Теоретическая часть

Вредоносное ПО (вирусы, трояны, вымогатели) — это программы, созданные для нанесения вреда компьютерам, сети или пользователям.

Их основные типы:

- Вирусы — внедряются в другие программы или файлы и размножаются при их запуске, нарушая работу системы.

- Трояны — маскируются под полезные программы, но дают злоумышленникам удалённый доступ или выполняют вредоносные действия.

- Вымогатели — шифруют данные жертвы и требуют выкуп за их разблокировку.

Принцип работы:

Вирусы внедряются в файлы и запускаются вместе с ними, трояны обманывают пользователя и запускаются как легитимные программы, вымогатели после проникновения блокируют данные и требуют выкуп.

Принципы анализа вредоносного ПО:

- Статический анализ — изучение кода, структуры и инструкций программы без её запуска.

Позволяет понять, что делает программа, выявить подозрительные участки, но не показывает поведение в реальном времени.

- Динамический анализ — запуск программы в изолированной среде (песочнице) для наблюдения её поведения: обращения к файлам, сети, изменения реестра и т.д.

Помогает выявить реальные действия вредоносного ПО.

Основные правила безопасности при исследовании ВПО:

- Использование изолированной среды (виртуальная машина или песочница), чтобы избежать заражения основной системы.

- Обеспечение сетевой изоляции или контролируемого сетевого трафика.

- Регулярное создание снимков виртуальной машины для быстрого отката.

- Не запускать подозрительное ПО на реальной операционной системе.

- Использовать антивирусные и мониторинговые инструменты для контроля действий ПО.

Практическая часть

1)Установка Windows 10 на виртуальную машину

1. **Скачай образ Windows 10 (ISO):**

С официального сайта Microsoft (может потребоваться лицензионный ключ).

Или с других надежных источников.

2. **Установи VirtualBox (или VMware):**

3. **Создай виртуальную машину (ВМ) в VirtualBox:**

Открой VirtualBox.

Нажми "Создать" ("New").

Введи имя ВМ, выбери тип "Microsoft Windows" и версию "Windows 10 (64-bit)".

Выдели достаточно памяти (минимум 2 ГБ, рекомендуется 4 ГБ или больше).

Создай виртуальный жесткий диск (VDI, динамический размер).

Выдели минимум 50 ГБ

4. **Установи Windows 10 на ВМ:**

Выбери созданную ВМ и нажми "Настроить" ("Settings").

Перейди в раздел "Носители" ("Storage").

В разделе "Контроллер: IDE" добавь ISO-образ Windows 10.

Запусти ВМ.

Следуй инструкциям установщика Windows 10.

5. **Установи Guest Additions:**

После установки Windows 10, в меню ВМ выбери "Устройства" -> "Подключить образ диска Дополнений гостевой ОС" ("Devices" -> "Insert Guest Additions CD image").

Запусти установку с виртуального диска.

Перезагрузи ВМ.

2. Скачивание и установка программ 1. **FakeNet-NG:**

Скачай с GitHub

Это Python-скрипт, поэтому установи Python 3.x (если ещё не установлен) и pip.

Открой командную строку (CMD) от имени администратора.

Перейди в каталог FakeNet-NG.

Выполни pip install -r requirements.txt для установки зависимостей.

Запуск: python FakeNet.py

2. **Process Monitor:**

Скачай с сайта Microsoft: <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>

Распакуй архив.

Запусти Procmon.exe от имени администратора.

3. **Ghidra:**

Скачай с сайта NSA: <https://ghidra-sre.org/>

Распакуй архив.

Запусти ghidraRun.bat (или ghidraRun для Linux/macOS).

Установи Java Development Kit (JDK), если потребуется.

3. Отключение сети в ВМ

1. **Открой VirtualBox Manager.**

2. **Выбери свою виртуальную машину с Windows 10.**

3. **Нажми "Настроить" ("Settings").**

4. **Перейди в раздел "Сеть" ("Network").**

5. **В выпадающем списке "Присоединить" ("Attached to") выбери "Not Attached" ("Не подключено").**

6. **Нажми "OK".** Теперь ВМ не имеет сетевого подключения.

Дополнительные замечания

**Снимки (Snapshots):** Перед установкой ПО сделай снимок ВМ в VirtualBox ("Машина" -> "Сделать снимок") ("Machine" -> "Take Snapshot").

Это позволит быстро вернуться к исходному состоянию в случае проблем.

**Безопасность:** Учитывай, что анализ вредоносного ПО требует осторожности.

Важно, чтобы ВМ была изолирована от основной системы.

2) 1. Статический анализ в Ghidra

1. **Открытие файла в Ghidra:** \* Запусти Ghidra. \* Создай новый проект ("File" -> "New Project").

Импортируй файл eicar.com ("File" -> "Import File").

Ghidra может предложить анализ файла.

Прими предложение ("Yes").

Убедитесь, что выбран правильный формат файла (COM).

После завершения анализа открой CodeBrowser для анализа кода ("Open CodeBrowser").

2. **Поиск строки** EICAR-STANDARD-...**:**

В CodeBrowser перейди в "Window" -> "Defined Strings".

В списке строк найди строку, начинающуюся с "EICAR-STANDARD-".

Кликни дважды на этой строке.

Ghidra покажет, где эта строка используется в коде.

3. **Определение действий, выполняемых программой:**

Посмотри, где находится код, использующий эту строку.

Обычно, это код, который выводит строку на экран.

Проанализируй код вокруг места, где используется строка EICAR-STANDARD-.... COM-файлы обычно содержат короткий код.

В COM-файлах обычно используются прерывания DOS (int 21h) для вывода текста на экран.

Посмотри, есть ли вызовы таких прерываний.

Определи, как строка передаётся в функцию вывода.

Обрати внимание на вызовы других функций и переходы.

4. **Подтверждение результата:**

Действительно, файл EICAR, как правило, выводит текстовое сообщение на экран и завершается.

2. Динамический анализ с FakeNet-NG

1. **Запуск FakeNet-NG:**

Открой командную строку от имени администратора.

Перейди в каталог, где находится FakeNet.py.

Запусти FakeNet-NG: python FakeNet.py -i <IP-адрес виртуальной машины> Укажите IP-адрес виртуальной машины, чтобы FakeNet перехватывал трафик только от этой машины.

Если IP не указать, FakeNet будет перехватывать весь трафик, что может быть нежелательно.

Убедись, что FakeNet-NG запустился без ошибок и слушает порты.

2. **Запуск** eicar.com**:**

Внутри виртуальной машины, открой командную строку (CMD).

Перейди в каталог, где находится eicar.com.

Запусти eicar.com.

Программа должна вывести текстовое сообщение.

3. **Проверка логов FakeNet:**

В окне командной строки, где запущен FakeNet-NG, посмотри, появились ли новые записи в логах.

Файл eicar.com обычно не делает сетевых запросов, поэтому в логах FakeNet-NG, скорее всего, не будет новых записей.

4. **Анализ результатов:**

Если FakeNet не зафиксировал никаких сетевых запросов, значит eicar.com не использует сеть.

**Почему нет сетевых запросов?**

Файл eicar.com — это простой тестовый файл для проверки работы антивирусов.

Он разработан таким образом, чтобы не представлять реальной угрозы и не выполнять сетевые операции.

Он просто выводит строку на экран и завершает свою работу.

3)1. Запуск Process Monitor с фильтрами

1. **Запуск Process Monitor:**

Запусти Procmon.exe от имени администратора.

Process Monitor начнёт захватывать все события в системе.

2. **Настройка фильтров:**

Нажми Ctrl+L или выбери "Filter" -> "Filter..." в меню.

Добавь следующие фильтры:

Process Name is eicar.com Then Include \* Operation is CreateFile Then Include \* Operation is RegSetValue Then Include

Убедись, что остальные операции не включены (исключены).

Нажми "Apply" и "OK".

Теперь Process Monitor будет показывать только события, связанные с процессом eicar.com и операциями CreateFile и RegSetValue.

2. Запуск eicar.com и наблюдение

1. **Запуск** eicar.com**:**

Открой командную строку (CMD).

Перейди в каталог, где находится eicar.com.

Запусти eicar.com.

Программа должна вывести текстовое сообщение.

2. **Наблюдение в Process Monitor:**

В окне Process Monitor посмотри, появились ли новые события.

Обрати внимание на следующие моменты:

Создает ли eicar.com какие-либо файлы (операция CreateFile)?

Изменяет ли eicar.com какие-либо записи в реестре (операция RegSetValue)?

Запускает ли eicar.com какие-либо процессы (операция CreateProcess, которую мы не фильтруем, но можно посмотреть)?

3. Анализ поведения Обычно файл eicar.com демонстрирует следующие действия:

**Создание файлов:** Как правило, eicar.com не создает никаких файлов.

Это всего лишь тестовый файл, который выводит текстовое сообщение.

**Изменение реестра:** eicar.com не изменяет реестр.

**Запуск процессов:** eicar.com не запускает никаких других процессов.

Если ты видишь другие действия (создание файлов, изменение реестра, запуск процессов), это может быть признаком того, что файл eicar.com модифицирован или что-то не так с твоей системой.

В этом случае, необходимо провести более тщательный анализ.

В типичном случае, ты не увидишь событий CreateFile или RegSetValue, потому что eicar.com не выполняет эти действия.

Давай подытожим:

1. Запустил Process Monitor с нужными фильтрами.

2. Запустил eicar.com.

3. Увидел, что eicar.com не создает файлы и не меняет реестр.

Ответы на вопросы

**1. Какие признаки вредоносного поведения вы обнаружили?**

В примере с game.exe:

Чтение содержимого файла secret.txt.

Попытка отправки данных на удалённый сервер example.com.

Эти признаки указывают на потенциальную кражу данных или несанкционированную передачу информации.

**2. Почему EICAR безопасен для анализа?**

EICAR (EICAR-STANDARD-ANTIVIRUS-TEST-FILE) - это безопасный тестовый файл, предназначенный для проверки работы антивирусных программ.

Он содержит известную сигнатуру, которая позволяет антивирусам его обнаружить, но не выполняет вредоносных действий.

Его задача - убедиться, что антивирус работает правильно.

**3. Как FakeNet помогает анализировать сетевую активность?**

FakeNet-NG перехватывает сетевые запросы, сделанные программой, и перенаправляет их на локальные службы, имитируя реальные сетевые соединения.

Это позволяет:

Увидеть, какие домены или IP-адреса пытается подключиться программа.

Проанализировать данные, которые программа отправляет и получает.

Изолировать программу от реальной сети и предотвратить несанкционированную передачу данных.

**4. Что такое вредоносное программное обеспечение (ВПО) и какие его основные типы?**

ВПО - это программное обеспечение, предназначенное для нанесения вреда компьютерным системам, сетям или данным. Основные типы:

**Вирусы:**

Заражают файлы и распространяются при их запуске.

**Черви:**

Распространяются по сети, не требуя участия пользователя.

**Трояны:**

Маскируются под полезные программы, но выполняют вредоносные действия.

**Программы-вымогатели (Ransomware):** Шифруют данные и требуют выкуп за их восстановление.

**Шпионское ПО (Spyware):**

Собирает информацию о пользователе и отправляет её злоумышленникам.

**Рекламное ПО (Adware):**

Отображает нежелательную рекламу.

**Бэкдоры (Backdoors):**

Предоставляют злоумышленникам удалённый доступ к системе.

**Руткиты (Rootkits):**

Скрывают присутствие вредоносного ПО в системе.

**5. Каковы основные цели анализа вредоносного ПО?**

Определение функциональности и поведения вредоносного ПО.

Выявление методов распространения.

Поиск уязвимостей, которые использует вредоносное ПО.

Разработка методов обнаружения и предотвращения атак.

Сбор информации для расследования киберпреступлений.

**6. В чем заключается разница между статическим и динамическим анализом вредоносного кода?**

**Статический анализ:**

Изучение кода без его выполнения. Анализируется структура, строки, функции.

**Динамический анализ:**

Анализ поведения вредоносного ПО во время его выполнения.

Наблюдаются действия с файлами, реестром, сетью.

**7. Какие инструменты наиболее распространены для статического анализа вредоносных программ и какие их основные функции?**

**Дизассемблеры (IDA Pro, Ghidra, radare2):** Преобразуют машинный код в ассемблерный для анализа.

**Декомпиляторы (Ghidra, IDA Pro):**

Пытаются восстановить исходный код из машинного кода.

**Анализаторы строк (strings):**

Извлекают текстовые строки из файла.

**Hex-редакторы:**

Позволяют просматривать и редактировать бинарные файлы.

**PE-анализаторы (PEview):**

Анализируют структуру PE-файлов (Windows).

**8. Что такое обфускация кода и как она используется во вредоносных программах?**

Обфускация - это намеренное усложнение кода для затруднения его анализа и понимания.

Используется для скрытия вредоносной функциональности, защиты от обнаружения и обратной разработки.

9. Каковы возможные признаки наличия вредоносного кода в программном обеспечении?

Необычная активность сети.

Изменения в файловой системе или реестре без явного действия пользователя.

Высокая загрузка процессора или использование памяти без видимой причины.

Появление новых файлов или программ, которые не устанавливались.

Сообщения об ошибках или сбои в работе системы.

Замедление работы системы.

**10. Что такое песочница (sandbox), и как она помогает в анализе вредоносного ПО?**

Песочница - это изолированная среда, в которой можно безопасно запускать и анализировать вредоносное ПО.

Все изменения, сделанные ВПО в песочнице, не влияют на основную систему.

**11. Каковы типичные методы, используемые для обхода защитных механизмов антивирусов?**

Обфускация кода.

Шифрование.

Использование полиморфизма и метаморфизма (изменение кода при каждом заражении).

Эксплойты для обхода систем безопасности.

Использование социальной инженерии.

Упаковка кода (использование упаковщиков для сжатия и сокрытия исполняемого кода).

**12. Что такое "инъекция кода" и как она применяется во вредоносных программах?**

Инъекция кода - это внедрение вредоносного кода в легитимный процесс.

Используется для скрытия вредоносной активности и обхода защитных механизмов.

**13. Каковы основные этапы процесса динамического анализа?**

**Подготовка:**

Настройка изолированной среды (песочницы).

2. **Запуск:**

Запуск вредоносного ПО в песочнице.

3. **Мониторинг:** Наблюдение за поведением ВПО (изменения в файлах, реестре, сетевые подключения).

4. **Анализ:** Анализ собранных данных для определения функциональности и целей ВПО.

5. **Отчётность:** Составление отчёта о результатах анализа.

**14. Какие данные можно извлечь при помощи статического анализа программного кода?**

Строки и константы.

Импортированные функции и библиотеки.

Структура кода (функции, циклы, условия).

Криптографические алгоритмы.

URL-адреса и IP-адреса.

Метаданные файла.

**15. Как изучение поведения вредоносных программ помогает в разработке методов защиты и предотвращения атак?**

Анализ ВПО позволяет понять:

Какие уязвимости используются.

Как распространяется ВПО.

Какие цели преследуют злоумышленники.

На основе этих знаний разрабатываются сигнатуры для антивирусов, методы обнаружения вторжений и рекомендации по безопасности.

**16. Каковы наиболее распространенные сценарии использования вредоносного ПО?**

Кража данных (логины, пароли, финансовая информация). • Шпионаж.

Вандализм (удаление или повреждение данных).

Вымогательство (ransomware).

Создание ботнетов для DDoS-атак или рассылки спама.

Финансовые махинации.

Кибершпионаж (кража секретов).

**17. Что такое обратная разработка (reverse engineering) и как она применяется в исследовании вредоносных программ?**

Обратная разработка - это процесс анализа программного обеспечения для понимания его структуры и функциональности без доступа к исходному коду.

В анализе ВПО она используется для:

Восстановления алгоритмов работы ВПО.

Поиска уязвимостей.

Обхода обфускации и шифрования.

Разработки методов обнаружения и предотвращения.

**18. Каковы основные риски, связанные с анализом вредоносного ПО в реальной среде?**

Заражение системы.

Потеря данных.

Нанесение ущерба другим системам в сети.

Раскрытие конфиденциальной информации.

**19. Какие меры предосторожности следует соблюдать во время анализа вредоносного кода?**

Использовать изолированную среду (песочницу).

Отключить сетевое подключение.

Сделать резервные копии данных.

Использовать антивирусное ПО.

Использовать виртуальную машину или специализированную платформу для анализа.

Проявлять осторожность и не запускать подозрительные файлы на основной системе.

20. Как анализ вредоносного ПО может помочь в расследованиях кибератак?

Определение источника атаки.

Выявление целей атаки.

Оценка ущерба.

Сбор доказательств для судебного преследования.

Разработка мер по предотвращению повторных атак.

21. Какие аспекты этики следует учитывать при анализе и исследовании вредоносного ПО?

Соблюдение законодательства и лицензионных соглашений.

Уважение конфиденциальности и неприкосновенности частной жизни.

Использование полученных знаний только в законных целях.

Не нанесение вреда другим системам или пользователям.

Ответственное раскрытие информации об уязвимостях.