САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики-процессов управления

Программа бакалавриата
“Большие данные и распределенная цифровая платформа”

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Программирование в Linux»
на тему
«Разработка детектора подозрительного трафика»

Студент гр. 23Б15-пу
Трофимов И.А.

Преподаватель
Киямов Ж. У.

Санкт-Петербург
2024 г.

Оглавление

1. Цель работы 3
2. Описание задачи 3
3. Теоретическая часть 5
4. Основные шаги программы 9
5. Описание программы 9
6. Рекомендации пользователя 13
7. Рекомендации программиста 13
8. Исходный код программы 13
9. Контрольный пример 14
10. Вывод 16
11. Источники 16

Цель работы

Разработать механизм обнаружения и блокировки подозрительного сетевого трафика на основе анализа пакетов с использованием библиотеки Scapy, а также исследовать методы защиты сетевой инфраструктуры от потенциальных угроз, таких как сканирование портов, атаки DDoS и другие виды аномального сетевого трафика.

Описание задачи

Задача состоит в разработке механизма для обнаружения и блокировки подозрительного сетевого трафика с использованием Python и библиотеки Scapy. Этот механизм должен анализировать входящий сетевой трафик, выявлять потенциальные угрозы и автоматически принимать меры для их блокировки. Применяя различные методы анализа пакетов, можно будет обнаружить такие угрозы, как сканирование портов, DoS-атаки, аномально большие пакеты и другие типы подозрительного поведения в сети.

Этапы реализации:

1. **Установка необходимых библиотек:**

    - Установить библиотеку Scapy, которая предоставляет удобные инструменты для анализа и манипуляции сетевыми пакетами. Это позволит слушать и обрабатывать сетевой трафик на Python.
2. **Создание скрипта для анализа сетевого трафика:**

    - Написать Python-скрипт с использованием Scapy, который будет перехватывать пакеты из сети в реальном времени.

- Проанализировать заголовки пакетов, такие как IP-адреса, порты, протоколы и другие параметры, для дальнейшего применения правил обнаружения подозрительного трафика.

3. **Определение правил и логики для выявления подозрительного трафика:**

   - Разработать логику, основанную на различных типах аномального поведения в сети. Например:
     - Аномально большие или малые пакеты.
     - Чрезмерное количество запросов с одного IP-адреса за короткий промежуток времени (атакующие сканируют порты).
     - Повторяющиеся запросы на один и тот же ресурс.

4. **Блокировка подозрительного трафика:**

   - После обнаружения подозрительных пакетов, скрипт должен предпринять меры для блокировки этих пакетов. Это может быть реализовано с помощью отправки ICMP сообщений "Destination Unreachable" или блокировки IP-адресов/портов с помощью средств сетевого экрана (например, iptables или аналогичных инструментов).

5. **Тестирование и настройка:**

   - Провести тестирование на различных типах сетевого трафика (например, нормальный трафик, сканирование портов, атаки на отказ в обслуживании).

- Настроить параметры работы скрипта (например, пороговые значения для анализа пакетов, количество попыток в единицу времени) для повышения эффективности работы механизма.

6. **Непрерывное обновление и мониторинг:**

   - Поддерживать механизм в актуальном состоянии, обновляя правила для обнаружения новых угроз и сценариев атак.
   - Регулярно анализировать логи работы системы для выявления новых аномалий и улучшения защиты.

Эти этапы позволяют не только создать инструмент для защиты от текущих угроз, но и обеспечить возможность его адаптации к новым типам атак, что делает систему более устойчивой и гибкой.

Теоретическая часть

Обнаружение и блокировка подозрительного сетевого трафика — важная задача в области информационной безопасности. Сетевой трафик может содержать различные виды атак, такие как сканирование портов, DoS-атаки, брутфорс, а также передача вредоносных данных. Для защиты сети от таких угроз необходимо внедрять системы, которые способны автоматически анализировать трафик, выявлять аномалии и предпринимать меры по их блокировке. Одним из инструментов для выполнения этой задачи является библиотека Scapy, которая предоставляет широкие возможности для анализа, генерации и манипуляции сетевыми пакетами.

Сетевой трафик состоит из множества пакетов данных, передаваемых по сети, каждый из которых содержит информацию о источнике, получателе и типе данных. Анализ этих пакетов позволяет выявить подозрительные

активности. Программные решения, которые занимаются обнаружением таких аномалий, часто используют различные алгоритмы и сигнатуры для фильтрации вредоносных пакетов.

В этом контексте можно выделить несколько ключевых аспектов:

1. **Сетевой трафик и его анализ**: Сетевой трафик можно анализировать по различным характеристикам пакетов, таким как IP-адреса, порты, протоколы и размер пакетов. Обнаружение аномальных или подозрительных характеристик помогает выявить возможные атаки.

2. **Типы атак и их особенности**:

    - **Сканирование портов**: попытка злоумышленника определить открытые порты на сервере.
    - **DoS-атаки**: атаки, направленные на перегрузку ресурсов сервера (например, отправка большого количества пакетов, чтобы исчерпать пропускную способность).
    - **Аномалии в трафике**: например, чрезмерная частота запросов с одного IP-адреса может свидетельствовать о попытке взлома.
3. **Методы обнаружения трафика**: Основные методы включают анализ сигнатур (по заранее определенным шаблонам) и аномального поведения. Первый метод эффективен, когда известна точная сигнатура атаки, в то время как второй — полезен для обнаружения новых или ранее неизвестных угроз.

4. **Меры блокировки трафика**: После обнаружения подозрительных пакетов необходимо принять меры для их блокировки. Это может быть сделано с помощью фильтрации пакетов или отправки сообщений об

ошибке (например, ICMP Destination Unreachable).

Основные шаги программы

1. **Импорт необходимых библиотек**:
    - Подключение библиотеки Scapy для работы с сетевыми пакетами и их анализом.
    - Импорт других необходимых библиотек, например, для логирования или отправки ICMP-сообщений (например, os, time).
2. **Настройка захвата сетевого трафика**:
    - Настройка прослушивания сети для захвата пакетов. Это может быть сделано с помощью метода sniff() из библиотеки Scapy, который позволяет перехватывать пакеты в реальном времени.
    - Установка фильтров для захвата только нужных пакетов (например, только TCP или UDP пакеты, или пакеты с определенными IP-адресами).
3. **Анализ захваченных пакетов**:
    - Для каждого захваченного пакета извлекаются ключевые параметры: IP-адреса, порты, протоколы, флаги, размер пакета и другие характеристики.
    - В зависимости от структуры пакета, проверяется, соответствует ли он подозрительному поведению (например, сканирование портов, аномальная частота запросов и т.д.).
4. **Определение подозрительного трафика**:

- Разработка правил для выявления подозрительного поведения. Например:
  - **Сканирование портов**: если пакеты приходят на разные порты с одного IP-адреса в короткий промежуток времени.
  - **Аномалии по размеру пакетов**: если размер пакета превышает или слишком мал по сравнению с обычными значениями.
  - **Чрезмерная частота запросов**: если с одного IP-адреса приходит слишком много пакетов за короткий период времени (например, попытки брутфорс-атаки).
- Реализация логики для классификации пакетов как подозрительных.

5. **Блокировка подозрительного трафика**:

   - После того как подозрительный пакет или источник трафика определен, необходимо заблокировать его:
     - Отправить ICMP-сообщение "Destination Unreachable" для блокировки конкретных IP-адресов или портов.
     - Использовать фильтрацию пакетов на уровне системы (например, через iptables в Linux или аналогичные инструменты) для блокировки IP-адресов или портов.
   - В случае более сложных атак можно интегрировать программу с системами предотвращения вторжений (IDS/IPS).

6. **Логирование и отчетность**:

   - Каждый подозрительный пакет или блокировка фиксируется в логах для дальнейшего анализа.

- Программа должна вести журнал, в котором указывается, какой IP-адрес или порт был заблокирован, а также информация о типе атаки или аномалии.

7. **Непрерывное обновление и мониторинг**:

   - Постоянное отслеживание состояния сети и сканирование пакетов на наличие новых угроз.
   - Обновление правил обнаружения в зависимости от выявленных новых типов атак.
   - Реализация периодических проверок для мониторинга работы системы защиты и корректировки параметров фильтрации.

8. **Тестирование и оптимизация**:

   - После реализации программы необходимо провести тестирование с различными сценариями сетевых атак (например, сканирование портов, DDoS-атаки).
   - Настройка параметров программы для повышения точности и эффективности обнаружения и блокировки подозрительного трафика.

Эти шаги образуют основу работы программы, обеспечивая надежное обнаружение и блокировку угроз в реальном времени.

Описание программы

Программа предназначена для обнаружения и блокировки подозрительного сетевого трафика в реальном времени. Она использует библиотеку Scapy для захвата и анализа сетевых пакетов, а также реализует механизмы обнаружения аномалий и сетевых атак на основе заранее заданных правил и

сигнатур. Программа анализирует такие параметры, как IP-адреса, порты, размер пакетов, а также частоту и типы запросов, чтобы выявить потенциальные угрозы, такие как сканирование портов или DDoS-атаки.

При обнаружении подозрительного трафика программа принимает меры по блокировке источников угроз, отправляя ICMP-сообщения о недостижимости или применяя фильтрацию на уровне сети для блокировки определенных IP-адресов или портов. Все действия логируются для дальнейшего анализа и мониторинга, что позволяет улучшать эффективность работы программы и адаптировать её под новые угрозы.

Программа обеспечивает эффективную защиту сети, своевременно выявляя и блокируя угрозы, что способствует повышению безопасности информационной инфраструктуры.

Таблица 1. main.py

Функция	Описание	Результат
start_sniffing	запускает процесс захвата сетевого трафика с использованием библиотеки Scapy	None

Таблица 2. blocker.py

Функция	Описание	Результат
block_ip	блокирует Ip	None
unblock_ip	разблокирует Ip	None

Таблица 4. GUI.py

Функция	Описание	Результат
add_request	добавляет запрос в список, если указана причина, то выделяет красным	None

add_blocked_ip	добавляет Ip в список заблокированных	None
block_selected_ip	блокирует выбранный Ip из списка запросов	None
unblock_selected_ip	разблокирует выбранный Ip из списка запросов	None

Таблица 5. logger.py

Функция	Описание	Результат
log_suspicious_event	создает log	None

Рекомендации пользователя

Конфигурация: Для запуска программы, сначала убедитесь, что у вас установлены все необходимые библиотеки. После этого выполните основной скрипт программы, которая автоматически начнет прослушивать сетевой трафик. Она будет захватывать пакеты и анализировать их на предмет подозрительных действий, таких как сканирование портов или аномальные запросы. Если программа обнаружит угрозу, она покажет это вам, вы можете заблокировать IP, после чего он попадает в список заблокированных IP. Также вы можете разблокировать этот IP. Программа будет работать в фоновом режиме, и для её остановки можно просто прервать выполнение скрипта, нажав завершив скрипт.

Права доступа: Убедитесь, что у программы есть необходимые права для чтения данных в исходном каталоге и записи в каталог для резервных копий. Также настройте права на файл журнала, чтобы программа могла записывать ошибки и успешные операции.

Рекомендации программиста

Для успешной разработки программы изучите сетевые протоколы (IP, TCP, UDP, ICMP) и документацию Scapy для эффективного анализа пакетов. Организуйте код модульно, разделив функции захвата, анализа и блокировки трафика. Добавьте логирование для фиксации угроз и регулярно обновляйте правила обнаружения. Тщательно тестируйте программу в разных условиях, чтобы избежать ложных срабатываний, и оптимизируйте её для работы в реальном времени при высокой нагрузке. Это обеспечит надёжную и эффективную работу программы.

Исходный код программы:

https://github.com/hanglider/Jasur-Labs/tree/main/linux

Контрольный пример

Запустите main.py (Рис. 1)

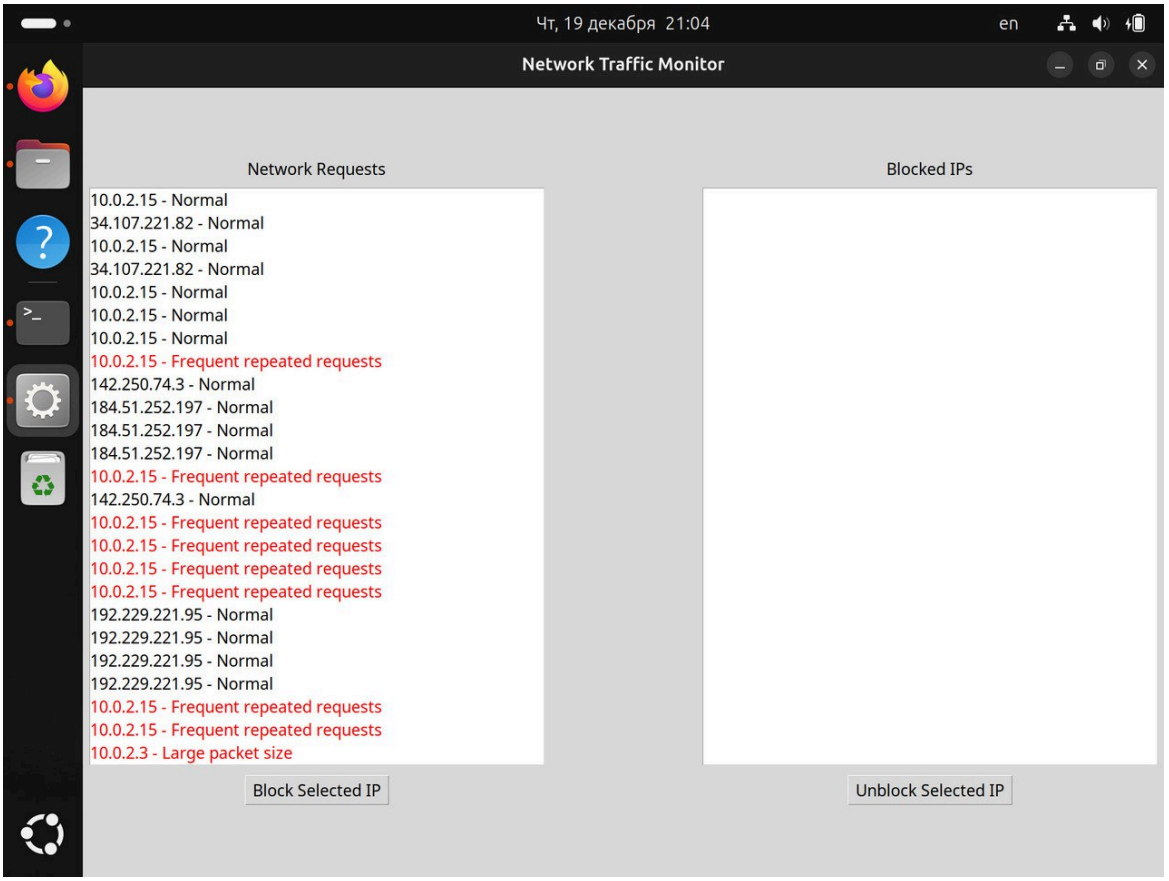

Рис.1 Рабочая область

Выберите подозрительный пакет и заблокируйте его, он попадет в список заблокированных (Рис. 2)

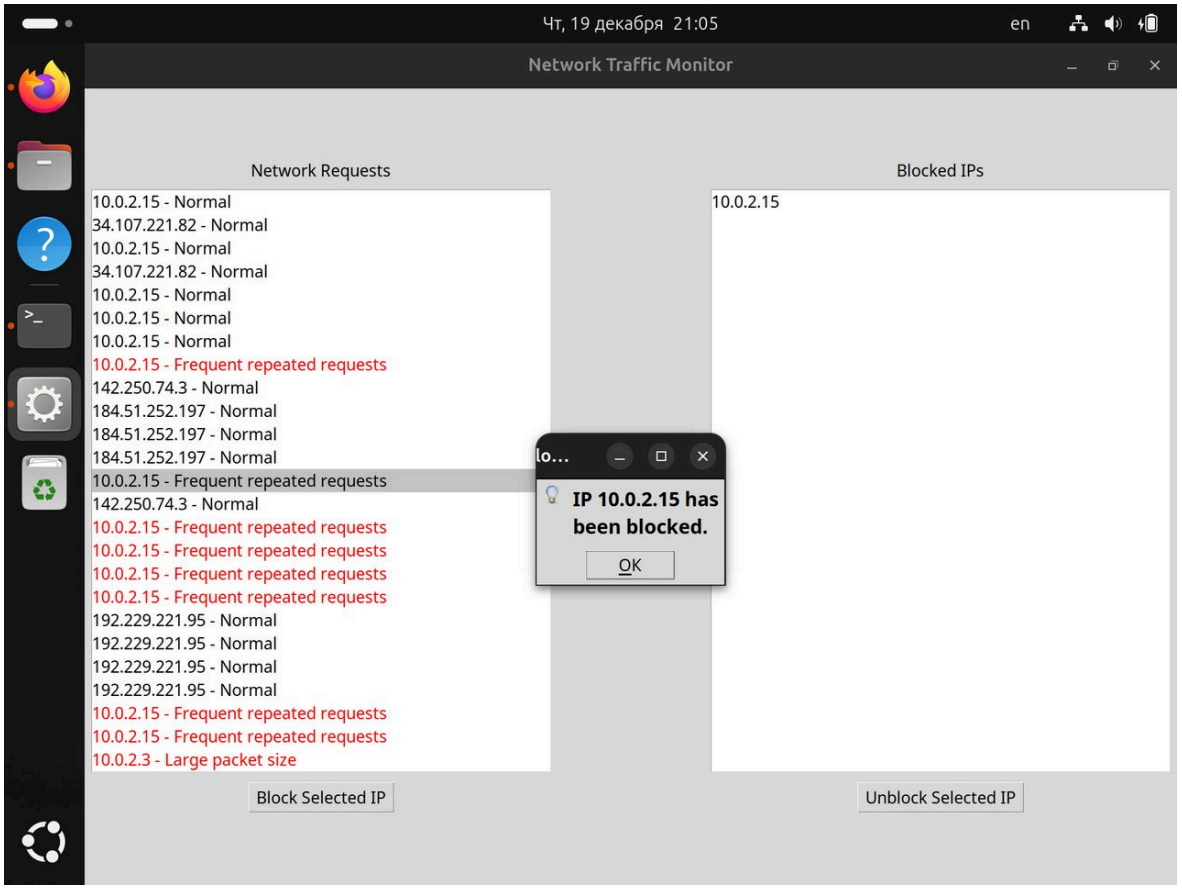

Рис.2

Также можно разблокировать заблокированный Ip, нужно выбрать его и нажать кнопку “разблокировать” (Рис. 3)

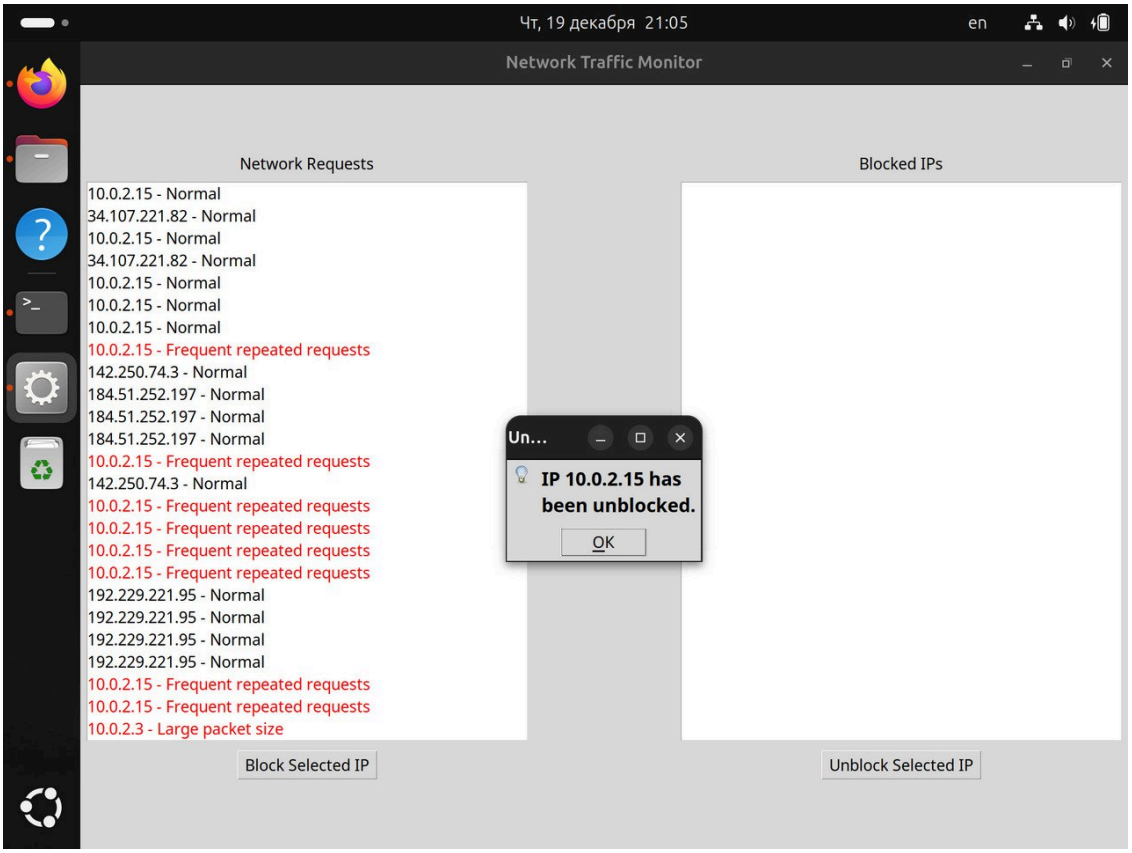

Рис.3

Вывод

В ходе выполнения работы была разработана программа для обнаружения и блокировки подозрительного сетевого трафика. Программа успешно выполняет захват, анализ и обработку сетевых пакетов в реальном времени, а также принимает меры для предотвращения возможных угроз. Реализованный функционал позволяет эффективно реагировать на подозрительный трафик, обеспечивая защиту сети. Дальнейшее развитие программы возможно за счёт обновления правил обнаружения, улучшения алгоритмов анализа и оптимизации производительности.

Источники

Scapy// https://docs.python.org/3/library/scapy.html *(дата обращения: 19.12.2024)*