Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №6-8 по курсу**

**«Операционные системы»**

Студентка:

Варламова Анна Борисовна

Группа: М80-207Б-20

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 27.12.2021

## **Задание**

Реализовать распределенную систему по обработке запросов. В данной системе должно существовать 2 вида узлов: «управляющий » и «вычислительный». Необходимо объединить данные узлы в соответствии с той топологией, которая определена вариантом. Связь между узлами необходимо осуществить при помощи сервера сообщений zmq. Также в данной системе необходимо предусмотреть проверку доступности узлов в соответствии с вариантом.

**Вариант задания:** 7. Топология —3- бинарное дерево. Тип вычислительной команды — 4-поиск подстроки в строке. Тип проверки узлов на доступность —2- пинг узлов по ID.

# **Общие сведения о программе**

Программа состоит из двух файлов, которые компилируются в исполнительные файлы(которые представляют управляющий и вычислительные узлы). Общение между процессами происходит с помощью библиотеки zmq.

# **Общий метод и алгоритм решения**

* Управляющий узел принимает команды, обрабатывает их и пересылает дочерним узлам(или выводит сообщение об ошибке) сокеты на REQ - REP.
* Дочерние узлы проверяют, может ли быть команда выполнена в данном узле, если нет, то команда пересылается в один из дочерних узлов, из которого возвращается некоторое сообщение(об успехе или об ошибке), которое потом пересылается обратно по дереву.
* Для корректной проверки на доступность узлов используется дерево, эмулирующее поведение узлов в данной топологии(например, при удалении узла, удаляются все его потомки).
* При удалении узла все его потомки рекурсивно уничтожаются.

**Код программы**

parent.cpp

#include "zmq.hpp"

#include <sstream>

#include <string>

#include <iostream>

#include <zconf.h>

#include <vector>

#include <signal.h>

#include <sstream>

#include <set>

#include <algorithm>

// g++ parent.cpp -lzmq -o main -w

int main(){

zmq::context\_t context(1); // служебная структура контекст

zmq::socket\_t main\_socket(context, ZMQ\_REP); // поднятие сокета в контексте ZMQ\_REP для отправки запросов и получения ответов

std::string adr = "tcp://127.0.0.1:300";

std::string command;

int child\_id = 0;

while(1){

std::cout << "command:";

std::cin >> command;

if(command == "create") {

if(child\_id == 0){

int id;

std::cin >> id;

int id\_tmp = id - 1;

main\_socket.bind(adr + std::to\_string(++id\_tmp));

std::string new\_adr = adr + std::to\_string(id\_tmp);

char\* adr\_ = new char[new\_adr.size() + 1];

memcpy(adr\_, new\_adr.c\_str(), new\_adr.size() + 1);

char\* id\_ = new char[std::to\_string(id).size() + 1];

memcpy(id\_, std::to\_string(id).c\_str(), std::to\_string(id).size() + 1);

char\* args[] = {"./child", adr\_, id\_, NULL};

int id2 = fork();

if (id2 == -1) {

std::cout << "Unable to create first worker node" << std::endl;

id = 0;

exit(1);

} else if(id2 == 0){

execv("./child", args);

} else {

child\_id = id;

}

zmq::message\_t message;

main\_socket.recv(&message);

std::string recieved\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << recieved\_message << std::endl;

delete [] adr\_;

delete [] id\_;

} else {

int id;

std::cin >> id;

std::string message\_string = command + " " + std::to\_string(id);

zmq::message\_t message(message\_string.size());

memcpy(message.data(), message\_string.c\_str(), message\_string.size());

main\_socket.send(message);

// catch message from new node

main\_socket.recv(&message);

std::string recieved\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << recieved\_message << std::endl;

}

} else if (command == "exec") {

int id;

std::string big, small;

std::cin >> id;

std::cin >> big >> small;

std::string message\_string = command + " " + std::to\_string(id) + " " + big + " " + small;

zmq::message\_t message(message\_string.size());

memcpy(message.data(), message\_string.c\_str(), message\_string.size());

main\_socket.send(message);

// return value from map

main\_socket.recv(&message);

std::string recieved\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << recieved\_message << std::endl;

} else if (command == "ping") {

int id;

std::cin >> id;

std::string message\_string = command + " " + std::to\_string(id);

zmq::message\_t message(message\_string.size());

memcpy(message.data(), message\_string.c\_str(), message\_string.size());

main\_socket.send(message);

// receive answer from child

main\_socket.recv(&message);

std::string recieved\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << recieved\_message << std::endl;

} else if(command == "kill"){

int id;

std::cin >> id;

if(child\_id == 0){

std::cout << "Error: there isn't nodes" << std::endl;

} else if(child\_id == id){

std::string kill\_message = command + " " + std::to\_string(id);

zmq::message\_t message(kill\_message.size());

memcpy(message.data(), kill\_message.c\_str(), kill\_message.size());

main\_socket.send(message);

main\_socket.recv(&message);

std::string received\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << received\_message << std::endl;

std::cout << "Tree deleted successfully" << std::endl;

return 0;

} else {

std::string kill\_message = command + " " + std::to\_string(id);

zmq::message\_t message(kill\_message.size());

memcpy(message.data(), kill\_message.c\_str(), kill\_message.size());

main\_socket.send(message);

main\_socket.recv(&message);

std::string received\_message(static\_cast<char\*>(message.data()), message.size());

std::cout << received\_message << std::endl;

}

} else if(command == "exit"){

if(child\_id){

std::string kill\_message = "DIE";

zmq::message\_t message(kill\_message.size());

memcpy(message.data(), kill\_message.c\_str(), kill\_message.size());

main\_socket.send(message);

std::cout << "Tree was deleted" << std::endl;

}

main\_socket.close();

context.close();

break;

} else {

std::cout << "Error: incorrect command**\n**";

}

}

}

**child.cpp**

\#include "zmq.hpp"

#include <sstream>

#include <string>

#include <iostream>

#include <zconf.h>

#include <vector>

#include <signal.h>

#include <fstream>

#include <algorithm>

#include <map>

// g++ child.cpp -lzmq -o child -w

void send\_message(std::string message\_string, zmq::socket\_t& socket){

zmq::message\_t message\_back(message\_string.size());

memcpy(message\_back.data(), message\_string.c\_str(), message\_string.size());

if(!socket.send(message\_back)){

std::cout << "Error: can't send message from node with pid " << getpid() << std::endl;

}

}

int main(int argc, char \* argv[]){

std::string adr = argv[1];

zmq::context\_t context(1);

zmq::socket\_t main\_socket(context, ZMQ\_REQ);

main\_socket.connect(argv[1]);

send\_message("OK: " + std::to\_string(getpid()), main\_socket);

int id = std::stoi(argv[2]); // id of this node

std::map<std::string, int> m;

int left\_id = 0;

int right\_id = 0;

zmq::context\_t context\_l(1);

zmq::context\_t context\_r(1);

zmq::socket\_t left\_socket(context\_l, ZMQ\_REP);

std::string adr\_left = "tcp://127.0.0.1:300";

zmq::socket\_t right\_socket(context\_r, ZMQ\_REP);

std::string adr\_right = "tcp://127.0.0.1:300";

while(1){

zmq::message\_t message\_main;

main\_socket.recv(&message\_main);

std::string recieved\_message(static\_cast<char\*>(message\_main.data()), message\_main.size());

std::string command;

for(int i = 0; i < recieved\_message.size(); ++i){

if(recieved\_message[i] != ' '){

command += recieved\_message[i];

} else {

break;

}

}

if(command == "exec"){

int id\_proc; // id of node for adding

std::string id\_proc\_;

std::string big, small, for\_return;

int flag = 0;

std::vector<int> answers;

for(int i = 5; i < recieved\_message.size(); ++i){

if(recieved\_message[i] != ' '){

id\_proc\_ += recieved\_message[i];

} else {

break;

}

}

id\_proc = std::stoi(id\_proc\_);

if(id\_proc == id) { // id == proc\_id

for(int i = 6 + id\_proc\_.size(); i < recieved\_message.size(); ++i){

if (recieved\_message[i] == ' ')

++flag;

if ((recieved\_message[i] != ' ') && (flag == 0)) {

big += recieved\_message[i];

} else if ((recieved\_message[i] != ' ') && (flag == 1)){

small += recieved\_message[i];

}

}

if (big.size() >= small.size()) {

int start = 0;

while(big.find(small, start) != -1){

start = big.find(small, start);

answers.push\_back(start);

++start;

}

}

if(answers.size() == 0){

for\_return = "-1";

}else{

for\_return = std::to\_string(answers[0]);

for(int i = 1; i < answers.size();++i) {

for\_return = for\_return + ";" + std::to\_string(answers[i]);

}

}

for\_return = "OK:" + id\_proc\_ + ":" + for\_return;

send\_message(for\_return, main\_socket);

} else {

if(id > id\_proc){

if(left\_id == 0){ // if node not exists

std::string message\_string = "Error:id: Not found";

send\_message("Error:id: Not found", main\_socket);

} else {

zmq::message\_t message(recieved\_message.size());

memcpy(message.data(), recieved\_message.c\_str(), recieved\_message.size());

if(!left\_socket.send(message)){

std::cout << "Error: can't send message to left node from node with pid: " << getpid() << std::endl;

}

// catch and send to parent

if(!left\_socket.recv(&message)){

std::cout << "Error: can't receive message from left node in node with pid: " << getpid() << std::endl;

}

if(!main\_socket.send(message)){

std::cout << "Error: can't send message to main node from node with pid: " << getpid() << std::endl;

}

}

} else {

if(right\_id == 0){ // if node not exists

std::string message\_string = "Error:id: Not found";

zmq::message\_t message(message\_string.size());

memcpy(message.data(), message\_string.c\_str(), message\_string.size());

if(!main\_socket.send(message)){

std::cout << "Error: can't send message to main node from node with pid: " << getpid() << std::endl;

}

} else {

zmq::message\_t message(recieved\_message.size());

memcpy(message.data(), recieved\_message.c\_str(), recieved\_message.size());

if(!right\_socket.send(message)){

std::cout << "Error: can't send message to right node from node with pid: " << getpid() << std::endl;

}

// catch and send to parent

if(!right\_socket.recv(&message)){

std::cout << "Error: can't receive message from left node in node with pid: " << getpid() << std::endl;

}

if(!main\_socket.send(message)){

std::cout << "Error: can't send message to main node from node with pid: " << getpid() << std::endl;

}

}

}

}

} else if(command == "create"){

int id\_proc; // id of node for creating

std::string id\_proc\_;

for(int i = 7; i < recieved\_message.size(); ++i){

if(recieved\_message[i] != ' '){

id\_proc\_ += recieved\_message[i];

} else {

break;

}

}

id\_proc = std::stoi(id\_proc\_);

if(id\_proc == id){

send\_message("Error: Already exists", main\_socket);

} else if(id\_proc > id){

if(right\_id == 0){ // there is not right node

right\_id = id\_proc;

int right\_id\_tmp = right\_id - 1;

right\_socket.bind(adr\_right + std::to\_string(++right\_id\_tmp));

adr\_right += std::to\_string(right\_id\_tmp);

char\* adr\_right\_ = new char[adr\_right.size() + 1];

memcpy(adr\_right\_, adr\_right.c\_str(), adr\_right.size() + 1);

char\* right\_id\_ = new char[std::to\_string(right\_id).size() + 1];

memcpy(right\_id\_, std::to\_string(right\_id).c\_str(), std::to\_string(right\_id).size() + 1);

char\* args[] = {"./child", adr\_right\_, right\_id\_, NULL};

int f = fork();

if(f == 0){

execv("./child", args);

} else if (f == -1){

std::cout << "Error in forking in node with pid: " << getpid() << std::endl;

} else {

// catch message from new node

zmq::message\_t message\_from\_node;

if(!right\_socket.recv(&message\_from\_node)){

std::cout << "Error: can't receive message from right node in node with pid:" << getpid() << std::endl;

}

std::string recieved\_message\_from\_node(static\_cast<char\*>(message\_from\_node.data()), message\_from\_node.size());

// send message to main node

if(!main\_socket.send(message\_from\_node)){

std::cout << "Error: can't send message to main node from node with pid:" << getpid() << std::endl;

}

}

delete [] adr\_right\_;

delete [] right\_id\_;

} else { // send task to right node

send\_message(recieved\_message, right\_socket);

// catch and send to parent

zmq::message\_t message;

if(!right\_socket.recv(&message)){

std::cout << "Error: can't receive message from left node in node with pid: " << getpid() << std::endl;

}

if(!main\_socket.send(message)){

std::cout << "Error: can't send message to main node from node with pid: " << getpid() << std::endl;

}

}

} else {

if(left\_id == 0){ // there is not left node

left\_id = id\_proc;

int left\_id\_tmp = left\_id - 1;

left\_socket.bind(adr\_left + std::to\_string(++left\_id\_tmp));

adr\_left += std::to\_string(left\_id\_tmp);

char\* adr\_left\_ = new char[adr\_left.size() + 1];

memcpy(adr\_left\_, adr\_left.c\_str(), adr\_left.size() + 1);

char\* left\_id\_ = new char[std::to\_string(left\_id).size() + 1];

memcpy(left\_id\_, std::to\_string(left\_id).c\_str(), std::to\_string(left\_id).size() + 1);

char\* args[] = {"./child", adr\_left\_, left\_id\_, NULL};

int f = fork();

if(f == 0){

execv("./child", args);

} else if(f == -1){

std::cout << "Error in forking in node with pid: " << getpid() << std::endl;

} else {

// catch message from new node

zmq::message\_t message\_from\_node;

if(!left\_socket.recv(&message\_from\_node)){

std::cout << "Error: can't receive message from left node in node with pid:" << getpid() << std::endl;

}

std::string recieved\_message\_from\_node(static\_cast<char\*>(message\_from\_node.data()), message\_from\_node.size());

// send message to main node

if(!main\_socket.send(message\_from\_node)){

std::cout << "Error: can't send message to main node from node with pid:" << getpid() << std::endl;

}

}

delete [] adr\_left\_;

delete [] left\_id\_;

} else { // send task to left node

send\_message(recieved\_message, left\_socket);

// catch and send to parent

zmq::message\_t message;

if(!left\_socket.recv(&message)){

std::cout << "Error: can't receive message from left node in node with pid: " << getpid() << std::endl;

}

if(!main\_socket.send(message)){

std::cout << "Error: can't send message to main node from node with pid: " << getpid() << std::endl;

}

}

}

} else if(command == "ping") {

int id\_proc; // id of node for creating

std::string id\_proc\_;

for(int i = 5; i < recieved\_message.size(); ++i){

if(recieved\_message[i] != ' '){

id\_proc\_ += recieved\_message[i];

} else {

break;

}

}

id\_proc = std::stoi(id\_proc\_);

if(id\_proc == id){

send\_message("OK: 1", main\_socket);

} else if(id\_proc < id) {

if(left\_id == 0){

send\_message("OK: 0", main\_socket);

} else {

left\_socket.send(message\_main);

zmq::message\_t answ;

left\_socket.recv(&answ);

main\_socket.send(answ);

}

} else if(id\_proc > id) {

if(right\_id == 0){

send\_message("OK: 0", main\_socket);

} else {

right\_socket.send(message\_main);

zmq::message\_t answ;

right\_socket.recv(&answ);

main\_socket.send(answ);

}

}

} else if(command == "kill") {

int id\_proc; // id of node for killing

std::string id\_proc\_;

for(int i = 5; i < recieved\_message.size(); ++i){

if(recieved\_message[i] != ' '){

id\_proc\_ += recieved\_message[i];

} else {

break;

}

}

id\_proc = std::stoi(id\_proc\_);

if(id\_proc > id){

if(right\_id == 0){

send\_message("Error: there isn`t node with this id", main\_socket);

} else {

if(right\_id == id\_proc){

send\_message("Ok: " + std::to\_string(right\_id), main\_socket);

send\_message("DIE", right\_socket);

right\_socket.unbind(adr\_right);

adr\_right = "tcp://127.0.0.1:300";

right\_id = 0;

} else {

right\_socket.send(message\_main);

zmq::message\_t message;

right\_socket.recv(&message);

main\_socket.send(message);

}

}

} else if(id\_proc < id){

if(left\_id == 0){

send\_message("Error: there isn`t node with this id", main\_socket);

} else {

if(left\_id == id\_proc){

send\_message("Ok: " + std::to\_string(left\_id), main\_socket);

send\_message("DIE", left\_socket);

left\_socket.unbind(adr\_left);

adr\_left = "tcp://127.0.0.1:300";

left\_id = 0;

} else {

left\_socket.send(message\_main);

zmq::message\_t message;

left\_socket.recv(&message);

main\_socket.send(message);

}

}

}

} else if (command == "DIE") {

if (left\_id){

send\_message("DIE", left\_socket);

left\_socket.unbind(adr\_left);

adr\_left = "tcp://127.0.0.1:300";

left\_id = 0;

}

if (right\_id){

send\_message("DIE", right\_socket);

right\_socket.unbind(adr\_right);

adr\_right = "tcp://127.0.0.1:300";

right\_id = 0;

}

main\_socket.unbind(adr);

return 0;

}

}

}

## **Использование утилиты strace**

ann@ann:~/os/lab6$ strace ./main

execve("./main", ["./main"], 0x7ffdeaa21980 /\* 63 vars \*/) = 0

brk(NULL) = 0x556b105e4000

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=80719, ...}) = 0

mmap(NULL, 80719, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fd9d73f5000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libzmq.so.5", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P?\1\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=630464, ...}) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d73f3000

mmap(NULL, 2725560, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d6f46000

mprotect(0x7fd9d6fd9000, 2097152, PROT\_NONE) = 0

mmap(0x7fd9d71d9000, 28672, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x93000) = 0x7fd9d71d9000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libstdc++.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\304\10\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=1594864, ...}) = 0

mmap(NULL, 3702848, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d6bbd000

mprotect(0x7fd9d6d36000, 2097152, PROT\_NONE) = 0

mmap(0x7fd9d6f36000, 49152, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x179000) = 0x7fd9d6f36000

mmap(0x7fd9d6f42000, 12352, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d6f42000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libgcc\_s.so.1", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\300\*\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=96616, ...}) = 0

mmap(NULL, 2192432, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d69a5000

mprotect(0x7fd9d69bc000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d6bbb000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x16000) = 0x7fd9d6bbb000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\20\35\2\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2030928, ...}) = 0

mmap(NULL, 4131552, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d65b4000

mprotect(0x7fd9d679b000, 2097152, PROT\_NONE) = 0

mmap(0x7fd9d699b000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1e7000) = 0x7fd9d699b000

mmap(0x7fd9d69a1000, 15072, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d69a1000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libsodium.so.23", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\340\251\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=330440, ...}) = 0

mmap(NULL, 2425864, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d6363000

mprotect(0x7fd9d63b3000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d65b2000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x4f000) = 0x7fd9d65b2000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libpgm-5.2.so.0", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000;\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=293784, ...}) = 0

mmap(NULL, 2406448, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d6117000

mprotect(0x7fd9d615e000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d635d000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x46000) = 0x7fd9d635d000

mmap(0x7fd9d635f000, 14384, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d635f000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libnorm.so.1", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000\374\1\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=522248, ...}) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d73f1000

mmap(NULL, 3340624, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d5de7000

mprotect(0x7fd9d5e64000, 2097152, PROT\_NONE) = 0

mmap(0x7fd9d6064000, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x7d000) = 0x7fd9d6064000

mmap(0x7fd9d6067000, 719184, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d6067000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/librt.so.1", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\"\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=31680, ...}) = 0

mmap(NULL, 2128864, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d5bdf000

mprotect(0x7fd9d5be6000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d5de5000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x6000) = 0x7fd9d5de5000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libpthread.so.0", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000b\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=144976, ...}) = 0

mmap(NULL, 2221184, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d59c0000

mprotect(0x7fd9d59da000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d5bd9000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19000) = 0x7fd9d5bd9000

mmap(0x7fd9d5bdb000, 13440, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d5bdb000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libm.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\200\272\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=1700792, ...}) = 0

mmap(NULL, 3789144, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d5622000

mprotect(0x7fd9d57bf000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d59be000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x19c000) = 0x7fd9d59be000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d73ef000

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d73ec000

arch\_prctl(ARCH\_SET\_FS, 0x7fd9d73ecb80) = 0

mprotect(0x7fd9d699b000, 16384, PROT\_READ) = 0

mprotect(0x7fd9d59be000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d5bd9000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d5de5000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d6bbb000, 4096, PROT\_READ) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d73ea000

mprotect(0x7fd9d6f36000, 40960, PROT\_READ) = 0

mprotect(0x7fd9d6064000, 8192, PROT\_READ) = 0

mprotect(0x7fd9d635d000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d65b2000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d71d9000, 24576, PROT\_READ) = 0

mprotect(0x556b0fcf2000, 4096, PROT\_READ) = 0

mprotect(0x7fd9d7409000, 4096, PROT\_READ) = 0

munmap(0x7fd9d73f5000, 80719) = 0

set\_tid\_address(0x7fd9d73ece50) = 8955

set\_robust\_list(0x7fd9d73ece60, 24) = 0

rt\_sigaction(SIGRTMIN, {sa\_handler=0x7fd9d59c5cb0, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_SIGINFO, sa\_restorer=0x7fd9d59d2980}, NULL, 8) = 0

rt\_sigaction(SIGRT\_1, {sa\_handler=0x7fd9d59c5d50, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7fd9d59d2980}, NULL, 8) = 0

rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

brk(NULL) = 0x556b105e4000

brk(0x556b10605000) = 0x556b10605000

futex(0x7fd9d6f4309c, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

futex(0x7fd9d6f430a8, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

openat(AT\_FDCWD, "/sys/devices/system/cpu/online", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "0-3\n", 8192) = 4

close(3) = 0

openat(AT\_FDCWD, "/sys/devices/system/cpu", O\_RDONLY|O\_NONBLOCK|O\_CLOEXEC|O\_DIRECTORY) = 3

fstat(3, {st\_mode=S\_IFDIR|0755, st\_size=0, ...}) = 0

getdents(3, /\* 22 entries \*/, 32768) = 656

getdents(3, /\* 0 entries \*/, 32768) = 0

close(3) = 0

getpid() = 8955

sched\_getaffinity(8955, 128, [0, 1, 2, 3]) = 8

openat(AT\_FDCWD, "/etc/nsswitch.conf", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=556, ...}) = 0

read(3, "# /etc/nsswitch.conf\n#\n# Example"..., 4096) = 556

read(3, "", 4096) = 0

close(3) = 0

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=80719, ...}) = 0

mmap(NULL, 80719, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fd9d73f5000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/tls/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/tls/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/tls/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/tls/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/tls/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/tls/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/tls/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/tls", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64-linux-gnu", {st\_mode=S\_IFDIR|0755, st\_size=16384, ...}) = 0

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/tls/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/tls/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/tls/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/tls/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/tls/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/tls/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/tls/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/tls", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64-linux-gnu/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64-linux-gnu", {st\_mode=S\_IFDIR|0755, st\_size=69632, ...}) = 0

openat(AT\_FDCWD, "/lib/tls/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/tls/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/tls/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/tls/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/tls/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/tls/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/tls/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/tls", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/lib", {st\_mode=S\_IFDIR|0755, st\_size=4096, ...}) = 0

openat(AT\_FDCWD, "/usr/lib/tls/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/tls/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/tls/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/tls/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/tls/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/tls/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/tls/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/tls", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/haswell/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/haswell/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/haswell/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/haswell", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/x86\_64/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib/x86\_64", 0x7ffcf5f73d40) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/usr/lib/libnss\_db.so.2", O\_RDONLY|O\_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/usr/lib", {st\_mode=S\_IFDIR|0755, st\_size=4096, ...}) = 0

munmap(0x7fd9d73f5000, 80719) = 0

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=80719, ...}) = 0

mmap(NULL, 80719, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fd9d73f5000

close(3) = 0

access("/etc/ld.so.nohwcap", F\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libnss\_files.so.2", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P#\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=47568, ...}) = 0

mmap(NULL, 2168632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fd9d5410000

mprotect(0x7fd9d541b000, 2093056, PROT\_NONE) = 0

mmap(0x7fd9d561a000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xa000) = 0x7fd9d561a000

mmap(0x7fd9d561c000, 22328, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fd9d561c000

close(3) = 0

mprotect(0x7fd9d561a000, 4096, PROT\_READ) = 0

munmap(0x7fd9d73f5000, 80719) = 0

openat(AT\_FDCWD, "/etc/protocols", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=2932, ...}) = 0

read(3, "# Internet (IP) protocols\n#\n# Up"..., 4096) = 2932

read(3, "", 4096) = 0

close(3) = 0

eventfd2(0, EFD\_CLOEXEC) = 3

fcntl(3, F\_GETFL) = 0x2 (flags O\_RDWR)

fcntl(3, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

fcntl(3, F\_GETFL) = 0x802 (flags O\_RDWR|O\_NONBLOCK)

fcntl(3, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

getrandom("\x70\x88\x38\x74\x18\x25\xd0\x04\xe8\x53\xed\x35\xed\x36\x6f\x82", 16, 0) = 16

getrandom("\xbc\x84\xbc\x70\x2f\x6b\xb0\x65\x3b\x03\x79\x5e\x6d\x5e\xf4\x30", 16, 0) = 16

eventfd2(0, EFD\_CLOEXEC) = 4

fcntl(4, F\_GETFL) = 0x2 (flags O\_RDWR)

fcntl(4, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

fcntl(4, F\_GETFL) = 0x802 (flags O\_RDWR|O\_NONBLOCK)

fcntl(4, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

epoll\_create1(EPOLL\_CLOEXEC) = 5

epoll\_ctl(5, EPOLL\_CTL\_ADD, 4, {0, {u32=274688032, u64=93918324549664}}) = 0

epoll\_ctl(5, EPOLL\_CTL\_MOD, 4, {EPOLLIN, {u32=274688032, u64=93918324549664}}) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7fd9d4c0f000

mprotect(0x7fd9d4c10000, 8388608, PROT\_READ|PROT\_WRITE) = 0

clone(child\_stack=0x7fd9d540eb70, flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, parent\_tidptr=0x7fd9d540f9d0, tls=0x7fd9d540f700, child\_tidptr=0x7fd9d540f9d0) = 8956

openat(AT\_FDCWD, "/proc/self/task/8956/comm", O\_RDWR) = 6

write(6, "ZMQbg/0", 7) = 7

close(6) = 0

eventfd2(0, EFD\_CLOEXEC) = 6

fcntl(6, F\_GETFL) = 0x2 (flags O\_RDWR)

fcntl(6, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

fcntl(6, F\_GETFL) = 0x802 (flags O\_RDWR|O\_NONBLOCK)

fcntl(6, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

epoll\_create1(EPOLL\_CLOEXEC) = 7

epoll\_ctl(7, EPOLL\_CTL\_ADD, 6, {0, {u32=274703472, u64=93918324565104}}) = 0

epoll\_ctl(7, EPOLL\_CTL\_MOD, 6, {EPOLLIN, {u32=274703472, u64=93918324565104}}) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7fd9d440e000

mprotect(0x7fd9d440f000, 8388608, PROT\_READ|PROT\_WRITE) = 0

clone(child\_stack=0x7fd9d4c0db70, flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, parent\_tidptr=0x7fd9d4c0e9d0, tls=0x7fd9d4c0e700, child\_tidptr=0x7fd9d4c0e9d0) = 8957

openat(AT\_FDCWD, "/proc/self/task/8957/comm", O\_RDWR) = 8

write(8, "ZMQbg/1", 7) = 7

close(8) = 0

eventfd2(0, EFD\_CLOEXEC) = 8

fcntl(8, F\_GETFL) = 0x2 (flags O\_RDWR)

fcntl(8, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

fcntl(8, F\_GETFL) = 0x802 (flags O\_RDWR|O\_NONBLOCK)

fcntl(8, F\_SETFL, O\_RDWR|O\_NONBLOCK) = 0

fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(136, 1), ...}) = 0

write(1, "command:", 8command:) = 8

fstat(0, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(136, 1), ...}) = 0

read(0, create 5

"create 5\n", 1024) = 9

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

socket(AF\_NETLINK, SOCK\_RAW|SOCK\_CLOEXEC, NETLINK\_ROUTE) = 9

bind(9, {sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, 12) = 0

getsockname(9, {sa\_family=AF\_NETLINK, nl\_pid=8955, nl\_groups=00000000}, [12]) = 0

sendto(9, {{len=20, type=RTM\_GETLINK, flags=NLM\_F\_REQUEST|NLM\_F\_DUMP, seq=1640791805, pid=0}, {ifi\_family=AF\_UNSPEC, ...}}, 20, 0, {sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, 12) = 20

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base=[{{len=1296, type=RTM\_NEWLINK, flags=NLM\_F\_MULTI, seq=1640791805, pid=8955}, {ifi\_family=AF\_UNSPEC, ifi\_type=ARPHRD\_LOOPBACK, ifi\_index=if\_nametoindex("lo"), ifi\_flags=IFF\_UP|IFF\_LOOPBACK|IFF\_RUNNING|0x10000, ifi\_change=0}, [{{nla\_len=7, nla\_type=IFLA\_IFNAME}, "lo"}, {{nla\_len=8, nla\_type=IFLA\_TXQLEN}, 1000}, {{nla\_len=5, nla\_type=IFLA\_OPERSTATE}, 0}, {{nla\_len=5, nla\_type=IFLA\_LINKMODE}, 0}, {{nla\_len=8, nla\_type=IFLA\_MTU}, 65536}, {{nla\_len=8, nla\_type=IFLA\_GROUP}, 0}, {{nla\_len=8, nla\_type=IFLA\_PROMISCUITY}, 0}, {{nla\_len=8, nla\_type=IFLA\_NUM\_TX\_QUEUES}, 1}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SEGS}, 65535}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SIZE}, 65536}, {{nla\_len=8, nla\_type=IFLA\_NUM\_RX\_QUEUES}, 1}, {{nla\_len=5, nla\_type=IFLA\_CARRIER}, 1}, {{nla\_len=12, nla\_type=IFLA\_QDISC}, "noqueue"}, {{nla\_len=8, nla\_type=IFLA\_CARRIER\_CHANGES}, 0}, {{nla\_len=5, nla\_type=IFLA\_PROTO\_DOWN}, 0}, {{nla\_len=8, nla\_type=0x2f /\* IFLA\_??? \*/}, "\x00\x00\x00\x00"}, {{nla\_len=8, nla\_type=0x30 /\* IFLA\_??? \*/}, "\x00\x00\x00\x00"}, {{nla\_len=36, nla\_type=IFLA\_MAP}, {mem\_start=0, mem\_end=0, base\_addr=0, irq=0, dma=0, port=0}}, {{nla\_len=10, nla\_type=IFLA\_ADDRESS}, "\x00\x00\x00\x00\x00\x00"}, {{nla\_len=10, nla\_type=IFLA\_BROADCAST}, "\x00\x00\x00\x00\x00\x00"}, {{nla\_len=196, nla\_type=IFLA\_STATS64}, {rx\_packets=13785, tx\_packets=13785, rx\_bytes=831821, tx\_bytes=831821, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=100, nla\_type=IFLA\_STATS}, {rx\_packets=13785, tx\_packets=13785, rx\_bytes=831821, tx\_bytes=831821, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=12, nla\_type=IFLA\_XDP}, {{nla\_len=5, nla\_type=IFLA\_XDP\_ATTACHED}, 0}}, {{nla\_len=756, nla\_type=IFLA\_AF\_SPEC}, "\x84\x00\x02\x00\x80\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00"...}]}, {{len=1304, type=RTM\_NEWLINK, flags=NLM\_F\_MULTI, seq=1640791805, pid=8955}, {ifi\_family=AF\_UNSPEC, ifi\_type=ARPHRD\_ETHER, ifi\_index=if\_nametoindex("eno1"), ifi\_flags=IFF\_UP|IFF\_BROADCAST|IFF\_MULTICAST, ifi\_change=0}, [{{nla\_len=9, nla\_type=IFLA\_IFNAME}, "eno1"}, {{nla\_len=8, nla\_type=IFLA\_TXQLEN}, 1000}, {{nla\_len=5, nla\_type=IFLA\_OPERSTATE}, 2}, {{nla\_len=5, nla\_type=IFLA\_LINKMODE}, 0}, {{nla\_len=8, nla\_type=IFLA\_MTU}, 1500}, {{nla\_len=8, nla\_type=IFLA\_GROUP}, 0}, {{nla\_len=8, nla\_type=IFLA\_PROMISCUITY}, 0}, {{nla\_len=8, nla\_type=IFLA\_NUM\_TX\_QUEUES}, 1}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SEGS}, 65535}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SIZE}, 65536}, {{nla\_len=8, nla\_type=IFLA\_NUM\_RX\_QUEUES}, 1}, {{nla\_len=5, nla\_type=IFLA\_CARRIER}, 0}, {{nla\_len=13, nla\_type=IFLA\_QDISC}, "fq\_codel"}, {{nla\_len=8, nla\_type=IFLA\_CARRIER\_CHANGES}, 1}, {{nla\_len=5, nla\_type=IFLA\_PROTO\_DOWN}, 0}, {{nla\_len=8, nla\_type=0x2f /\* IFLA\_??? \*/}, "\x00\x00\x00\x00"}, {{nla\_len=8, nla\_type=0x30 /\* IFLA\_??? \*/}, "\x01\x00\x00\x00"}, {{nla\_len=36, nla\_type=IFLA\_MAP}, {mem\_start=0, mem\_end=0, base\_addr=0, irq=0, dma=0, port=0}}, {{nla\_len=10, nla\_type=IFLA\_ADDRESS}, "\xb0\x0c\xd1\xea\x46\x8f"}, {{nla\_len=10, nla\_type=IFLA\_BROADCAST}, "\xff\xff\xff\xff\xff\xff"}, {{nla\_len=196, nla\_type=IFLA\_STATS64}, {rx\_packets=0, tx\_packets=0, rx\_bytes=0, tx\_bytes=0, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=100, nla\_type=IFLA\_STATS}, {rx\_packets=0, tx\_packets=0, rx\_bytes=0, tx\_bytes=0, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=12, nla\_type=IFLA\_XDP}, {{nla\_len=5, nla\_type=IFLA\_XDP\_ATTACHED}, 0}}, {{nla\_len=756, nla\_type=IFLA\_AF\_SPEC}, "\x84\x00\x02\x00\x80\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00"...}]}], iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 2600

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base={{len=1296, type=RTM\_NEWLINK, flags=NLM\_F\_MULTI, seq=1640791805, pid=8955}, {ifi\_family=AF\_UNSPEC, ifi\_type=ARPHRD\_ETHER, ifi\_index=if\_nametoindex("wlo1"), ifi\_flags=IFF\_UP|IFF\_BROADCAST|IFF\_RUNNING|IFF\_MULTICAST|0x10000, ifi\_change=0}, [{{nla\_len=9, nla\_type=IFLA\_IFNAME}, "wlo1"}, {{nla\_len=8, nla\_type=IFLA\_TXQLEN}, 1000}, {{nla\_len=5, nla\_type=IFLA\_OPERSTATE}, 6}, {{nla\_len=5, nla\_type=IFLA\_LINKMODE}, 1}, {{nla\_len=8, nla\_type=IFLA\_MTU}, 1500}, {{nla\_len=8, nla\_type=IFLA\_GROUP}, 0}, {{nla\_len=8, nla\_type=IFLA\_PROMISCUITY}, 0}, {{nla\_len=8, nla\_type=IFLA\_NUM\_TX\_QUEUES}, 4}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SEGS}, 65535}, {{nla\_len=8, nla\_type=IFLA\_GSO\_MAX\_SIZE}, 65536}, {{nla\_len=8, nla\_type=IFLA\_NUM\_RX\_QUEUES}, 1}, {{nla\_len=5, nla\_type=IFLA\_CARRIER}, 1}, {{nla\_len=7, nla\_type=IFLA\_QDISC}, "mq"}, {{nla\_len=8, nla\_type=IFLA\_CARRIER\_CHANGES}, 2}, {{nla\_len=5, nla\_type=IFLA\_PROTO\_DOWN}, 0}, {{nla\_len=8, nla\_type=0x2f /\* IFLA\_??? \*/}, "\x01\x00\x00\x00"}, {{nla\_len=8, nla\_type=0x30 /\* IFLA\_??? \*/}, "\x01\x00\x00\x00"}, {{nla\_len=36, nla\_type=IFLA\_MAP}, {mem\_start=0, mem\_end=0, base\_addr=0, irq=0, dma=0, port=0}}, {{nla\_len=10, nla\_type=IFLA\_ADDRESS}, "\xd4\x3b\x04\x3a\xa7\x82"}, {{nla\_len=10, nla\_type=IFLA\_BROADCAST}, "\xff\xff\xff\xff\xff\xff"}, {{nla\_len=196, nla\_type=IFLA\_STATS64}, {rx\_packets=30530, tx\_packets=19500, rx\_bytes=25906404, tx\_bytes=4949527, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=100, nla\_type=IFLA\_STATS}, {rx\_packets=30530, tx\_packets=19500, rx\_bytes=25906404, tx\_bytes=4949527, rx\_errors=0, tx\_errors=0, rx\_dropped=0, tx\_dropped=0, multicast=0, collisions=0, rx\_length\_errors=0, rx\_over\_errors=0, rx\_crc\_errors=0, rx\_frame\_errors=0, rx\_fifo\_errors=0, rx\_missed\_errors=0, tx\_aborted\_errors=0, tx\_carrier\_errors=0, tx\_fifo\_errors=0, tx\_heartbeat\_errors=0, tx\_window\_errors=0, rx\_compressed=0, tx\_compressed=0, rx\_nohandler=0}}, {{nla\_len=12, nla\_type=IFLA\_XDP}, {{nla\_len=5, nla\_type=IFLA\_XDP\_ATTACHED}, 0}}, {{nla\_len=756, nla\_type=IFLA\_AF\_SPEC}, "\x84\x00\x02\x00\x80\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00\x01\x00\x00\x00"...}]}, iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 1296

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base={{len=20, type=NLMSG\_DONE, flags=NLM\_F\_MULTI, seq=1640791805, pid=8955}, 0}, iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 20

sendto(9, {{len=20, type=RTM\_GETADDR, flags=NLM\_F\_REQUEST|NLM\_F\_DUMP, seq=1640791806, pid=0}, {ifa\_family=AF\_UNSPEC, ...}}, 20, 0, {sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, 12) = 20

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base=[{{len=76, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET, ifa\_prefixlen=8, ifa\_flags=IFA\_F\_PERMANENT, ifa\_scope=RT\_SCOPE\_HOST, ifa\_index=if\_nametoindex("lo")}, [{{nla\_len=8, nla\_type=IFA\_ADDRESS}, 127.0.0.1}, {{nla\_len=8, nla\_type=IFA\_LOCAL}, 127.0.0.1}, {{nla\_len=7, nla\_type=IFA\_LABEL}, "lo"}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_PERMANENT}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=4294967295, ifa\_valid=4294967295, cstamp=224, tstamp=224}}]}, {{len=88, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET, ifa\_prefixlen=24, ifa\_flags=0, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=8, nla\_type=IFA\_ADDRESS}, 192.168.1.64}, {{nla\_len=8, nla\_type=IFA\_LOCAL}, 192.168.1.64}, {{nla\_len=8, nla\_type=IFA\_BROADCAST}, 192.168.1.255}, {{nla\_len=9, nla\_type=IFA\_LABEL}, "wlo1"}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_NOPREFIXROUTE}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=79771, ifa\_valid=79771, cstamp=1205, tstamp=660738}}]}], iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 164

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base=[{{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=128, ifa\_flags=IFA\_F\_PERMANENT, ifa\_scope=RT\_SCOPE\_HOST, ifa\_index=if\_nametoindex("lo")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, ::1}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=4294967295, ifa\_valid=4294967295, cstamp=224, tstamp=224}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_PERMANENT}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=128, ifa\_flags=0, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, 2a00:1370:8137:6796:1484:6013:4b01:bc7}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=577, ifa\_valid=577, cstamp=1361, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_NOPREFIXROUTE}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=64, ifa\_flags=IFA\_F\_SECONDARY, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, 2a00:1370:8137:6796:1dfd:c02b:dba8:950c}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=311, ifa\_valid=311, cstamp=1202, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_SECONDARY}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=64, ifa\_flags=0, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, 2a00:1370:8137:6796:ad39:a00d:5808:4400}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=311, ifa\_valid=311, cstamp=1202, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_MANAGETEMPADDR|IFA\_F\_NOPREFIXROUTE}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=64, ifa\_flags=IFA\_F\_SECONDARY, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, fd60:e320:1618:1:1dfd:c02b:dba8:950c}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=311, ifa\_valid=311, cstamp=1202, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_SECONDARY}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=64, ifa\_flags=0, ifa\_scope=RT\_SCOPE\_UNIVERSE, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, fd60:e320:1618:1:a2d2:41d3:2efe:aa5e}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=311, ifa\_valid=311, cstamp=1202, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_MANAGETEMPADDR|IFA\_F\_NOPREFIXROUTE}]}, {{len=72, type=RTM\_NEWADDR, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, {ifa\_family=AF\_INET6, ifa\_prefixlen=64, ifa\_flags=IFA\_F\_PERMANENT, ifa\_scope=RT\_SCOPE\_LINK, ifa\_index=if\_nametoindex("wlo1")}, [{{nla\_len=20, nla\_type=IFA\_ADDRESS}, fe80::e44:a50:7435:c1b8}, {{nla\_len=20, nla\_type=IFA\_CACHEINFO}, {ifa\_prefered=4294967295, ifa\_valid=4294967295, cstamp=1000, tstamp=661844}}, {{nla\_len=8, nla\_type=IFA\_FLAGS}, IFA\_F\_PERMANENT|IFA\_F\_NOPREFIXROUTE}]}], iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 504

recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base={{len=20, type=NLMSG\_DONE, flags=NLM\_F\_MULTI, seq=1640791806, pid=8955}, 0}, iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 20

close(9) = 0

socket(AF\_INET, SOCK\_STREAM|SOCK\_CLOEXEC, IPPROTO\_TCP) = 9

setsockopt(9, SOL\_SOCKET, SO\_REUSEADDR, [1], 4) = 0

bind(9, {sa\_family=AF\_INET, sin\_port=htons(3005), sin\_addr=inet\_addr("127.0.0.1")}, 16) = 0

listen(9, 100) = 0

getsockname(9, {sa\_family=AF\_INET, sin\_port=htons(3005), sin\_addr=inet\_addr("127.0.0.1")}, [128->16]) = 0

write(6, "\1\0\0\0\0\0\0\0", 8) = 8

write(8, "\1\0\0\0\0\0\0\0", 8) = 8

clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLD, child\_tidptr=0x7fd9d73ece50) = 8998

futex(0x7fd9d69a1918, FUTEX\_WAKE\_PRIVATE, 1) = 1

poll([{fd=8, events=POLLIN}], 1, -1) = 1 ([{fd=8, revents=POLLIN}])

read(8, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

poll([{fd=8, events=POLLIN}], 1, -1) = 1 ([{fd=8, revents=POLLIN}])

read(8, "\1\0\0\0\0\0\0\0", 8) = 8

write(6, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

write(6, "\1\0\0\0\0\0\0\0", 8) = 8

write(1, "OK: 8998\n", 9OK: 8998

) = 9

write(1, "command:", 8command:) = 8

read(0, create 9

"create 9\n", 1024) = 9

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

write(6, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, -1) = 1 ([{fd=8, revents=POLLIN}])

read(8, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

poll([{fd=8, events=POLLIN}], 1, -1) = 1 ([{fd=8, revents=POLLIN}])

read(8, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

write(1, "OK: 9049\n", 9OK: 9049

) = 9

write(1, "command:", 8command:) = 8

read(0, exec 9

"exec 9\n", 1024) = 7

read(0, qwertyhjklkjhgfd

"qwertyhjklkjhgfd\n", 1024) = 17

read(0, j

"j\n", 1024) = 2

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

write(6, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, -1) = 1 ([{fd=8, revents=POLLIN}])

read(8, "\1\0\0\0\0\0\0\0", 8) = 8

poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

write(1, "OK:9:7;11\n", 10OK:9:7;11

) = 10

write(1, "command:", 8command:) = 8

## **Демонстрация работы программы**

ann@ann:~/os/lab6$ ./main

command:create 686

OK: 6035

command:create 54

OK: 6063

command:ping 2

OK: 0

command:create 8

OK: 6184

command:create 4585

OK: 6192

command:create 455

OK: 6253

command:ping 54

OK: 1

command:kill 54

Ok: 54

command:ping 54

OK: 0

command:ping 2

OK: 0

command:ping 8

OK: 0

command:create 45

OK: 6562

command:exec 45

aaadorayy

dora

OK:45:3

command:exec 45

aaaaaaaaaaaaaaaaaaaaaaaaa

aa

OK:45:0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23

command:exec 45

yhtjrktrjkhjfk

b

OK:45:-1

command:exec 45

yt

hgjfkd

OK:45:-1

command:exit

Tree was deleted

## **Вывод**

Лабораторная работа была сложной (самая сложная лабораторная работа за все время обучения в МАИ), но очень интересной. Ведь в ней сразу можно применить знания, полученные в ходе выполнения предыдущих лабораторных работ, так как здесь и многопоточность, и межпроцессорное взаимодействие, основанное на очередях сообщений, и синхронизация потоков. А помимо всего этого также необходимо разобраться с дополнительной библиотекой (zmq), которую пришлось дополнительно устанавливать. Единственное, что мне не очень понравилось – наличие небольшого количества документации по этой библиотеки и более сложная отладка программ. Заодно я вспомнила функции работы со строками, а именно поиск подстроки.