**NET**

1. Простой вариант (одно из):

• Написать фильтр сетевых пакетов на основе nfqueue и iptables и протестировать

скорость работы

• Протестировать работу сокетов tcp при различных настройках setsockopt

2. Сложный вариант (одно из):

• Написать фильтр пакетов на основе интерфейса netfilter

• **Реализовать rpc-программу для linux с поддержкой аутентификации**

**(rpcinfo,rpcbind)**

* **Remote Procedure Call (RPC)** thường được sử dụng để gọi các hàm từ xa trên máy chủ yêu cầu kết quả hành động. Bạn có thể sử dụng khi bạn yêu cầu tính toán phức tạp hoặc muốn kích hoạt một thủ tục từ xa trên máy chủ, với quá trình ẩn khỏi máy khách.
* Viết server tính toán sau đó thực hiện tính toán và xác thực = **(rpcinfo,rpcbind)**
* **(rpcinfo,rpcbind)**

**rpcinfo và rpcbind là hai công cụ quan trọng trong hệ thống RPC (Remote Procedure Call) của Unix/Linux.**

1. **rpcinfo:**
   * **Hiển thị thông tin về các dịch vụ RPC đang chạy trên một máy chủ.**
   * **Cung cấp danh sách các phiên bản RPC, chương trình, và giao thức đang được sử dụng.**
   * **Có thể kiểm tra xem dịch vụ RPC nào đang hoạt động và có thể kết nối được hay không.**
2. **rpcbind:**
   * **Là dịch vụ cho phép các ứng dụng RPC tìm thấy nhau trên mạng.**
   * **Khi một ứng dụng RPC khởi động, nó đăng ký với rpcbind, cho phép rpcbind biết địa chỉ và thông tin phiên bản của dịch vụ đó.**
   * **Khi một client muốn gọi một dịch vụ RPC, nó sẽ liên hệ với rpcbind để lấy thông tin về dịch vụ cụ thể.**

máy khác hoàn toàn có thể liên hệ với dịch vụ RPC của server, miễn là các điều kiện sau được thỏa mãn:

1. **Mạng kết nối**: Máy client và server phải nằm trên cùng một mạng hoặc có thể kết nối với nhau qua mạng (internet hoặc VPN).
2. **Cấu hình firewall**: Tường lửa trên server phải cho phép kết nối đến cổng mà dịch vụ RPC đang lắng nghe. Thông thường, rpcbind chạy trên cổng 111, còn các dịch vụ RPC khác có thể sử dụng các cổng động.
3. **Dịch vụ đang chạy**: Dịch vụ RPC mà máy client muốn kết nối phải đang hoạt động và được đăng ký với rpcbind trên server.
4. **Quyền truy cập**: Có thể có các biện pháp bảo mật, như xác thực hoặc quyền truy cập, yêu cầu client phải có quyền hợp lệ để kết nối với dịch vụ.

Nhiệm vụ là (**không cài đc netfilter!)**

Sử dụng netfilter để thêm và kiểm tra xem trước khi có filter và sau khi có netfilter tốc độ như thế nào

Nfqueue : net fiter queue

**NFQUEUE (Netfilter Queue)** là một cơ chế trong Netfilter, một phần của tường lửa Linux, cho phép bạn chuyển các gói tin (packets) từ một chuỗi (chain) Netfilter đến không gian người dùng (user-space) để xử lý. Điều này hữu ích trong các tình huống cần phân tích, thay đổi hoặc quyết định về số phận của các gói tin dựa trên các yếu tố mà kernel không hỗ trợ hoặc xử lý đơn giản

**NFQUEUE** và **iptables quản lý và xử lý các gói tin trong hệ thống mạng Linux.** vai trò và chức năng khác nhau trong việc kiểm soát luồng gói tin qua hệ thống. iptables là một công cụ dòng lệnh để thiết lập, duy trì và kiểm tra các bảng (**tables**) của Netfilter trong nhân Linux. sử dụng để cấu hình các quy tắc tường lửa, cho phép hoặc chặn các gói tin mạng dựa trên nhiều tiêu chí khác nhau (như địa chỉ IP, cổng, giao thức, v.v.).

#### **Các khái niệm cơ bản của iptables:**

* **Tables (Bảng)**: iptables có nhiều bảng, nhưng hai bảng chính là:
  + filter: Bảng chính cho phép hoặc chặn gói tin.
  + nat: Được dùng cho việc dịch địa chỉ mạng (NAT).
* **Chains (Chuỗi)**: Một chuỗi là một tập hợp các quy tắc. Ví dụ về chuỗi trong bảng filter:
  + INPUT: Xử lý các gói tin đi vào hệ thống.
  + OUTPUT: Xử lý các gói tin rời khỏi hệ thống.
  + FORWARD: Xử lý các gói tin được chuyển tiếp qua hệ thống.
* **Rules (Quy tắc)**: Mỗi quy tắc trong một chuỗi chỉ định điều kiện mà các gói tin phải thỏa mãn, và hành động (target) cần thực hiện trên các gói tin đó (ví dụ: ACCEPT, DROP).

Ví dụ, một quy tắc cơ bản để chặn toàn bộ các gói tin đến từ một địa chỉ IP có thể như sau:

bash

Sao chép mã

iptables -A INPUT -s 192.168.1.100 -j DROP

Câu lệnh này sẽ thêm (-A) một quy tắc vào chuỗi INPUT để chặn (DROP) tất cả các gói tin từ địa chỉ IP 192.168.1.100.

### **2. NFQUEUE**

**NFQUEUE** là một mục tiêu (target) trong Netfilter, được sử dụng với iptables hoặc nftables để chuyển gói tin từ không gian kernel sang không gian người dùng.

cho phép một chương trình không gian người dùng (user-space program) xử lý gói tin trước khi quyết định cho phép, chặn, hoặc sửa đổi gói tin đó.

**iptables**:

* Sử dụng để **thiết lập các quy tắc tường lửa** trong kernel Linux.
* Có thể chặn (DROP), cho phép (ACCEPT), hoặc thực hiện hành động khác (như LOG, REJECT).

**NFQUEUE**:

* Là một trong những hành động (target) mà iptables có thể chỉ định cho các gói tin.
* Dùng để **chuyển gói tin sang không gian người dùng**, nơi bạn có thể viết chương trình để **xử lý chi tiết** gói tin (ví dụ: kiểm tra an ninh, lọc, phân tích).

->sử lí gói tin trước khi đưa vào nhân, tường lửa cấp độ cao hơn hoặc giúp CPU nhanh hơn

Tpc : Transmission Control Protocol

1.

• Viết bộ lọc gói mạng dựa trên nfqueue và iptables và kiểm tra tốc độ hoạt động

• Kiểm tra hoạt động của ổ cắm tcp với các cài đặt setsockopt khác nhau

2. Phương án khó (một trong):

• Viết bộ lọc gói dựa trên giao diện netfilter

• Triển khai chương trình rpc cho linux có hỗ trợ xác thực (rpcinfo,rpcbind)

Caif rpc

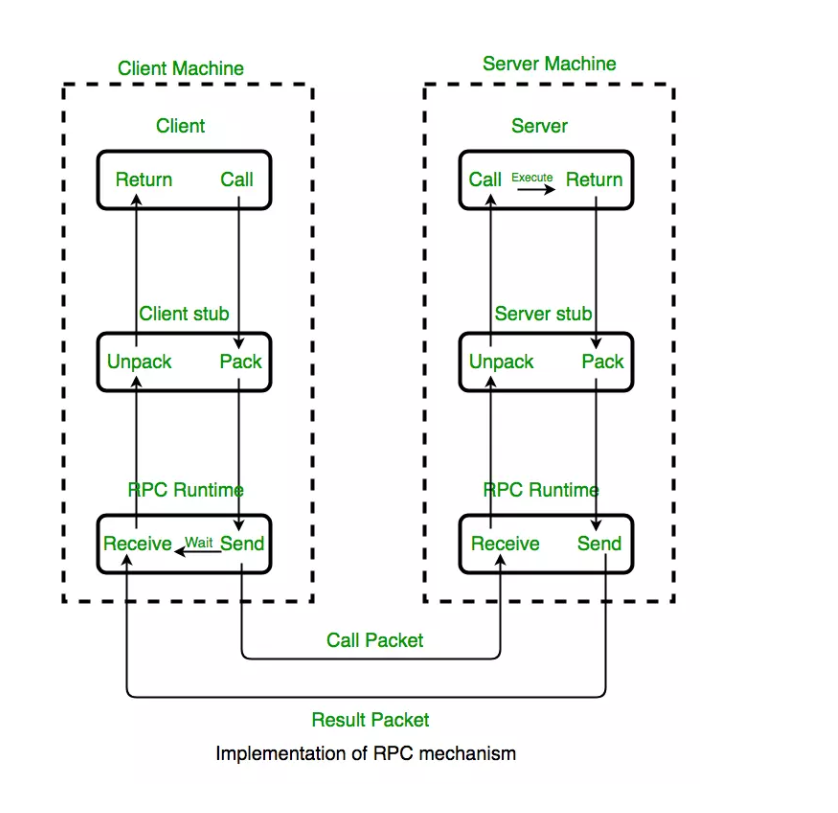
**sudo systemctl start rpcbind**

Kichs hoatj rpc

**sudo systemctl start rpcbind**

Rpc = remote procedure call

<https://viblo.asia/p/gioi-thieu-ve-rpc-va-grpc-E1XVOxOZ4Mz>



- Viết mã Python cho bộ lọc gói mạng dựa trên nfqueue và iptables

và kiểm tra tốc độ của máy chủ google.com

- tường lửa quyết định gói nào được phép vào và rời khỏi hệ thống. tường lửa được cung cấp bằng netfilter. iptables chỉ là một giao diện cho netfilter.

- Nếu chuỗi INPUT không có nghĩa là an toàn hơn, vì khi mạng hoạt động thì cả INPUT và OUTPUT đều được sử dụng.

- Sau khi hoàn thành công việc, chúng tôi viết chương trình rpc trên Linux có hỗ trợ xác thực.

Giải thích code

1

2.

Server

#include "IDL.h"

float \*

add\_6\_svc(values \*argp, struct svc\_req \*rqstp)

{

static float result;

// Perform addition

result = argp->num1 + argp->num2;

return &result;

}

float \*

sub\_6\_svc(values \*argp, struct svc\_req \*rqstp)

{

static float result;

// Perform subtraction

result = argp->num1 - argp->num2;

return &result;

}

float \*

mul\_6\_svc(values \*argp, struct svc\_req \*rqstp)

{

static float result;

// Perform multiplication

result = argp->num1 \* argp->num2;

return &result;

}

float \*

div\_6\_svc(values \*argp, struct svc\_req \*rqstp)

{

static float result;

// Perform division, check for division by zero

if (argp->num2 != 0) {

result = argp->num1 / argp->num2;

} else {

// Handle division by zero case (returning an error value, for example)

result = 0; // or some error code

}

return &result;

}

Nó định nghĩa các hàm cho phép thực hiện các phép toán cơ bản (cộng, trừ, nhân, chia) trên hai số thực. Dưới đây là giải thích chi tiết từng phần:

### **Cấu trúc**

1. **Hàm add\_6\_svc**:
   * Nhận một tham số kiểu values, có hai thuộc tính num1 và num2.
   * Tính tổng của num1 và num2.
   * Trả về địa chỉ của biến result, nơi chứa giá trị tổng.
2. **Hàm sub\_6\_svc**:
   * Tương tự như hàm cộng, nhưng thực hiện phép trừ (num1 - num2).
   * Trả về địa chỉ của biến result chứa kết quả.
3. **Hàm mul\_6\_svc**:
   * Thực hiện phép nhân giữa num1 và num2.
   * Trả về kết quả.
4. **Hàm div\_6\_svc**:
   * Thực hiện phép chia giữa num1 và num2.
   * Trước khi thực hiện phép chia, hàm kiểm tra nếu num2 khác 0 để tránh lỗi chia cho 0.
   * Nếu num2 bằng 0, hàm có thể trả về một giá trị mặc định (trong trường hợp này là 0), nhưng có thể cải thiện bằng cách trả về một mã lỗi hoặc thông báo khác.

### **Lưu ý về biến tĩnh**

* Các biến result trong mỗi hàm được khai báo là tĩnh (static). Điều này có nghĩa là giá trị của nó sẽ được duy trì giữa các lần gọi hàm, nhưng có thể gây ra lỗi nếu nhiều yêu cầu RPC đến cùng một lúc, vì các yêu cầu sẽ chia sẻ cùng một biến result.