**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Системное программирование в Linux»**

**на тему**

**«Обнаружение и блокировка подозрительного сетевого трафика»**

**Студент гр. 23Б15-пу**

**Черевко М.Е.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc184699083)
2. [Описание задачи (формализация задачи) 4](#_Toc184699084)
3. [Теоретическая часть 5](#_Toc184699085)

[**Методы обнаружения подозрительного трафика** 5](#_Toc184699086)

[**Библиотека Scapy для анализа трафика** 5](#_Toc184699087)

[**Блокировка подозрительного трафика с помощью iptables** 6](#_Toc184699088)

1. [Основные шаги программы 7](#_Toc184699089)
2. [Описание программы 9](#_Toc184699090)
3. [Рекомендации пользователя 11](#_Toc184699091)
4. [Рекомендации программиста 13](#_Toc184699092)

[**1. Установка необходимых библиотек Python**: 13](#_Toc184699093)

[**2**. **Установка утилит для работы с iptables:** 13](#_Toc184699094)

1. [Исходный код программы 13](#_Toc184699095)
2. [Контрольный пример 14](#_Toc184699096)
3. [Вывод 15](#_Toc184699097)
4. [Источники 16](#_Toc184699098)

# Цель работы

Цель работы — разработать систему мониторинга и защиты сети, способную выявлять и блокировать подозрительный сетевой трафик. В рамках работы создается скрипт на Python для прослушивания и анализа входящего трафика с использованием библиотеки **Scapy**. Система анализирует параметры пакетов для обнаружения потенциально вредоносной активности, такой как аномальные объемы данных или частые повторяющиеся запросы, и при необходимости блокирует подозрительные IP-адреса. При этом необходимо реализовать возможность разблокировать источник в случае отсутствия исходящей угрозы.

# Описание задачи (формализация задачи)

Задача предполагает следующие этапы:

1. **Сбор и анализ данных о сетевом трафике**: программа должна уметь перехватывать входящие сетевые пакеты, фиксировать их основные параметры, такие как IP-адрес источника, номер порта и размер пакета. Эта информация будет отображаться в графическом интерфейсе, что позволит визуализировать активные соединения.
2. **Определение критериев подозрительного трафика**: необходимо разработать и внедрить правила обнаружения аномального поведения, которые позволят отфильтровывать подозрительные пакеты. Такие правила могут включать:
   * Обнаружение IP-адресов, передающих аномально большие объемы данных.
   * Обнаружение IP-адресов, часто повторяющих запросы в короткий промежуток времени.
   * Распознавание подозрительных действий, таких как порт-сканирование (много соединений на разные порты от одного IP).
3. **Выявление подозрительного трафика**: при обнаружении пакета, удовлетворяющего условиям подозрительности, IP-адрес должен быть занесен в список подозрительных IP. В графическом интерфейсе для этих IP-адресов должно указываться, какая именно аномалия была выявлена.
4. **Блокировка подозрительных IP-адресов**: при нажатии пользователем кнопки блокировки для конкретного подозрительного IP программа должна добавить правило в **iptables** для блокировки данного IP-адреса, предотвращая дальнейшую передачу данных с него. При этом заблокированный IP должен отобразиться в отдельном списке, и для него должна быть доступна функция разблокировки.

Таким образом, задача заключается в создании программного обеспечения, которое будет обеспечивать анализ сетевого трафика в реальном времени, выявлять подозрительные соединения по заданным правилам и оперативно блокировать их по мере необходимости.

# Теоретическая часть

Мониторинг сетевого трафика — это процесс сбора и анализа данных, передаваемых по сети, для обеспечения безопасности и эффективности сетевой инфраструктуры. Одной из основных задач мониторинга является обнаружение аномалий и угроз, таких как сканирование портов, чрезмерная передача данных от определенных источников или повторяющиеся запросы, которые могут свидетельствовать о сетевых атаках.

### **Методы обнаружения подозрительного трафика**

Для анализа сетевого трафика применяются различные методы и подходы:

1. **Анализ сигнатур**. Этот метод основывается на заранее известных шаблонах поведения, которые характерны для определенных типов атак. Например, DDoS-атаки часто включают в себя большое количество запросов с одного или нескольких IP-адресов. Недостаток метода — ограниченная способность обнаруживать новые виды атак, которые не имеют сигнатур.
2. **Поведенческий анализ**. В отличие от сигнатурного, поведенческий анализ ищет аномалии, отклоняющиеся от нормы, например, непропорционально большие объемы трафика от одного IP или частое сканирование портов. Поведенческий анализ эффективен против новых атак, так как он ориентируется на необычные изменения в сетевом поведении.
3. **Использование статистических методов**. Статистический анализ позволяет отслеживать метрики сетевого трафика, такие как частота запросов, объем передаваемых данных и распределение пакетов по времени. На основе статистики можно строить пороговые значения, за **рамки которых выходят подозрительные активности.**

### **Библиотека Scapy для анализа трафика**

**Scapy** — это мощная библиотека для Python, которая позволяет работать с сетевыми пакетами. Она предоставляет возможности для создания, отправки, получения и анализа пакетов на различных уровнях сетевой модели OSI. **Scapy** может использоваться для сниффинга (прослушивания) трафика, что делает ее идеальным инструментом для обнаружения аномалий в сетевом поведении.

Основные функции, которые предоставляет Scapy, включают:

* **Перехват пакетов**: позволяет прослушивать трафик в режиме реального времени, обрабатывая входящие пакеты и извлекая из них полезную информацию (например, IP-адреса источника и назначения, порты, размеры пакетов и заголовки).
* **Создание и отправка пакетов**: **Scapy** поддерживает формирование и отправку пакетов, что может быть полезно для активного сетевого тестирования и отправки управляющих ICMP-сообщений для блокировки.

### **Блокировка подозрительного трафика с помощью iptables**

Для блокировки подозрительных IP-адресов часто используются сетевые фильтры, такие как **iptables** — утилита, которая позволяет создавать правила фильтрации на уровне ядра **Linux**. **iptables** предоставляет гибкие возможности для блокировки и разрешения трафика на основе адресов, портов и других параметров. В рамках данной задачи блокировка подозрительного трафика осуществляется путем добавления правил для запрета входящих соединений с определенных IP-адресов, что предотвращает их взаимодействие с сетевой инфраструктурой.

# Основные шаги программы

1. **Инициализация GUI**:
   * При запуске программы создается графический интерфейс с использованием tkinter, который включает в себя три таблицы:
     + Для отображения всех входящих IP-адресов и соответствующих данных (порт, размер пакета).
     + Для отображения подозрительных IP-адресов.
     + Для отображения заблокированных IP-адресов.
   * Кнопки управления мониторингом и блокировкой/разблокировкой IP-адресов.
2. **Запуск мониторинга**:
   * Пользователь нажимает кнопку "Start Monitoring", что инициирует процесс захвата пакетов.
   * В фоновом потоке запускается функция для анализа входящих пакетов с помощью Scapy (scapy.sniff).
3. **Обработка пакетов**:
   * При получении пакета программа анализирует его источник (IP-адрес) и размер.
   * Если пакет имеет размер, превышающий заданный порог (например, 200 байт), этот IP-адрес добавляется в список подозрительных.
   * Информация о всех входящих IP-адресах добавляется в таблицу "All Incoming IPs" для отображения пользователю.
4. **Выявление подозрительных IP-адресов**:
   * IP-адреса с высоким трафиком (более 200 байт) автоматически добавляются в таблицу "Suspicious IPs" с пометкой "High Traffic".
   * Пользователь может заблокировать подозрительные IP-адреса, выбрав их и нажав кнопку "Block Selected IP".
5. **Блокировка IP-адресов**:
   * При блокировке IP-адреса с помощью iptables, адрес добавляется в таблицу "Blocked IPs", а также записывается правило в iptables для блокировки входящих соединений с этим IP.
6. **Разблокировка IP-адресов**:
   * Пользователь может разблокировать IP-адрес, выбрав его из таблицы "Blocked IPs" и нажав кнопку "Unblock Selected IP".
   * При этом правило в iptables удаляется, и IP-адрес убирается из списка заблокированных.
7. **Остановка мониторинга**:
   * Пользователь может остановить процесс мониторинга, нажав кнопку "Stop Monitoring", что прекращает захват пакетов.
8. **Обновление интерфейса**:
   * В реальном времени таблицы интерфейса обновляются с информацией о новых входящих, подозрительных и заблокированных IP-адресах.
9. **Работа с iptables**:

* При блокировке IP добавляется правило: **sudo iptables -A INPUT -s <ip\_address> -j DROP**. Добавляет правило (-A) в цепочку INPUT для блокировки всех пакетов, поступающих от указанного IP-адреса (-s <ip\_address>). Действие (-j DROP) указывает, что такие пакеты должны быть "отброшены" (без ответа).
* При разблокировке удаляется все правила для указанного IP: **sudo iptables -D INPUT -s <ip\_address> -j DROP.** Удаляет правило (-D) из цепочки INPUT, которое соответствует блокировке пакетов от указанного IP-адреса (-s <ip\_address>). После выполнения пакеты от этого IP снова будут проходить через брандмауэр.

1. **Завершение программы**:

* Программа завершает работу при закрытии окна GUI.

# Описание программы

Программа мониторинга сетевого трафика разработана на языке Python с использованием библиотеки Scapy для захвата и анализа сетевых пакетов. Для удобства пользователя был создан графический интерфейс с помощью tkinter, который позволяет контролировать процесс мониторинга трафика. Важнейшие функции программы включают анализ входящих пакетов для выявления подозрительных IP-адресов, а также возможность блокировать и разблокировать IP-адреса с использованием iptables. Информация о подозрительных и заблокированных IP-адресах отображается в интерфейсе. Логика программы организована в классе TrafficMonitor, который объединяет все ключевые функции, такие как обработка пакетов, запуск мониторинга и управление блокировкой IP-адресов.

Таблица 1. TrafficMonitorApp.py

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Описание | Возвращаемое значение | Параметры функции |
| \_\_init\_\_(self, root\_window) | |  | | --- | |  |  |  | | --- | | Инициализация интерфейса программы, создание всех GUI компонентов. | | None | root\_window — объект окна Tkinter для интерфейса |
| setup\_gui(self) | |  | | --- | |  |  |  | | --- | | Создание и настройка всех графических компонентов (таблиц, кнопок и меток) в интерфейсе. | | None | |  | | --- | |  |  |  | | --- | | None | |
| packet\_handler(self, pkt) | |  | | --- | |  |  |  |  |  | | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | Обработка пакетов, выявление подозрительных IP-адресов на основе размера трафика и обновление таблиц. | | | None | pkt — перехваченный сетевой пакет (тип scapy.Packet). |
| start\_sniffing(self) | |  | | --- | |  |  |  | | --- | | Запуск процесса мониторинга трафика, начало захвата пакетов. | | None | |  | | --- | |  |  |  | | --- | | None | |
| stop\_sniffing(self) | |  | | --- | |  |  |  | | --- | | Остановка процесса мониторинга трафика и захвата пакетов. | | None | None |
| block\_ip(self) | Блокировка выбранного подозрительного IP-адреса с добавлением правила в iptables. | None | None |
| unblock\_ip(self) | |  | | --- | |  |  |  | | --- | | Разблокировка выбранного заблокированного IP-адреса с удалением правила из iptables. | | None | None |
| add\_iptables\_rule(self, ip\_address) | Добавление правила для блокировки IP-адреса в iptables. | None | ip\_address — строка с IP-адресом для блокировки. |
| remove\_iptables\_rule(self, ip\_address) | Удаление правила блокировки для IP-адреса из iptables. | None | ip\_address — строка с IP-адресом для разблокировки. |

# Рекомендации пользователя

* 1. **Запуск программы:**
* Запустите программу, чтобы открыть графический интерфейс. Убедитесь, что у вас есть права администратора для использования iptables (вам потребуется вводить пароль sudo при блокировке и разблокировке IP).
  1. **Начало мониторинга:**
* Для начала мониторинга сетевого трафика нажмите кнопку **"Start Monitoring"**. Программа начнет захват пакетов с сети и отображать информацию о входящих IP-адресах в таблице **"All Incoming IPs"**.
  1. **Просмотр подозрительных IP-адресов**:
* В процессе мониторинга программа будет анализировать пакеты и выявлять подозрительные IP-адреса с высоким трафиком. Эти IP-адреса будут отображаться в таблице **"Suspicious IPs"** с пометкой **"High Traffic"**.
  1. **Блокировка подозрительных IP**:
* Если вы обнаружите подозрительные IP-адреса, которые нужно заблокировать, выберите их в таблице **"Suspicious IPs"** и нажмите кнопку **"Block Selected IP"**.
* Эти IP-адреса будут добавлены в таблицу **"Blocked IPs"**, а соответствующие правила блокировки будут автоматически добавлены в iptables.
  1. **Разблокировка IP-адресов**:
* Для разблокировки заблокированных IP-адресов перейдите в таблицу **"Blocked IPs"**, выберите нужный IP и нажмите кнопку **"Unblock Selected IP"**.
* Это удалит правило iptables для данного IP-адреса, и он снова станет доступен.
  1. **Остановка мониторинга**:
* Когда вы захотите прекратить мониторинг, нажмите кнопку **"Stop Monitoring"**. Это остановит захват пакетов и отключит отображение новых данных в таблицах.
  1. **Обновление интерфейса**:
* Интерфейс будет автоматически обновляться в реальном времени, показывая новые входящие IP-адреса, подозрительные и заблокированные IP. Обратите внимание на изменения в таблицах.
  1. **Использование прав администратора**:
* Для того, чтобы заблокировать или разблокировать IP-адреса с использованием iptables, вам потребуется права администратора. Программа запросит ваш пароль при выполнении команд для изменения правил iptables.
  1. **Безопасность**:
* Используйте программу осторожно, так как блокировка IP-адресов может повлиять на вашу сеть. Блокировка ошибок может вызвать потерю связи с важными устройствами.
* Убедитесь, что вы блокируете только те IP-адреса, которые действительно представляют угрозу, и всегда проверяйте их перед блокировкой.
  1. **Завершение работы программы**:
* Чтобы завершить работу программы, просто закройте окно с интерфейсом. Программа автоматически прекратит захват пакетов и остановит все процессы.

# Рекомендации программиста

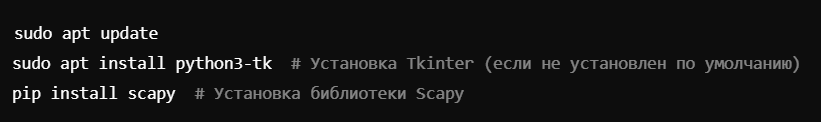
Для успешной работы программы **TrafficMonitorApp** потребуется установить несколько библиотек и утилит. Ниже представлены шаги, которые **необходимо выполнить:**

### **1. Установка необходимых библиотек Python:**

Программа использует библиотеку **Scapy** для захвата и анализа сетевых пакетов и **Tkinter** для создания графического интерфейса. Обе библиотеки могут быть установлены с помощью менеджера пакетов pip.

* **Scapy** — библиотека для работы с пакетами на сетевом уровне.
* **Tkinter** — стандартная библиотека Python для создания графических интерфейсов.

Для установки **Scapy** и **Tkinter** выполните следующие команды в терминале:



### **2**. **Установка утилит для работы с iptables:**

Программа использует утилиту iptables для блокировки и разблокировки IP-адресов. Убедитесь, что она установлена в вашей системе.

Для установки iptables выполните команду:

****

**3.** **Запуск программы с правами администратора:**

Для блокировки и разблокировки IP-адресов через **iptables** программе могут потребоваться права администратора. Запустите программу с использованием команды **sudo**, чтобы избежать ошибок доступа:



# Исходный код программы

<https://github.com/FraaaM/ip_traffic_scanner>

# Контрольный пример

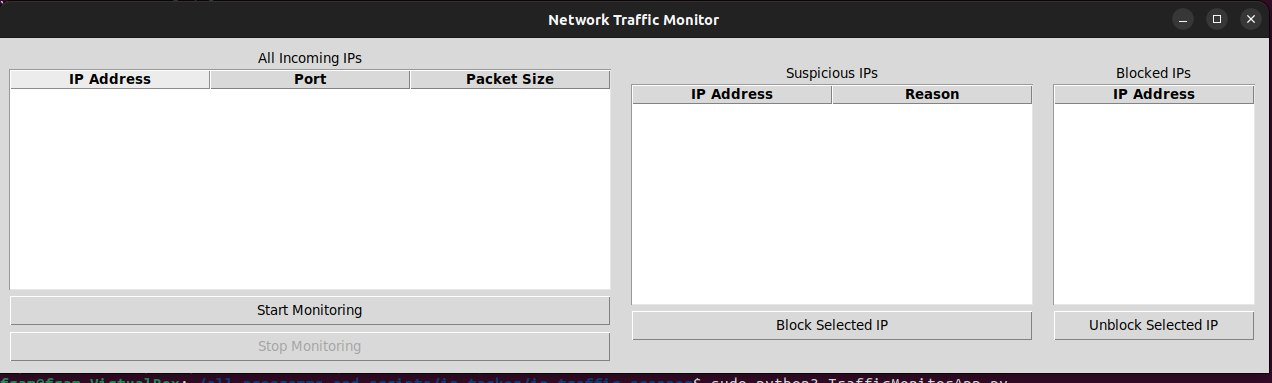
* 1. **Запуск исходного файла:** запустите файл **TrafficMonitorApp.py**, появится графический интерфейс (Рис. 1):

Рис. 1 Графический интерфейс

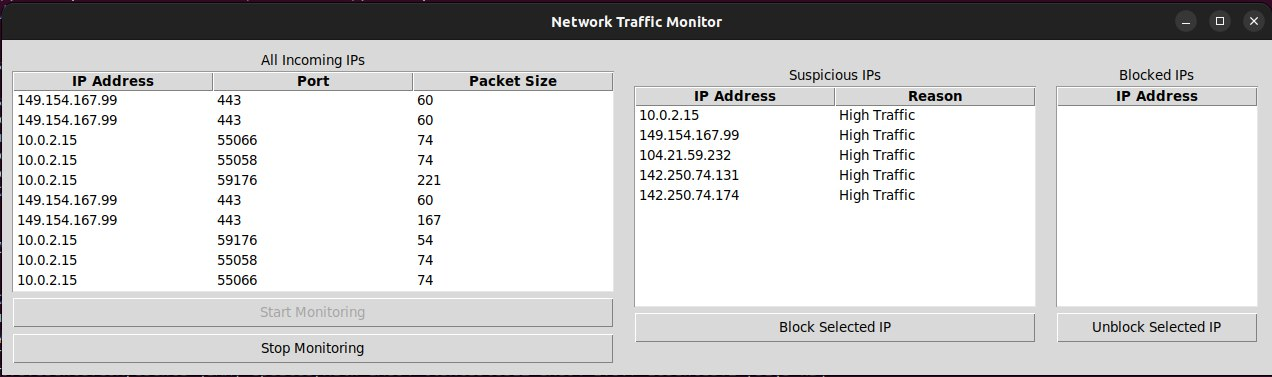
* 1. **Запуск мониторинга входящего сетевого трафика:** нажимаем на кнопку – “Start” и ждём, когда появятся первые трафики, а в месте с ними и подозрительные. (Рис. 2):

Рис. 2 Мониторинг

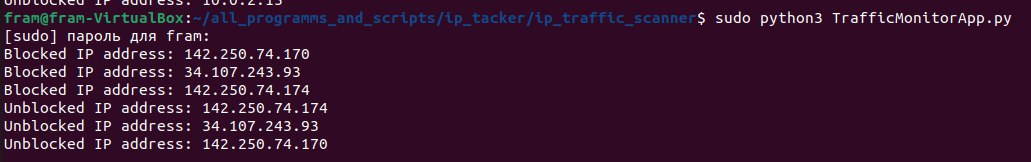
* 1. **Блокировка и Разблокировка:** выбираем какой-то адрес из таблицы подозрительных и блокируем, аналогично с разблокировкой. (Рис.3 и 4):

Рис. 3 Вывод в консоль

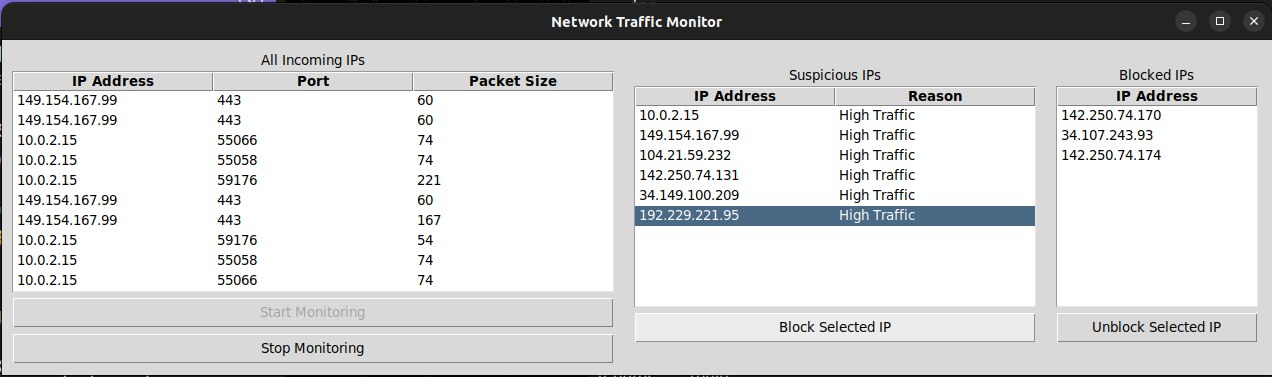


Рис. 4 Блокировка подозрительного трафика

# Вывод

Программа мониторинга сетевого трафика предоставляет удобный и эффективный инструмент для отслеживания и анализа входящих сетевых пакетов с использованием Python и библиотеки Scapy. С помощью графического интерфейса, созданного на Tkinter, пользователи могут легко управлять процессом мониторинга, блокировки и разблокировки IP-адресов.

Основные возможности программы включают:

* Мониторинг входящих IP-адресов в реальном времени.
* Выявление подозрительных IP-адресов с аномально высоким трафиком.
* Блокировка и разблокировка IP-адресов с использованием iptables.
* Отображение информации о текущих активных, подозрительных и заблокированных IP-адресах через таблицы.

Программа помогает повысить безопасность сети путем выявления и блокировки потенциально вредоносных источников трафика. Однако важно подходить к использованию блокировки IP-адресов осторожно, чтобы избежать случайного отключения важных сервисов или устройств.

Интуитивно понятный графический интерфейс и возможность работать с iptables через командную строку делают эту программу полезным инструментом для администраторов и пользователей, заинтересованных в защите своей сети от несанкционированного доступа.

# Источники

• scapy — Для захвата и анализа сетевого трафика.  
<https://ru.wikipedia.org/wiki/Scapy>

*Дата обращения: (10.12.2024)*

• tkinter — Для создания графического интерфейса.  
<https://docs.python.org/3/library/tkinter.html>

*Дата обращения: (10.12.2024)*

• subprocess — Для работы с системными процессами.  
<https://docs.python.org/3/library/subprocess.html>

*Дата обращения: (10.12.2024)*

• threading — Для организации многозадачности и параллельных потоков.  
[*https://docs.python.org/3/library/threading.html*](https://docs.python.org/3/library/threading.html)

*Дата обращения: (10.12.2024)*