Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторные работы №6-8 по курсу

«Операционные системы»

Управление серверами сообщений, применение отложенных вычислений, интеграция программных систем друг с другом.

Студент: Моисеенков Илья Павлович
Группа: М80 – 208Б-19
Вариант: 4
Преподаватель: Миронов Евгений Сергеевич
Дата: 11.01.2021
Оценка: хорошо
Подпись: ___________

Москва, 2021

1. Постановка задачи

Реализовать распределенную систему по асинхронной обработке запросов. В данной распределенной системе должно существовать 2 вида узлов: управляющий и вычислительный. Необходимо объединить данные узлы в соответствии с топологией «список списков». Связь между узлами необходимо осуществить при помощи технологии очередей сообщений. В данной системе необходимо предусмотреть проверку доступности узлов. При убийстве любого вычислительного узла система должна пытаться максимально сохранять свою работоспособность, а именно все дочерние узлы убитого узла могут стать недоступными, но родительские узлы должны сохранить свою работоспособность.

Управляющий узел отвечает за ввод команд от пользователя и отправку этих команд на вычислительные узлы.

2. Общие сведения о программе

Программа написана на языке C++ на операционной системе Ubuntu. В программе используется очередь сообщений ZeroMQ.

Программа поддерживает следующие команды:

- create [id] [parent_id] – создать новый узел [id], родителем которого является узел [parent_id]. Если [parent_id] = -1, то родительский узел – управляющий.
- kill [id] – удалить узел [id]. Все дочерние узлы будут также удалены.
- exec [id] add [key] [value] – добавить переменную [key] со значением [value] в словарь узла [id]
- exec [id] check [key] – запросить значение переменной [key] в словаре узла [id].
- pingall – проверить доступность узлов. Будет выведен список всех доступных на данный момент узлов.
- exit – выйти из программы.

3. Общий метод и алгоритм решения

В программе используется тип соединения Request-Response. Узлы передают информацию друг другу при помощи очереди сообщений. Все сообщения имеют следующий вид:

[id узла, которому предназначено сообщение] [команда] [аргументы]

Управляющий узел хранит структуру «список списков», в которую записывает id существующих узлов. При помощи этой структуры он определяет, в какой список нужно направить сообщение.

Вычислительный узел, получив сообщение, сравнивает свой id и id из сообщения. Если они совпадают, то узел начинает обрабатывать запрос, в противном случае узел направляет это же сообщение своему ребенку и ждет от него ответа.

Для удобства функции отправки и получения сообщений, а также функции для подключения к сокетам вынесены в отдельный заголовочный файл, который подключается к программам узлов.

Для хранения локального словаря используется контейнер std::unordered_map. Для проверки доступности узлов используется контейнер std::set. Управляющий узел отправляет запрос всем спискам узлов и получает в ответ строку с id всех доступных узлов списка. Все id добавляются в set, а потом выводятся на экран.

4. Основные файлы программы

topology.h

```
#include <list>
#include <stdexcept>

class topology {
private:
    std::list<std::list<int>> container;

public:
    void insert(int id, int parent_id) {
        if (parent_id == -1) {
            std::list<int> new_list;
            new_list.push_back(id);
            container.push_back(new_list);
        }
        else {
            int list_id = find(parent_id);
            if (list_id == -1) {
                throw std::runtime_error("Wrong parent id");
            }
            auto it1 = container.begin();
            std::advance(it1, list_id);
            for (auto it2 = it1->begin(); it2 != it1->end(); ++it2) {
                if (*it2 == parent_id) {
                    it1->insert(++it2, id);
                    return;
                }
            }
        }
    }

    int find(int id) {
        int cur_list_id = 0;
        for (auto it1 = container.begin(); it1 != container.end(); ++it1) {
            for (auto it2 = it1->begin(); it2 != it1->end(); ++it2) {
                if (*it2 == id) {
                    return cur_list_id;
                }
            }
            ++cur_list_id;
        }
        return -1;
    }

    void erase(int id) {
        int list_id = find(id);
        if (list_id == -1) {
            throw std::runtime_error("Wrong id");
        }
        auto it1 = container.begin();
        std::advance(it1, list_id);
        for (auto it2 = it1->begin(); it2 != it1->end(); ++it2) {
            if (*it2 == id) {
                it1->erase(it2, it1->end());
                if (it1->empty()) {
                    container.erase(it1);
                }
                return;
            }
        }
    }

    int get_first_id(int list_id) {
        auto it1 = container.begin();
        std::advance(it1, list_id);
        if (it1->begin() == it1->end()) {
            return -1;
        }
        return *(it1->begin());
    }
};
```

zmq_functions.h

```
#include <zmq.hpp>
#include <iostream>

const int MAIN_PORT = 4040;

void send_message(zmq::socket_t& socket, const std::string& msg) {
    //std::cout << "Start sending " << msg << std::endl;
    zmq::message_t message(msg.size());
    memcpy(message.data(), msg.c_str(), msg.size());
    socket.send(message);
    //std::cout << "Finished sending " << msg << std::endl;
}

std::string receive_message(zmq::socket_t& socket) {
    //std::cout << "Start receiving" << std::endl;
    zmq::message_t message;
    int chars_read;
    try {
        chars_read = (int)socket.recv(&message);
    }
    catch (...) {
        chars_read = 0;
    }
    if (chars_read == 0) {
        return "Error: node is unavailable [zmq_func]";
    }
    std::string received_msg(static_cast<char*>(message.data()), message.size
());
    //std::cout << "Received " << received_msg << std::endl;
    return received_msg;
}

void connect(zmq::socket_t& socket, int id) {
    std::string address = "tcp://127.0.0.1:" + std::to_string(MAIN_PORT + id)
;
    socket.connect(address);
    //std::cout << "Socket connected to " << address << std::endl;
}

void disconnect(zmq::socket_t& socket, int id) {
    std::string address = "tcp://127.0.0.1:" + std::to_string(MAIN_PORT + id)
;
    socket.disconnect(address);
    //std::cout << "Socket disconnected from " << address << std::endl;
}

void bind(zmq::socket_t& socket, int id) {
    std::string address = "tcp://127.0.0.1:" + std::to_string(MAIN_PORT + id)
;
    socket.bind(address);
    //std::cout << "Socket binded to " << address << std::endl;
}

void unbind(zmq::socket_t& socket, int id) {
    std::string address = "tcp://127.0.0.1:" + std::to_string(MAIN_PORT + id)
;
    socket.unbind(address);
    //std::cout << "Socket unbinded from " << address << std::endl;
}
```

control.cpp

```
#include <unistd.h>
#include <sstream>
#include <set>

#include "zmq_functions.h"
#include "topology.h"

int main() {
    topology network;
    std::vector<zmq::socket_t> branches;
    zmq::context_t context;

    std::string cmd;
    while (std::cin >> cmd) {

        if (cmd == "create") {
            int node_id, parent_id;
            std::cin >> node_id >> parent_id;

            if (network.find(node_id) != -1) {
                std::cout << "Error: already exists" << std::endl;
            }
            else if (parent_id == -1) {
                pid_t pid = fork();
                if (pid < 0) {
                    perror("Can't create new process");
                    return -1;
                }
                if (pid == 0) {
                    execl("./counting", "./counting", std::to_string(node_id)
.c_str(), NULL);
                    perror("Can't execute new process");
                    return -2;
                }

                branches.emplace_back(context, ZMQ_REQ);
                branches[branches.size() - 1].setsockopt(ZMQ_SNDTIMEO, 5000);
                bind(branches[branches.size() - 1], node_id);
                send_message(branches[branches.size() - 1], std::to_string(no
de_id) + "pid");

                std::string reply = receive_message(branches[branches.size()
- 1]);
```

```
                std::cout << reply << std::endl;
                network.insert(node_id, parent_id);
            }
            else if (network.find(parent_id) == -1) {
                std::cout << "Error: parent not found" << std::endl;
            }
            else {
                int branch = network.find(parent_id);
                send_message(branches[branch], std::to_string(parent_id) + "c
reate " + std::to_string(node_id));

                std::string reply = receive_message(branches[branch]);
                std::cout << reply << std::endl;
                network.insert(node_id, parent_id);
            }
        }
        else if (cmd == "exec") {
            int dest_id;
            std::string exec_cmd, key;
            std::cin >> dest_id >> exec_cmd >> key;
            int branch = network.find(dest_id);
            if (branch == -1) {
                std::cout << "ERROR: incorrect node id" << std::endl;
            }
            else {
                if (exec_cmd == "check") {
                    send_message(branches[branch], std::to_string(dest_id) +
"check " + key);
                }
                else if (exec_cmd == "add") {
                    int value;
                    std::cin >> value;
                    send_message(branches[branch], std::to_string(dest_id) +
"add " + key + " " + std::to_string(value));
                }

                std::string reply = receive_message(branches[branch]);
                std::cout << reply << std::endl;
            }
        }
        else if (cmd == "kill") {
            int id;
            std::cin >> id;
            int branch = network.find(id);
            if (branch == -1) {
                std::cout << "ERROR: incorrect node id" << std::endl;
            }
            else {
                bool is_first = (network.get_first_id(branch) == id);
                send_message(branches[branch], std::to_string(id) + " kill");

                std::string reply = receive_message(branches[branch]);
                std::cout << reply << std::endl;
                network.erase(id);
                if (is_first) {
                    unbind(branches[branch], id);
                    branches.erase(branches.begin() + branch);
                }
            }
        }
        else if (cmd == "pingall") {
            std::set<int> available_nodes;
            for (size_t i = 0; i < branches.size(); ++i) {
                int first_node_id = network.get_first_id(i);
                send_message(branches[i], std::to_string(first_node_id) + " p
ingall");

                std::string received_message = receive_message(branches[i]);
                std::istringstream reply(received_message);
                int node;
                while(reply >> node) {
                    available_nodes.insert(node);
                }
            }
            std::cout << "OK: ";
            if (available_nodes.empty()) {
                std::cout << "no available nodes" << std::endl;
            }
            else {
                for (auto v : available_nodes) {
                    std::cout << v << " ";
                }
                std::cout << std::endl;
            }
        }
        else if (cmd == "exit") {
            for (size_t i = 0; i < branches.size(); ++i) {
                int first_node_id = network.get_first_id(i);
                send_message(branches[i], std::to_string(first_node_id) + " k
ill");

                std::string reply = receive_message(branches[i]);
                if (reply != "OK") {
                    std::cout << reply << std::endl;
                }
                else {
                    unbind(branches[i], first_node_id);
                }
            }
            exit(0);
        }
        else {
            std::cout << "Incorrect cmd" << std::endl;
        }
    }
}
```

counting.cpp

```
#include <unordered_map>
#include <unistd.h>
#include <sstream>
#include <unordered_map>

#include "zmq_functions.h"

int main(int argc, char* argv[]) {
    if (argc != 2 && argc != 3) {
        throw std::runtime_error("Wrong args for counting node");
    }
    int cur_id = std::atoi(argv[1]);
    int child_id = -1;
    if (argc == 3) {
        child_id = std::atoi(argv[2]);
    }

    std::unordered_map<std::string, int> dictionary;

    zmq::context_t context;
    zmq::socket_t parent_socket(context, ZMQ_REP);
    connect(parent_socket, cur_id);

    zmq::socket_t child_socket(context, ZMQ_REQ);
    child_socket.setsockopt(ZMQ_SNDTIMEO, 5000);
    if (child_id != -1) {
        bind(child_socket, child_id);
    }

    std::string message;
    while (true) {
        message = receive_message(parent_socket);
        std::istringstream request(message);
        int dest_id;
        request >> dest_id;

        std::string cmd;
        request >> cmd;

        if (dest_id == cur_id) {

            if (cmd == "pid") {
                send_message(parent_socket, "OK: " + std::to_string(getpid())
);
            }

            else if (cmd == "create") {
                int new_child_id;
                request >> new_child_id;
                if (child_id != -1) {
                    unbind(child_socket, child_id);
                }
                bind(child_socket, new_child_id);
                pid_t pid = fork();
                if (pid < 0) {
                    perror("Can't create new process");
                    return -1;
                }
                if (pid == 0) {
                    execl("./counting", "./counting", std::to_string(new_chil
d_id).c_str(), std::to_string(child_id).c_str(), NULL);
                    perror("Can't execute new process");
                    return -2;
                }
                send_message(child_socket, std::to_string(new_child_id) + "pi
d");
                child_id = new_child_id;
                send_message(parent_socket, receive_message(child_socket));
            }

            else if (cmd == "check") {
                std::string key;
                request >> key;
                if (dictionary.find(key) != dictionary.end()) {
                    send_message(parent_socket, "OK: " + std::to_string(cur_i
d) + ": " + std::to_string(dictionary[key]));
                }
                else {
                    send_message(parent_socket, "OK: " + std::to_string(cur_i
d) + ": '" + key + "' not found");
                }
            }

            else if (cmd == "add") {
                std::string key;
                int value;
                request >> key >> value;
                dictionary[key] = value;
                send_message(parent_socket, "OK: " + std::to_string(cur_id));
            }

            else if (cmd == "pingall") {
                std::string reply;
                if (child_id != -1) {
                    send_message(child_socket, std::to_string(child_id) + " p
ingall");
                    std::string msg = receive_message(child_socket);
                    reply += " " + msg;
                }
                send_message(parent_socket, std::to_string(cur_id) + reply);
            }

            else if (cmd == "kill") {
                if (child_id != -1) {
                    send_message(child_socket, std::to_string(child_id) + " k
ill");
                    std::string msg = receive_message(child_socket);
                    if (msg == "OK") {
                        send_message(parent_socket, "OK");
                    }
                    unbind(child_socket, child_id);
                    disconnect(parent_socket, cur_id);
                    break;
                }
                send_message(parent_socket, "OK");
                disconnect(parent_socket, cur_id);
                break;
            }
        }
        else if (child_id != -1) {
            send_message(child_socket, message);
            send_message(parent_socket, receive_message(child_socket));
            if (child_id == dest_id && cmd == "kill") {
                child_id = -1;
            }
        }
        else {
            send_message(parent_socket, "Error: node is unavailable");
        }
    }
}
```

5. Демонстрация работы программы

```
mosik@LAPTOP-69S778GL:~/os_lab6$ ./control
create 1 -1
OK: 17801
create 100 -1
OK: 17818
create 3 1
OK: 17833
create 4 3
OK: 17842
create 5 4
OK: 17857
create 200 100
OK: 17866
pingall
OK: 1 3 4 5 100 200
exec 3 add var1 10
OK: 3
exec 3 check var1
OK: 3: 10
exec 3 check var2
OK: 3: 'var2' not found
exec 4 check var1
OK: 4: 'var1' not found
create 2 1
OK: 17947
pingall
OK: 1 2 3 4 5 100 200
exec 3 check var1
OK: 3: 10
kill 2
OK
pingall
OK: 1 100 200
kill -1
ERROR: incorrect node id
kill 200
OK
pingall
OK: 1 100
exec 3 check var1
ERROR: incorrect node id
exit

mosik@LAPTOP-69S778GL:~/os_lab6$ strace -o strace_log.txt ./control
create 0 -1
OK: 20242
exit
mosik@LAPTOP-69S778GL:~/os_lab6$ cat strace_log.txt
execve("./control", ["./control"], 0x7fffd7c8fb10 /* 19 vars */) = 0
brk(NULL)                               = 0x7fffdb4a2000
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=43552, ...}) = 0
mmap(NULL, 43552, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f71eb6c4000
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) =
0x7f71eb6c0000
mmap(NULL, 2725560, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f71eb160000
mprotect(0x7f71eb1f3000, 2097152, PROT_NONE) = 0
mmap(0x7f71eb3f3000, 28672, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x93000) = 0x7f71eb3f3000
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) =
0x7f71eadd0000
mprotect(0x7f71eaf49000, 2097152, PROT_NONE) = 0
mmap(0x7f71eb149000, 49152, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x179000) = 0x7f71eb149000
mmap(0x7f71eb155000, 12352, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f71eb155000
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) =
0x7f71eabb0000
mprotect(0x7f71eabc7000, 2093056, PROT_NONE) = 0
mmap(0x7f71eadc6000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x16000) = 0x7f71eadc6000
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) =
0x7f71ea7b0000
mprotect(0x7f71ea997000, 2097152, PROT_NONE) = 0
mmap(0x7f71eab97000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1e7000) = 0x7f71eab97000
mmap(0x7f71eab9d000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f71eab9d000
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) =
0x7f71ea550000
mprotect(0x7f71ea5a0000, 2093056, PROT_NONE) = 0
mmap(0x7f71ea79f000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4f000) = 0x7f71ea79f000
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) =
0x7f71ea300000
mprotect(0x7f71ea347000, 2093056, PROT_NONE) = 0
mmap(0x7f71ea546000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x46000) = 0x7f71ea546000
```

```
mmap(0x7f71ea548000, 14384, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f71ea548000
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) =
0x7f71eb6b0000
mmap(NULL, 3340624, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f71e9fd0000
mprotect(0x7f71ea04d000, 2097152, PROT_NONE) = 0
mmap(0x7f71ea24d000, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x7d000) = 0x7f71ea24d000
mmap(0x7f71ea250000, 719184, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f71ea250000
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) =
0x7f71e9dc0000
mprotect(0x7f71e9dc7000, 2093056, PROT_NONE) = 0
mmap(0x7f71e9fc6000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x6000) = 0x7f71e9fc6000
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) =
0x7f71e9ba0000
mprotect(0x7f71e9bba000, 2093056, PROT_NONE) = 0
mmap(0x7f71e9db9000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x19000) = 0x7f71e9db9000
mmap(0x7f71e9dbb000, 13440, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f71e9dbb000
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) =
0x7f71e9800000
mprotect(0x7f71e999d000, 2093056, PROT_NONE) = 0
mmap(0x7f71e9b9c000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x19c000) = 0x7f71e9b9c000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f71eb6a0000
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f71eb690000
arch_prctl(ARCH_SET_FS, 0x7f71eb690b80) = 0
mprotect(0x7f71eab97000, 16384, PROT_READ) = 0
mprotect(0x7f71e9b9c000, 4096, PROT_READ) = 0
mprotect(0x7f71e9db9000, 4096, PROT_READ) = 0
mprotect(0x7f71e9fc6000, 4096, PROT_READ) = 0
mprotect(0x7f71eadc6000, 4096, PROT_READ) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f71eb680000
mprotect(0x7f71eb149000, 40960, PROT_READ) = 0
mprotect(0x7f71ea24d000, 8192, PROT_READ) = 0
mprotect(0x7f71ea546000, 4096, PROT_READ) = 0
mprotect(0x7f71ea79f000, 4096, PROT_READ) = 0
mprotect(0x7f71eb3f3000, 24576, PROT_READ) = 0
mprotect(0x7f71eba0d000, 4096, PROT_READ) = 0
mprotect(0x7f71eb629000, 4096, PROT_READ) = 0
munmap(0x7f71eb6c4000, 43552)           = 0
set_tid_address(0x7f71eb690e50)         = 20241
set_robust_list(0x7f71eb690e60, 24)     = 0
rt_sigaction(SIGRTMIN, {sa_handler=0x7f71e9ba5cb0, sa_mask=[],
sa_flags=SA_RESTORER|SA_SIGINFO, sa_restorer=0x7f71e9bb2980}, NULL, 8) = 0
rt_sigaction(SIGRT_1, {sa_handler=0x7f71e9ba5d50, sa_mask=[],
sa_flags=SA_RESTORER|SA_RESTART|SA_SIGINFO, sa_restorer=0x7f71e9bb2980}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=8192*1024}) = 0
brk(NULL)                               = 0x7fffdb4a2000
brk(0x7fffdb4c3000)                     = 0x7fffdb4c3000
gettimeofday({tv_sec=1609673745, tv_usec=387429}, {tz_minuteswest=0, tz_dsttime=0}) =
0
futex(0x7f71eb15609c, FUTEX_WAKE_PRIVATE, 2147483647) = 0
futex(0x7f71eb1560a8, FUTEX_WAKE_PRIVATE, 2147483647) = 0
time(NULL)                              = 1609673745 (2021-01-03T14:35:45+0300)
openat(AT_FDCWD, "/sys/devices/system/cpu/online", O_RDONLY|O_CLOEXEC) = 3
read(3, "0-7\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, /* 13 entries */, 32768)    = 336
getdents(3, /* 0 entries */, 32768)     = 0
close(3)                                = 0
getpid()                                = 20241
sched_getaffinity(20241, 128, [0, 1, 2, 3, 4, 5, 6, 7]) = 64
openat(AT_FDCWD, "/etc/nsswitch.conf", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=513, ...}) = 0
read(3, "# /etc/nsswitch.conf\n#\n# Example"..., 4096) = 513
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=43552, ...}) = 0
mmap(NULL, 43552, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f71eb6c4000
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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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=4096, ...}) = 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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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=4096, ...}) = 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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70)        = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70)    = -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", 0x7fffe3679a70)     = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70)    = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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", 0x7fffe3679a70) = -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(0x7f71eb6c4000, 43552)           = 0
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=43552, ...}) = 0
mmap(NULL, 43552, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f71eb6c4000
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) =
0x7f71e95e0000
mprotect(0x7f71e95eb000, 2093056, PROT_NONE) = 0
mmap(0x7f71e97ea000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xa000) = 0x7f71e97ea000
mmap(0x7f71e97ec000, 22328, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f71e97ec000
close(3)                                = 0
mprotect(0x7f71e97ea000, 4096, PROT_READ) = 0
munmap(0x7f71eb6c4000, 43552)           = 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
gettimeofday({tv_sec=1609673745, tv_usec=399158}, NULL) = 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("\x2a\x1c\xa3\x42\xdf\x17\x02\x09\x03\x45\x54\x73\xed\x15\xde\xac", 16, 0) =
16
getrandom("\xf8\x48\x09\xa9\xd0\x1e\x10\xe7\xbb\xac\x48\xcb\xb5\xdb\x18\x37", 16, 0) =
16
fstat(0, {st_mode=S_IFCHR|0660, st_rdev=makedev(4, 1), ...}) = 0
ioctl(0, TCGETS, {B38400 opost isig icanon echo ...}) = 0
read(0, "create 0 -1\n", 4096)          = 12
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f71eb690e50) = 20242
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
clock_gettime(CLOCK_MONOTONIC, {tv_sec=10895, tv_nsec=66665100}) = 0
```

```
epoll_create1(EPOLL_CLOEXEC)            = 5
epoll_ctl(5, EPOLL_CTL_ADD, 4, {0, {u32=3679145984, u64=140736872534016}}) = 0
epoll_ctl(5, EPOLL_CTL_MOD, 4, {EPOLLIN, {u32=3679145984, u64=140736872534016}}) = 0
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7f71e8dd0000
mprotect(0x7f71e8dd1000, 8388608, PROT_READ|PROT_WRITE) = 0
clone(child_stack=0x7f71e95cfb70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SET
TLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f71e95d09d0,
tls=0x7f71e95d0700, child_tidptr=0x7f71e95d09d0) = 20243
openat(AT_FDCWD, "/proc/self/task/20243/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
clock_gettime(CLOCK_MONOTONIC, {tv_sec=10895, tv_nsec=68580300}) = 0
epoll_create1(EPOLL_CLOEXEC)            = 7
epoll_ctl(7, EPOLL_CTL_ADD, 6, {0, {u32=3679146240, u64=140736872534272}}) = 0
epoll_ctl(7, EPOLL_CTL_MOD, 6, {EPOLLIN, {u32=3679146240, u64=140736872534272}}) = 0
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7f71e85c0000
mprotect(0x7f71e85c1000, 8388608, PROT_READ|PROT_WRITE) = 0
clone(child_stack=0x7f71e8dbfb70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SET
TLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tidptr=0x7f71e8dc09d0,
tls=0x7f71e8dc0700, child_tidptr=0x7f71e8dc09d0) = 20244
openat(AT_FDCWD, "/proc/self/task/20244/comm", O_RDWR) = 8
write(8, "ZMQbg/1", 7)                  = 7
close(8)                                = 0
clock_gettime(CLOCK_MONOTONIC, {tv_sec=10895, tv_nsec=69505700}) = 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
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=20241, nl_groups=00000000}, [12]) = 0
time(NULL)                              = 1609673748 (2021-01-03T14:35:48+0300)
sendto(9, "\24\0\0\0\22\0\1\3\24\254\361_\0\0\0\0\0\0\0\0", 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=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x01\x00\x08\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\xd8\
xc4\x97\xc2\x6b\x92\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x01\x00\x09\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x1c\
x1b\xb5\x88\x10\x48\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x01\x00\x13\x00\x00\x00\x43\x10\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x0a\
x00\x27\x00\x00\x13\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=76, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x04\x03\x01\x00\x00\x00\x49\x00\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 76
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x21\x03\x06\x00\x00\x00\x43\x10\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x1c\
x1b\xb5\x88\x10\x44\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x21\x03\x0f\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x1c\
x1b\xb5\x88\x10\x45\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=80, type=0x10 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673748, pid=20241},
"\x00\x00\x21\x03\x0c\x00\x00\x00\x40\x00\x00\x00\x00\x00\x00\x00\x0a\x00\x01\x00\x1e\
x1b\xb5\x88\x10\x44\x00\x00\x08\x00\x1b\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 80
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=1609673748, pid=20241}, 0}, iov_len=4096}], msg_iovlen=1, msg_controllen=0,
msg_flags=0}, 0) = 20
sendto(9, "\24\0\0\0\26\0\1\3\25\254\361_\0\0\0\0\0\0\0\0", 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=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x10\x00\x00\x08\x00\x00\x00\x08\x00\x01\x00\xa9\xfe\x93\xad\x08\x00\x04\x00\xa9\
xfe\xff\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x08\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\x3d\
x05\xea\xe6\x03\x7d\x93\xad\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x10\x00\x00\x09\x00\x00\x00\x08\x00\x01\x00\xa9\xfe\xd9\x3b\x08\x00\x04\x00\xa9\
xfe\xff\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x09\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\xe4\
x8e\x44\xf8\x07\xe3\xd9\x3b\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x18\x00\x00\x13\x00\x00\x00\x08\x00\x01\x00\xc0\xa8\x38\x01\x08\x00\x04\x00\xc0\
xa8\x38\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x13\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\x54\
x5a\x77\xc5\x46\x23\x7e\x74\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x08\x00\x00\x01\x00\x00\x00\x08\x00\x01\x00\x7f\x00\x00\x01\x08\x00\x04\x00\x7f\
xff\xff\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x80\x00\xfe\x01\x00\x00\x00\x14\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x01\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x18\x00\x00\x06\x00\x00\x00\x08\x00\x01\x00\xc0\xa8\x00\x25\x08\x00\x04\x00\xc0\
xa8\x00\xff\x14\x00\x06\x00\x51\x51\x01\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x06\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\xc5\
x1d\xff\x10\x1d\xa5\x8f\x8d\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x10\x00\x00\x0f\x00\x00\x00\x08\x00\x01\x00\xa9\xfe\xfe\x9f\x08\x00\x04\x00\xa9\
xfe\xff\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x0f\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\x91\
xc4\x97\xe8\x18\x2d\xfe\x9f\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=60, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x02\x10\x00\x00\x0c\x00\x00\x00\x08\x00\x01\x00\xa9\xfe\x13\x01\x08\x00\x04\x00\xa9\
xfe\xff\xff\x14\x00\x06\x00\xff\xff\xff\xff"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 60
recvmsg(9, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000},
msg_namelen=12, msg_iov=[{iov_base={{len=64, type=0x14 /* NLMSG_??? */,
flags=NLM_F_MULTI, seq=1609673749, pid=20241},
"\x0a\x40\x00\xfd\x0c\x00\x00\x00\x14\x00\x01\x00\xfe\x80\x00\x00\x00\x00\x00\x00\xb1\
x56\x78\x65\x03\x84\x13\x01\x14\x00\x06\x00"...}, iov_len=4096}], msg_iovlen=1,
msg_controllen=0, msg_flags=0}, 0) = 64
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=1609673749, pid=20241}, 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(4040), sin_addr=inet_addr("127.0.0.1")},
16) = 0
listen(9, 100)                          = 0
getsockname(9, {sa_family=AF_INET, sin_port=htons(4040),
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
poll([{fd=8, events=POLLIN}], 1, 0)     = 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)
clock_gettime(CLOCK_MONOTONIC, {tv_sec=10895, tv_nsec=74918500}) = 0
poll([{fd=8, events=POLLIN}], 1, 5000)  = 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)
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(6, "\1\0\0\0\0\0\0\0", 8)         = 8
fstat(1, {st_mode=S_IFCHR|0660, st_rdev=makedev(4, 1), ...}) = 0
ioctl(1, TCGETS, {B38400 opost isig icanon echo ...}) = 0
write(1, "OK: 20242\n", 10)             = 10
read(0, "exit\n", 4096)                 = 5
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
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=20242, si_uid=1000,
si_status=0, si_utime=0, si_stime=0} ---
poll([{fd=8, events=POLLIN}], 1, 0)     = 0 (Timeout)
poll([{fd=8, events=POLLIN}], 1, 0)     = 0 (Timeout)
write(6, "\1\0\0\0\0\0\0\0", 8)         = 8
lseek(0, -1, SEEK_CUR)                  = -1 ESPIPE (Illegal seek)
exit_group(0)                           = ?
+++ exited with 0 +++
```

6. Выводы

Данная лабораторная работа была направлена на изучении технологии очереди сообщений, на основе которой необходимо было построить сеть с заданной топологией.

Наряду с каналами и отображаемыми файлами, очереди сообщений являются достаточно удобным способом для взаимодействия между процессами. ZeroMQ предоставляет достаточно простой интерфейс для передачи сообщений, а также поддерживает все возможные типы соединений.

Полученные мной навыки работы с очередями сообщений можно использовать при проектировании мессенджеров, многопользовательских игр, да и вообще для любых мультипроцессорных программ.