объектам и для управления способностью пользователя выполнять различные системные операции на локальном компьютере. Существует два вида токена доступа: основной и олицетворяющий.

Также данная песочница составляет аналитику по MITRE ATT&CK (рисунок 8). И тут как раз описывается манипулирование процессом logon. Кроме того, описаны действия над реестром.

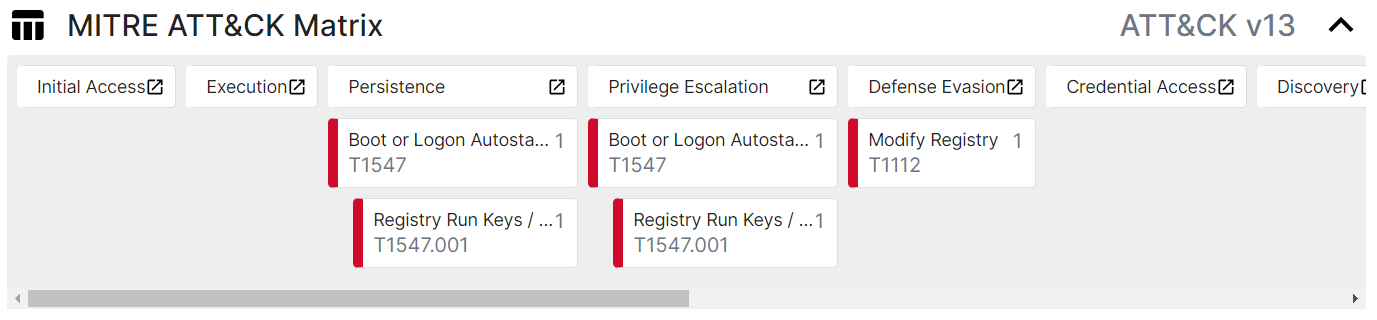


Рисунок 8 — Аналитика по MITRE ATT&CK

Песочница анализирует все запущенные пользователем процессы (рисунок 9). Каждый процесс можно раскрыть и посмотреть, какие действия он повлек (рисунки 10-17).

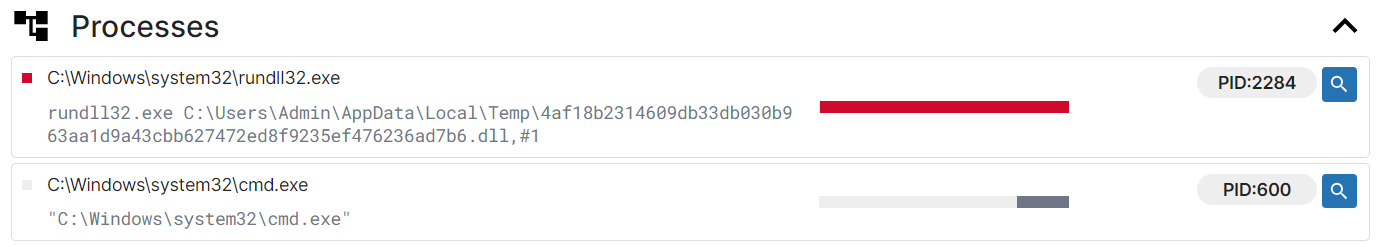


Рисунок — Анализ процессов

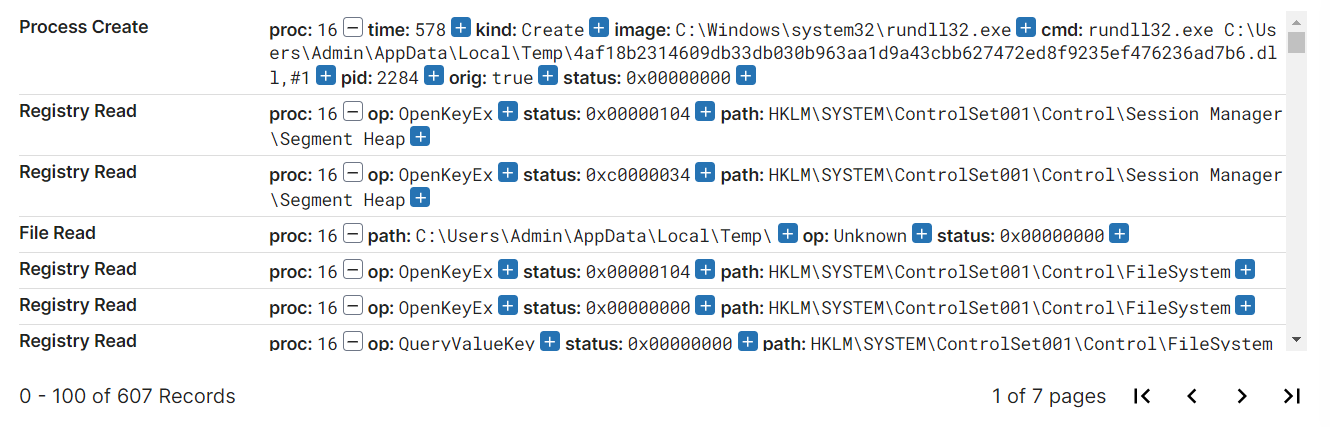


Рисунок — Действия, выполненные процессом (чтение реестра)

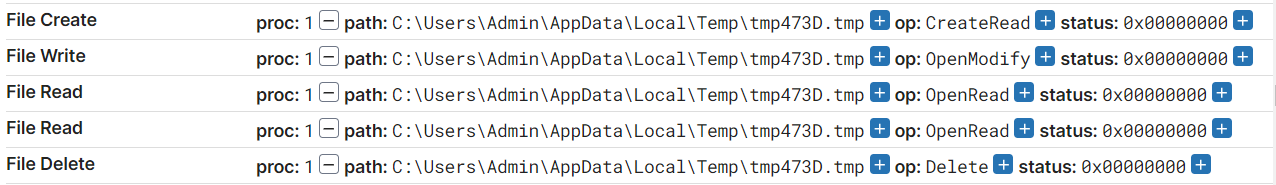


Рисунок — Действия, выполненные процессом (создание временного файла)

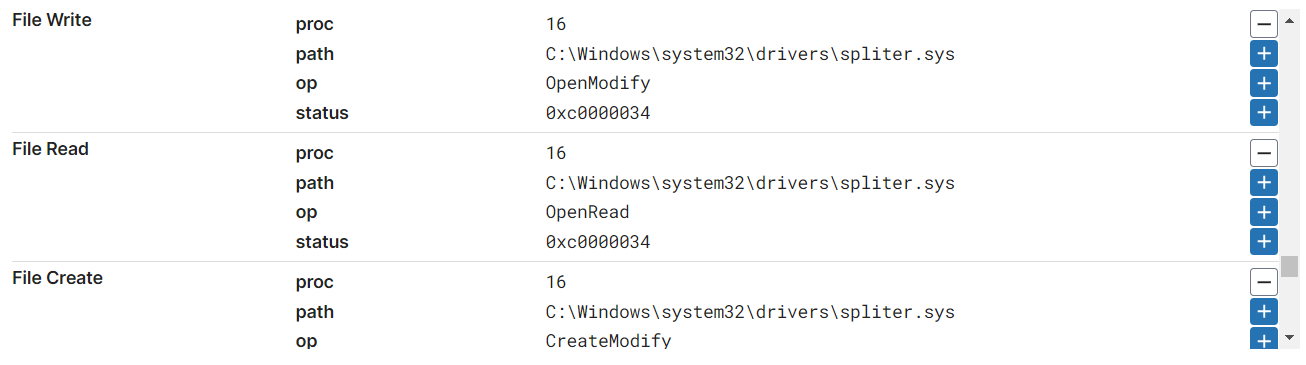


Рисунок 12 — Действия, выполненные процессом (модификация драйвера)

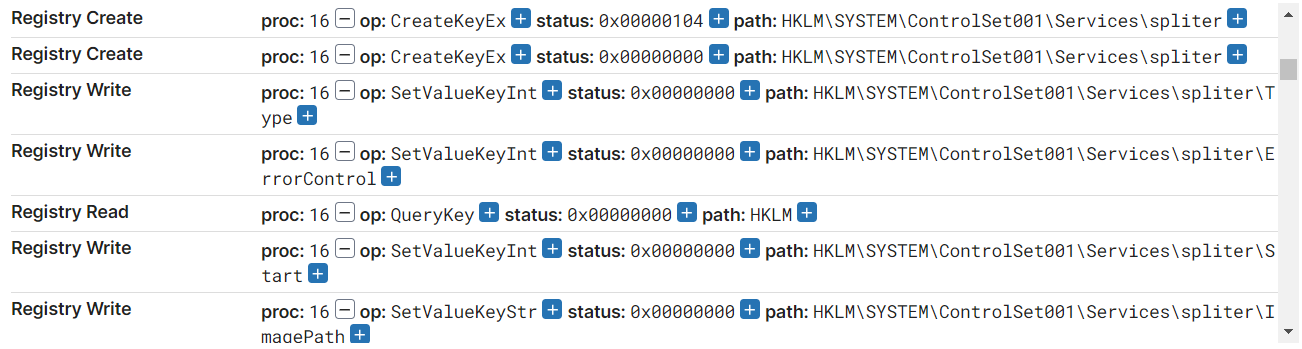


Рисунок 13 — Действия, выполненные процессом (настройка реестра для драйвера)

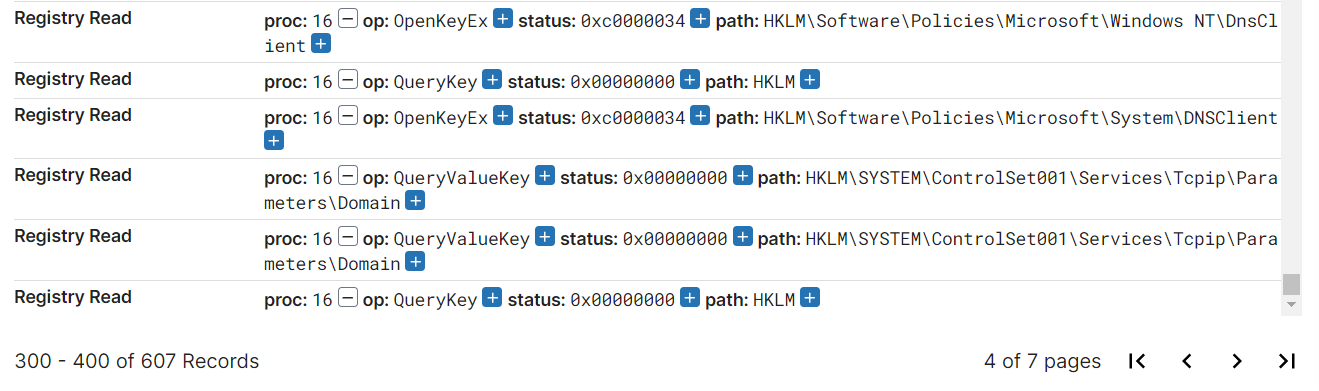


Рисунок 14 — Действия, выполненные процессом (сетевые взаимодействия)

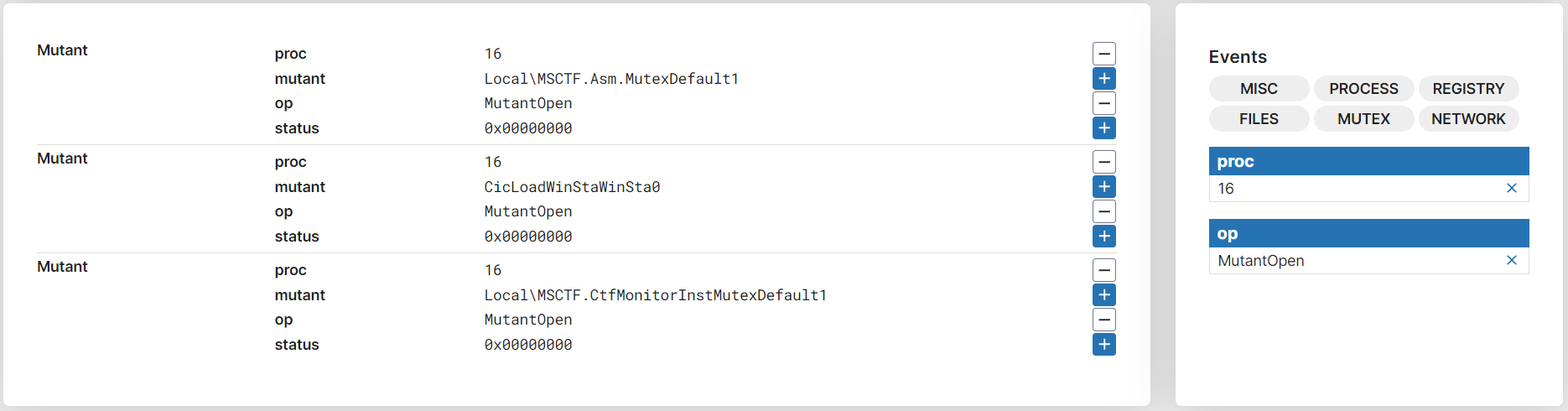


Рисунок 15 — Действия, выполненные процессом (мутанты)

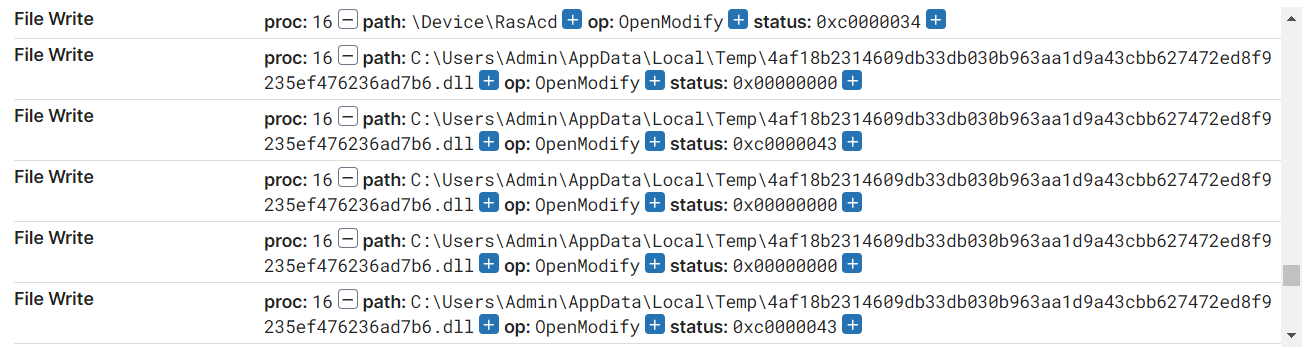


Рисунок 16 — Действия, выполненные процессом (запись в самого себя)

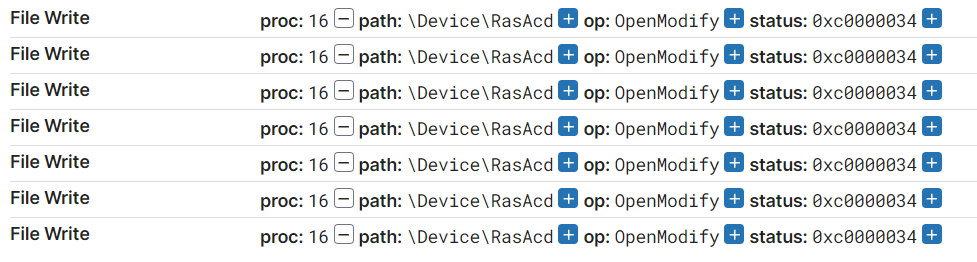


Рисунок — Действия, выполненные процессом (запись в RasAcd)

Как видно из рисунков, всего было выполнено более 600 действий, которые включают создание временного файла, модификация файла драйвера spliter.sys и настройка реестра для него. Большинство описанных действий — это чтение реестра: как сетевых компонентов, так и некоторых настроек машины. Также можем наблюдать запись в сам файл dll, возможно происходит инъекция, и запись в раздел Device/RasAcd. Кроме того, были выявлены действия типа мутант.

Файл rasacd.sys является программным компонентом драйвера автоматического подключения к удаленному доступу от Майкрософт. Rasacd.sys является драйвером ядра библиотеки динамических ссылок Windows и известен как драйвер автоматического подключения системы удаленного доступа (RAS), который обрабатывает удаленные подключения для различных запущенных сторонних программ.

В Windows Мутант — это объект ядра, который позволяет программам синхронизировать события между собой. Вредоносное ПО часто использует именованный мутант, чтобы гарантировать, что оно не заразит повторно тот же компьютер и не запустит только одну копию вредоносного ПО. Например, рассмотрим вредоносное ПО, которое передается через вредоносный документ Word. Каждый раз, когда документ открывается, вредоносная программа может без необходимости повторно заразить компьютер, увеличивая вероятность ее обнаружения. Чтобы избежать этого, вредоносная программа пытается открыть именованный мьютекс с заранее заданным жестко закодированным именем. Если [CreateMutex](https://docs.microsoft.com/en-us/windows/win32/api/synchapi/nf-synchapi-createmutexa) вызов завершится успешно, вредоносная программа может продолжить работу. Если вызов завершается неудачно, это, скорее всего, связано с тем, что уже запущена другая копия вредоносного ПО, поэтому вредоносное ПО завершит работу.

Кроме того, описаны сетевые взаимодействия (рисунок 18). При анализе на машине win10 взаимодействий было куда больше, однако в обоих случаях по пакетам нельзя определить что-либо значимое.

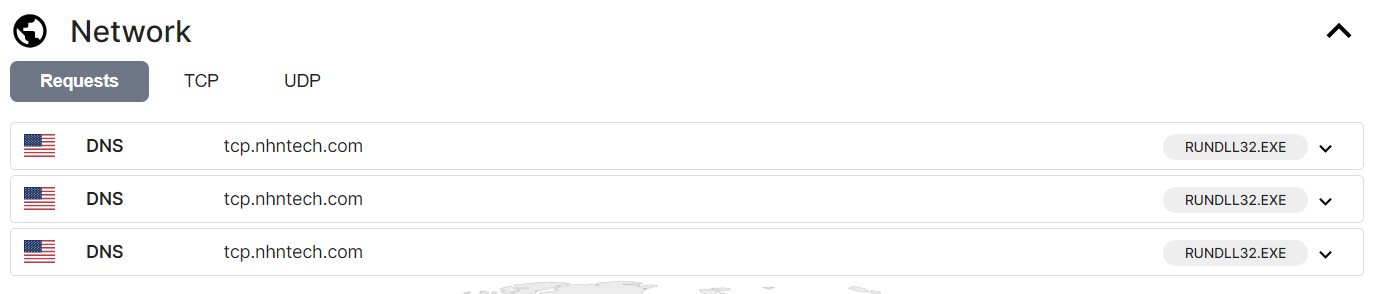


Рисунок 18 — Сетевое взаимодействие

## Углубленный динамический анализ

При распаковке файла из архива, он сразу был помечен, как вредоносный и удален системой, поэтому пришлось отключить Windows Defender (рисунок 19).

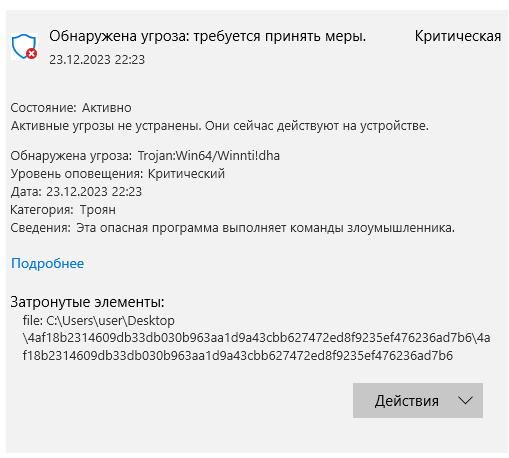


Рисунок ― Обнаружение вредоносного функционала файла

IDA Pro сразу определила файл, как dll и открыла его в соответствующем режиме. На рисунке 20 представлен декомпилированный псевдокод функции DllMain. Обратимся к документации Microsoft, чтобы разобраться в коде WinAPI.



Рисунок ― Псевдокод функции DllMain

hInstance — это дескриптор экземпляра или дескриптор модуля. Операционная система использует это значение для идентификации исполняемого файла или EXE-файла при загрузке в память. Некоторым функциям Windows требуется дескриптор экземпляра, например, для загрузки значков или растровых изображений. Соответственно, в данном случае переменная hinstDLL содержит в себе ссылку на некоторую библиотеку (возможно на исходную). hModule — это глобальная переменная.

Если выполняется некоторое условие fwdReason, то вызывается функция DisableThreadLibraryCalls. Она отключает уведомления DLL\_THREAD\_ATTACH и DLL\_THREAD\_DETACH для указанной библиотеки динамических ссылок (DLL). Это может уменьшить размер рабочего набора для некоторых приложений.

Далее проверяется значение некоторой глобальной переменной и вызываются две функции.

### sub\_180001240

На рисунке 21 представлен код первой функции (sub\_180001240).

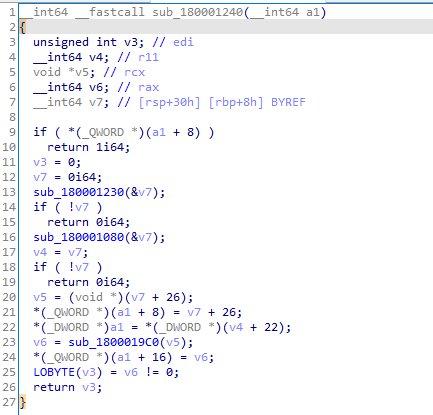


Рисунок ― Псевдокод sub\_180001240

Переменной v7 присваивается значение 0, затем она передается в функцию sub\_180001230, где ей присваивается значение некоторой глобальной переменной aTunnel (рисунок 22). Далее вызывается функция sub\_180001080 (рисунок 23).

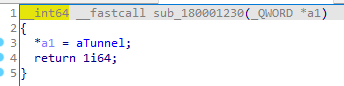
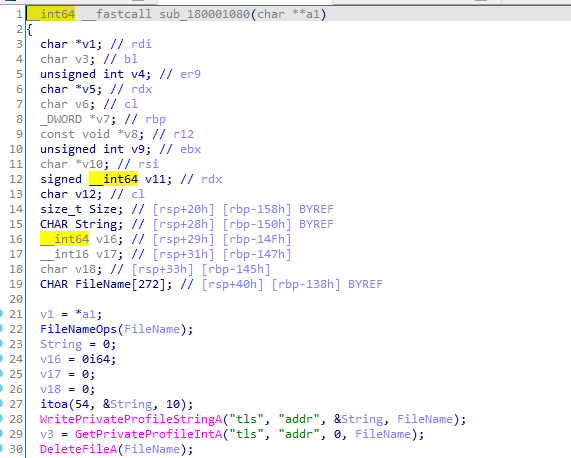


Рисунок — Псевдокод sub\_180001230



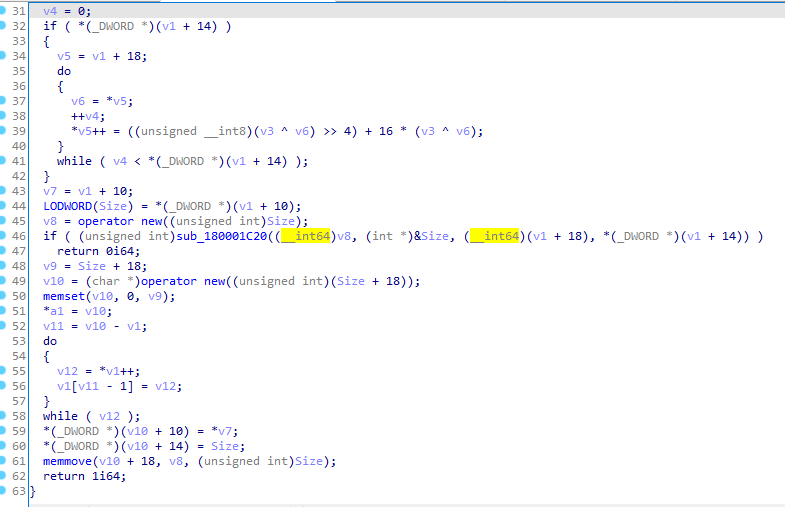


Рисунок — Псевдокод sub\_180001080

Функция sub\_180001080 содержит в себе уже куда больше интересного функционала. Во-первых, в ней сразу вызывается функция, для удобства названная FileOps. В ней вызываются такие функции WinAPI как GetTempPath и GetTempFileNameA. Первая извлекает путь к каталогу, предназначенному для временных файлов, а вторая создает имя для временного файла. Если сгенерировано уникальное имя файла, создается пустой файл и освобождается его дескриптор; в противном случае генерируется только имя файла. Таким образом в результате выполнения этой функции возвращается путь к уникальному временному файлу и создается сам файл.

Далее вызываются функции для работы с реестром. WritePrivateProfileString копирует строку в указанный раздел файла инициализации. GetPrivateProfileInt извлекает целое число, связанное с ключом, из указанного раздела файла инициализации. После чего файл удаляется. Как видим, указан раздел tls, в нем хранятся сертификаты.

Далее вызывается функция sub\_180001С20, в которой вызываются функции, проводящие некоторые вычисления над локальными и глобальными переменными, их суть сложно определить по псевдокоду. И также внутри sub\_180001С20 вызывается очень большая функция, наполненная различными switch-case, if-else и goto конструкциями. По строкам, содержащимся в ней, удалось сделать предположение, что там происходят какие-то проверки на наличие ошибок (рисунок 24).

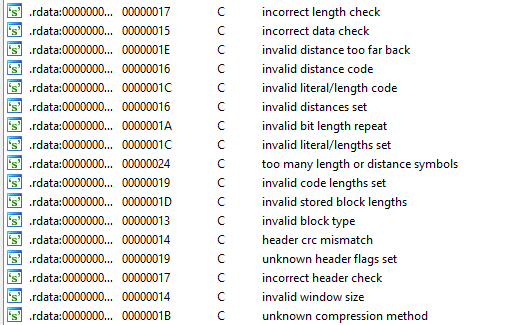
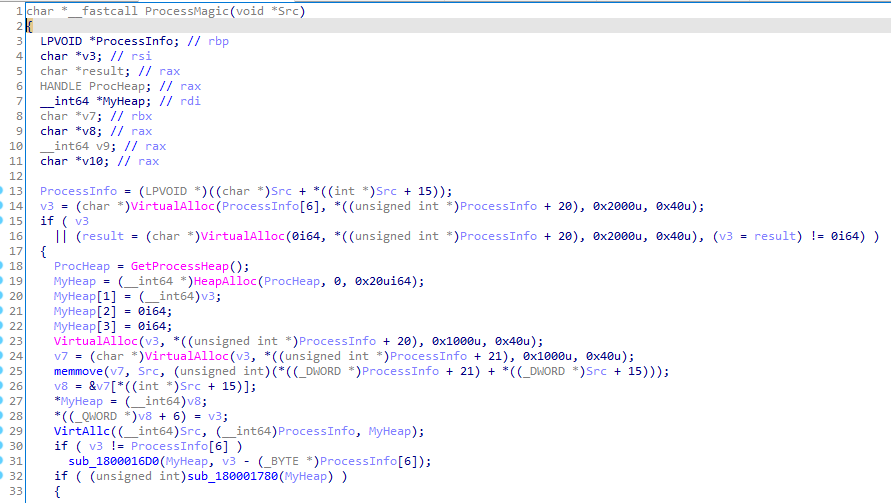


Рисунок — Строки, сообщающие об ошибках

Вернемся к функции sub\_180001240 (Рисунок 21). В самом конце вызывается функция sub\_1800019С0 (рисунок 25), которая проводит некоторые махинации с процессами. VirtualAlloc резервирует, фиксирует или изменяет состояние области страниц в виртуальном адресном пространстве вызывающего процесса. Память, выделяемая этой функцией, автоматически инициализируется равной нулю. GetProcessHeap извлекает дескриптор из кучи вызывающего процесса по умолчанию. Затем этот дескриптор можно использовать в последующих вызовах функций кучи. И наконец HeapAlloc выделяет блок памяти из кучи.

Далее вызывается функция sub\_180001780, внутри которой происходят следующие действия. IsBadReadPtr проверяет, что вызывающий процесс имеет доступ на чтение к указанному диапазону памяти. LoadLibraryA загружает указанный модуль в адресное пространство вызывающего процесса. GetProcAddress извлекает адрес экспортированной функции или переменной из указанной библиотеки динамических ссылок (DLL).



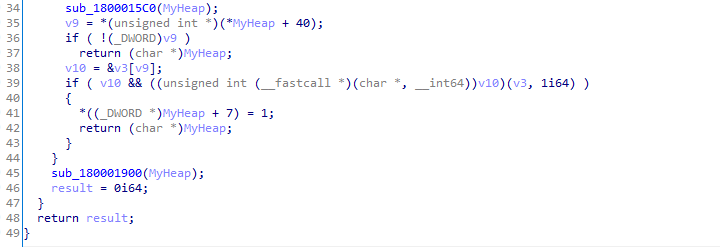


Рисунок — Псевдокод sub\_1800019С0

Далее вызывается функция sub\_1800015C0, в которой метод VirtualProtect изменяет защиту области зафиксированных страниц в виртуальном адресном пространстве вызывающего процесса. В самом конце вызывается функция sub\_180001900, в которой вызываются функции FreeLibrary, VirtualFree, GetProcessHeap и HeapFree.

### sub\_1800012D0

Вторая функция в DllMain это sub\_1800012D0. Её псевдокод представлен на рисунке 26. В ней в самом начале функция GetModuleFileNameA извлекает полный путь к файлу, содержащему указанный модуль. Модуль, должно быть, был загружен текущим процессом. Далее вызывается функция sub\_180001B40, в которой судя по всему происходят операции со строками, в том числе с аргументом ExportFunc. Затем снова вызывается GetModuleFileNameA: в этом случае параметр hModule имеет значение NULL, следовательно, функция извлекает путь к исполняемому файлу текущего процесса. Полученная строка переводится в нижний регистр и далее в ней ищется подстрока sysprep.exe.



Рисунок — Псевдокод sub\_1800012D0

Утилита Sysprep представляет собой встроенный в Windows инструмент с узкой областью применения. Используется он, в основном, для удаления из системы данных о привязке к аппаратным компонентам при создании эталонных образов, предназначенных для развертывания на ПК в корпоративных средах. На пользовательском уровне инструмент бывает полезен при замене комплектующих, когда нужно удалить драйвера старых устройств, а также при переносе настроенной операционной системы на другой компьютер с отличными характеристиками компонентов.

### CRT\_INIT

При дальнейшем анализе было выявлено, что львиная доля функционала вредоноса находится в функции CRT\_INIT. Посмотрим, откуда она вызывается. На рисунке 27 представлен псевдокод точки входа в dll DllEntryPoint. Оттуда вызывается функция \_DllMainCRTStartup, её псведокод представлен на рисунке 28. Наконец на рисунке 29 рассмотрим псевдокод CRT\_INIT, которая вызывается в \_DllMainCRTStartup перед DllMain.

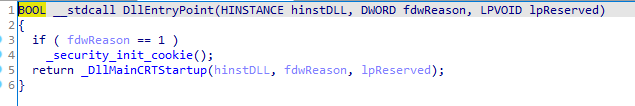


Рисунок — Псевдокод DllEntryPoint

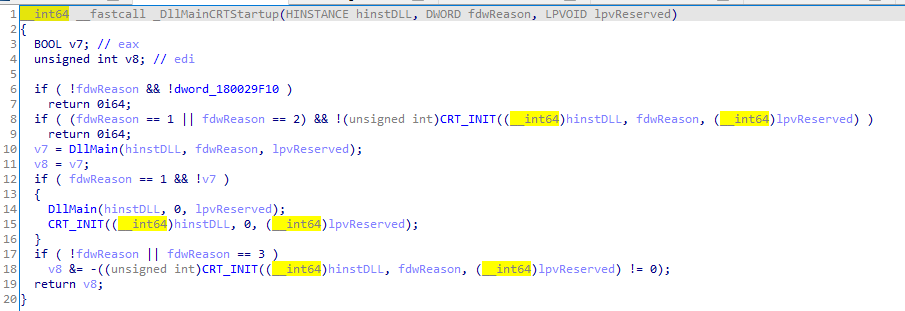


Рисунок — Псевдокод \_DllMainCRTStartup

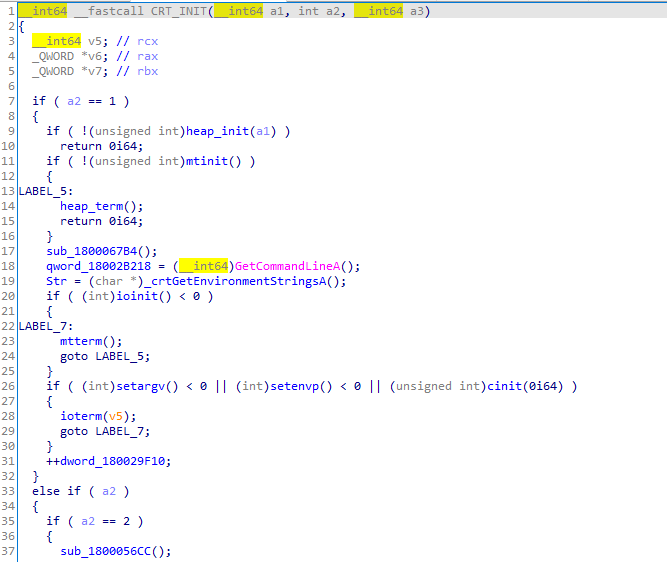
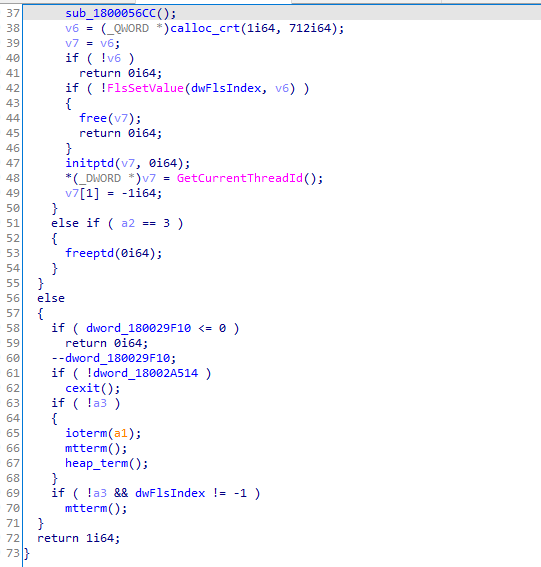
 

Рисунок — Псевдокод CRT\_INIT

Сначала вызывается функция heap\_init, в которой создается куча для процесса (HeapCreate) и через HeapSetInformation куча относится к классу HeapCompatibilityInformation.

Далее вызывается mtinit, в ней происходит init\_pointers: инициализация некоторых переменных и вызовы EncodePointer (закодированные указатели можно использовать для обеспечения другого уровня защиты значений указателей). Затем вызывается mtinitlocks, в которой происходит инициализация критической секции. После происходит FlsAlloc: выделение индекса локального хранилища fiber (FLS). Любой fiber в процессе может впоследствии использовать этот индекс для хранения и извлечения значений, которые являются локальными для fiber. Затем FlsSetValue сохраняет значение в слоте fiber local storage (FLS) вызывающего fiber для указанного индекса FLS. Потом вызывается функция initptd, в которой вызываются функции lock и unlock, смысл их действий понятен из названия — происходит работа с критической секцией. После вызывается функция GetCurrentThreadId, получающая идентификатор текущего потока. В самом конце вызывается mtterm, в которой через FlsFree освобождается ранее выделенный индекс и затем удаляются замки (критическая секция).

Также в процессе анализа функций инициализации была обнаружена функция обработки ошибок и исключений. Особый интерес в ней представляет проверка на наличие отладчика IsDebuggerPresent и завершение работы с исключением при его обнаружении. Декомпилированный код функции представлен на рисунке 30.

После mtinit в CRT\_INIT вызывается функция GetCommandLineA, которая извлекает строку командной строки для текущего процесса.

Затем вызывается функция \_crtGetEnvironmentStringsA. Внутри нее происходит вызов GetEnvironmentStringsW — извлечение переменных среды для текущего процесса. Далее происходит обработка извлеченных данных, в конце указатель освобождается.

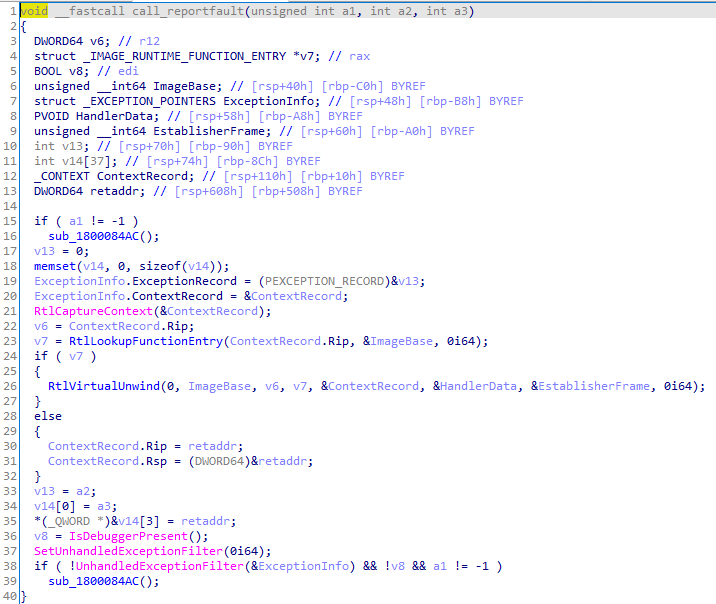


Рисунок — Функция с проверкой на наличие отладчика

Затем вызывается функция ioinit, внутри которой функция GetStartupInfoW извлекает содержимое структуры STARTUPINFO, которая была указана при создании вызывающего процесса. Происходит обработка данных структуры. Затем вызывается GetStdHandle извлекает дескриптор для указанного стандартного устройства (стандартный ввод, стандартный вывод или стандартная ошибка). Далее этот дескриптор ещё как-то обрабатывается. В самом конце вызывается SetHandleCount, которая изменяет количество дескрипторов файлов, доступных процессу. Для Win32 на основе DOS максимальное число дескрипторов файлов, доступных процессу по умолчанию, равно 20. Было установлено число не менее 32.

Далее рассмотрим конструкцию на строке 26:

if ( (int)setargv() < 0 || (int)setenvp() < 0 || cinit(0, v5) ).

Рассмотрим, что происходит в каждой из функций. В setargv происходит вызов двух основных функций. В первой функции происходит много всего, можно выделить вызов функции WinAPI GetACP, которая извлекает текущий идентификатор кодовой страницы ANSI Windows для операционной системы. Вторая функция называется parse\_cmdline и судя по всему её функционал соответствует названию.

Далее рассмотрим setenvp. Её функционал точно определить сложно, но она тоже вызывает функцию, которая влечет собой вызов GetACP, а также там есть обработка ошибок и последующий выход.

В cinit вызывается IsNonwritableInCurrentImage, из которой вызываются ValidateImageBase и FindPESection. Похоже, что происходит проверка на возможность записать что-то в указанную область памяти и поиск PE секции процесса.

Вернемся к анализу CRT\_INIT. Из интересного далее вызываются функции FlsSetValue и GetCurrentThreadId. Далее обрабатываются некоторые глобальные переменные и вызываются функции, которые освобождают выделенные ресурсы.

Собирая воедино результаты анализа, автор делает вывод о функционале CRT\_INIT. В основном здесь происходит некоторый подготовительный этап перед вызовом DllMain. Смысл подготовки заключается в выделении ресурсов: куч, переменных среды, проверок на возможность записать данные, а также проверка на наличие отладчика.

### Обфускация кода

Также были обнаружены некоторые средства обфускации. Например, после анализа осталось ещё около половины непроанализированных функций. Как оказалось, они нигде не вызываются, то есть находятся в секции данных исполняемого файла (рисунок 31) и непонятно, используются ли они где-то.

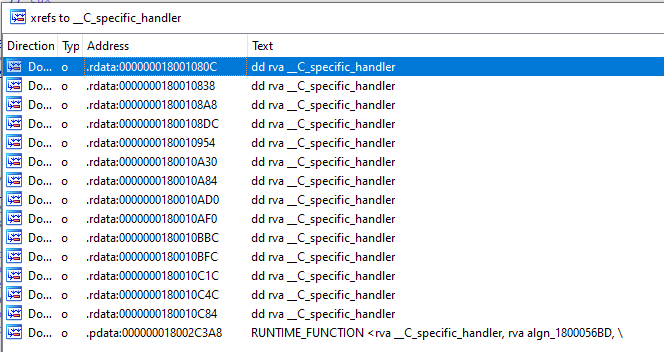


Рисунок — Функция находится в секции данных

Также были найдены функции-пустышки (рисунок 32). Некоторые функции просто возвращают 0 или 1. Кроме того были функции, которые просто вызывают другие функции и больше ничего не делают. Уровень обфускации можно оценить как средний. При этом исполняемый файл не был никак упакован или зашифрован.

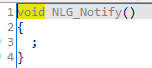


Рисунок — Функция-пустышка

## Классификация по MITRE ATT&CK

Смысл вредоноса заключается в модификации памяти вызываемого процесса, в инъекции в легитимное ПО, манипуляций с реестром. Возможно используется утилита sysprep.exe, которая служит для удаления драйверов, изменения аппаратной конфигурации в ОС без непосредственной замены железа, что влечет за собой сбои в работе компьютера. Видимо с помощью этой утилиты и была произведена подмена драйвера spliter.sys. Закрепление в системе происходит с помощью записи некоторых данных в logon процесс.

При этом обнаруженных при анализе в песочнице вызовов spliter.sys, subject.dll, rundll32.exe обнаружено не было. Возможно эта часть кода утеряна, затерта антивирусом, зашифрована, обфусцирована или просто-напросто нераспознана идой. Также не было обнаружено сетевого взаимодействия: ни найденного в песочнице домена, ни апишных функций, которые бы реализовывали данное взаимодействие.

По описанию более ранних троянов группировки [2] возможно, что система сама триггерит вирус неким событием. В таблице 1 представлено описание действий вредоносного файла по методике MITRE ATT&CK. Также на рисунке 33 есть описание тактик, предоставленное песочницей.

Таблица 1 — Вредоносные действия, техники MITRE ATT&CK

| Тактика MITRE ATT&CK | Описание |
| --- | --- |
| Закрепление в системе | Заражение существующих файлов— перезапись файлов, установка своих компонентов — создание временных файлов |
| Сбор информации о системе | Версия ОС, пути к файлам, информация о памяти процессов, значения ключей реестра |
| Поиск средств защиты в системе | Анализ процессов, файлов/директорий, реестра |
| Реализация вредоносных действий | Вызов sysprep.exe, подмена или модификация драйверов. |

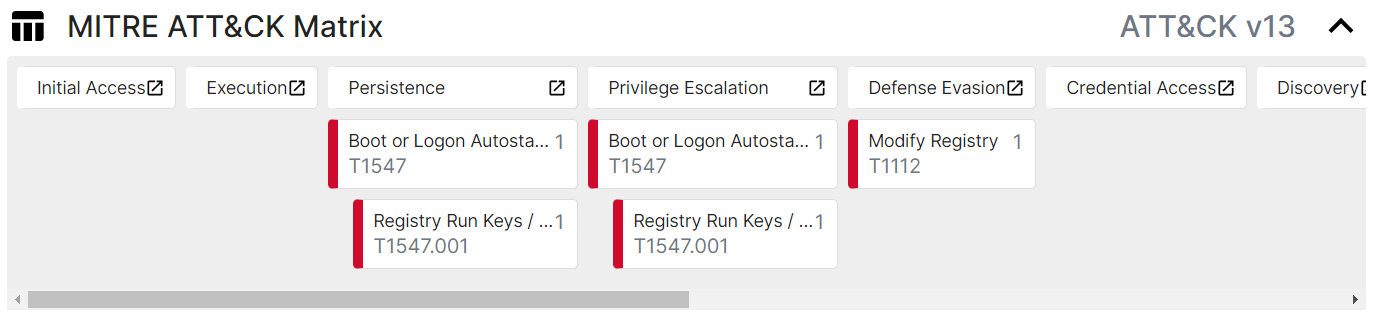


Рисунок 33 — Вредоносные действия, техники MITRE ATT&CK

# Вывод

В ходе выполнения данной лабораторной работы были изучены сценарии известных целенаправленных атак, рассмотрены способы их обнаружения и предотвращения. Был изучен пример вредоносного программного обеспечения (ВПО), используемого в целенаправленных атаках. В качестве варианта индивидуального задания была зучена группировка Winnti, которая специализируется на целенаправленных атаках на производителей компьютерных игр. Ещё в начале 10-ых Лаборатория Касперского опубликовала свое исследование этой группировки. Тип используемого ВПО можно охарактеризовать как троян и бэкдор, поскольку зачастую злоумышленник получает удаленный доступ к системе жертвы.

По вредоносному функционалу данный сэмпл производит инъекцию в кучу вызвавшего процесса, манипулирует данными реестра, памятью, создает временные файлы. Также ВПО вызывает утилиту sysprep.exe, которая позволяет удалять драйвера и управлять хардварными компонентами. В частности, происходит модификация файла драйвера spliter.sys и перезапись данных реестра об этом драйвере.

В файле присутствуют средства анти-отладки. После анализа ещё некоторых целенаправленных атак группировки winnti [3], [4] было выявлено, что обычно они используют набор вредоносного ПО, а также триггерят легитимное ПО для повышения привилегий и сбора информации, либо подменяют легитимное ПО (заражают его), чтобы нанести ущерб компьютеру жертвы.

# Список источников

1. <https://github.com/cyber-research/APTMalware/tree/master/samples/Winnti>
2. <https://securelist.ru/winnti-e-to-vam-ne-igrushki/201/>
3. <https://habr.com/ru/companies/lanit/articles/570516/>
4. <https://habr.com/ru/companies/lanit/articles/571592/>