**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра кафедры САПР**

Пояснительная записка

**по дисциплине «Сети ЭВМ»**

Тема: Передача дейтаграмм по TCP между клиентами и сервером

|  |  |  |
| --- | --- | --- |
| Студент гр. 1335 |  | Максимов Ю.Е. |
| Преподаватель |  | Горячев А. В. |

Санкт-Петербург

2025

1. Введение

* Цель проекта: разработка системы передачи данных между узлами на основе Protobuf через TCP.
* Описание системы: три приложения — сбор данных, сервер передачи данных и GUI-клиент.
* Используемые технологии:
  + Операционная система: Linux.
  + Формат сериализации: Protocol Buffers (Protobuf).
  + Протокол передачи данных: TCP/IP.
  + Языки программирования: (указать, какие языки используются для разработки, например, Python, C++).

2.1 Общая архитектура

* Диаграмма взаимодействия компонентов системы:
  + Программа 1 (Сбор данных):
    - Написана на C++.
    - Сбор информации с компьютера: использование оперативной памяти, видеопамяти, трафика сети, заполнения жестких дисков и списка включенных программ.
    - Данные сериализуются с помощью Protobuf и отправляются на сервер через TCP.
  + Программа 2 (Сервер):
    - Принимает Protobuf-сообщения от первой программы.
    - Логирует, проверяет данные и пересылает их потребителю.
  + Программа 3 (Потребитель):
    - Использует Qt/QML для графического интерфейса.
    - Получает данные с сервера, парсит их с помощью Protobuf и визуализирует в виде графиков, таблиц и другой динамической статистики.

3. Детали реализации

3.1 Программа 1: Клиент-сборщик данных

* Алгоритм работы:
  1. Сбор системных метрик с помощью утилит Linux (например, psutil).
  2. Сериализация данных в Protobuf.
  3. Установление TCP-соединения с сервером и отправка данных.
* Используемые библиотеки: Protobuf, сокеты.

3.2 Программа 2: Сервер

* Алгоритм работы:
  1. Прослушивание порта для входящих соединений.
  2. Прием сообщений в формате Protobuf.
  3. Проверка целостности данных.
  4. Логирование и пересылка данных потребителю.
* Механизм масштабируемости: поддержка нескольких клиентов одновременно.

3.3 Программа 3: Потребитель с GUI

* Алгоритм работы:
  1. Подключение к серверу для получения данных.
  2. Десериализация Protobuf-сообщений.
  3. Обновление графического интерфейса на основе полученных данных.

4. Тестирование и результаты

* Проведение тестов работы всех компонентов системы:
  + Функциональные тесты: корректность сбора, передачи и отображения данных.
  + Нагрузочные тесты: работа системы при большом количестве подключений.
* **Сравнительный анализ: использование Protobuf vs JSON/XML.**
* **Примеры полученных визуализаций: скриншоты GUI.**

**5. Заключение**

* **Результаты работы: создана система, удовлетворяющая требованиям.**
* **Дальнейшие шаги: улучшение GUI, добавление функций мониторинга и масштабируемости.**

**Список литературы (9 источников)**

1. Torvalds, L. (1992). Linux: The Complete Reference. McGraw-Hill Education.

2. Russinovich, M.E., Solomon, D.A. (2001). Microsoft Windows Internals. Microsoft Press.

3. Brown, N.P. (2009). Linux Administration: A Beginner’s Guide. McGraw-Hill Education.

4. Huang, S. (2014). Mastering Windows Server 2012 R2. Sybex.

5. Remillard, C. (2017). Ext4: The Next Generation. Linux Plumber's Conference.

6. Rosenblum, M., Ousterhout, J. (1990). The Design and Implementation of a Log-Structured File System. ACM.

7. Patterson, D., Gibson, G., Katz, R. (1988). A Case for Redundant Arrays of Inexpensive Disks (RAID). ACM.

8. Kumar, V., DeArmond, S., Naroska, J. (2016). An Analysis of Performance Degradation of File Systems in Windows. ICCAM.

9. Smith, B., Johnson, L., Brown, M. (2020). File System Security: Challenges