Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №5-7 по курсу**

**«Операционные системы»**

Группа: М80-206Б-22

Студент: Седов М.Д

Преподаватель: Миронов Е.С.

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

Дата: 02.12.2023

Москва, 2023

**Постановка задачи**

**Вариант 32.**

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 той топологией, которая определена вариантом. Связь между узлами необходимо осуществить при помощи технологии очередей сообщений. Также в данной системе необходимо предусмотреть проверку доступности узлов в соответствии с вариантом. При убийстве («kill -9») 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. Список основных поддерживаемых команд:

**Создание нового вычислительного узла**

Формат команды: create id [parent]

id – целочисленный идентификатор нового вычислительного узла

parent – целочисленный идентификатор родительского узла. Если топологией не предусмотрено введение данного параметра, то его необходимо игнорировать (если его ввели)

Формат вывода:

«Ok: pid», где pid – идентификатор процесса для созданного вычислительного узла

«Error: Already exists» - вычислительный узел с таким идентификатором уже существует «Error: Parent not found» - нет такого родительского узла с таким идентификатором

«Error: Parent is unavailable» - родительский узел существует, но по каким-то причинам с ним не удается связаться

«Error: [Custom error]» - любая другая обрабатываемая ошибка

Пример: > create 10 5 Ok: 3128

*Примечания: создание нового управляющего узла осуществляется пользователем программы при помощи запуска исполняемого файла. Id и pid — это разные идентификаторы.*

**Исполнение команды на вычислительном узлеНабора команд 2 (локальный целочисленный словарь)**

Формат команды сохранения значения: exec id name value id – целочисленный идентификатор вычислительного узла, на который отправляется команда

name – ключ, по которому будет сохранено значение (строка формата [A-Za-z0-9]+)

value – целочисленное значение

Формат команды загрузки значения: exec id name

Формат вывода:

«Ok:id: [result]», где result – результат выполненной команды

«Error:id: Not found» - вычислительный узел с таким идентификатором не найден

«Error:id: Node is unavailable» - по каким-то причинам не удается связаться с вычислительным узлом

«Error:id: [Custom error]» - любая другая обрабатываемая ошибка

Пример:

> exec 10 MyVar

Ok:10: 'MyVar' not found

> exec 10 MyVar 5

Ok:10

> exec 12 MyVar

Ok:12: 'MyVar' not found

> exec 10 MyVar

Ok:10: 5

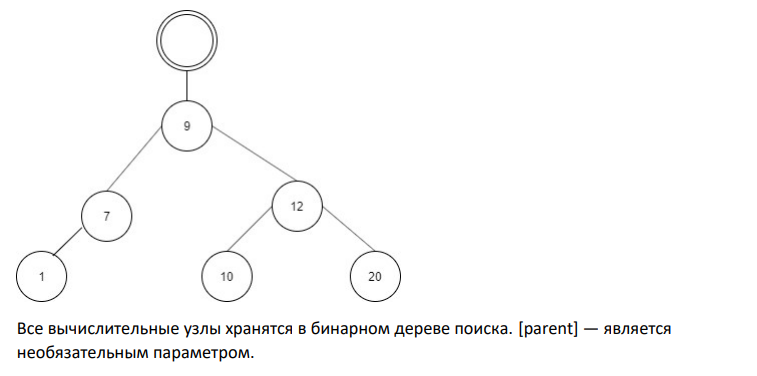
> exec 10 MyVar 7

Ok:10

> exec 10 MyVar

Ok:10: 7

*Примечания: Можно использовать std:map***Топология 3**

****

**Тип проверки доступности узлов**

**Команда проверки 3**

Формат команды: heartbit time

Каждый узел начинает сообщать раз в time миллисекунд о том, что он работоспособен. Если от узла нет сигнала в течении 4\*time миллисекунд, то должна выводится пользователю строка: «Heartbit: node id is unavailable now», где id – идентификатор недоступного вычислительного узла.

Пример:

> heartbit 2000

Ok

**Общий метод и алгоритм решения**

Распределенная система состоит из двух узлов: управляющего (control.cpp) и вычислительного (calculate.cpp), объеденные топологией, представляющей из себя бинарное дерево поиска (binTree.h). Связь между узлами происходит с помощью брокера сообщений ZeroMQ, а именно с помощью использования сокетов типа PUB-SUB (publisher-subscriber).  
Сокеты же инициализируются с помощью zmq\_ctx\_new() {создает контекст для сокета), zmq\_socket() {создающий сокет определенного типа на конкретном контексте} и подключение/привязка сокета к заданному адресу и порту {zmq\_connect() / zmq\_bind() }.

В коде управляющего узла создан интерфейс по обработки команд и передачи их на вычислительные узлы{zmq\_send()}. В коде вычислительного узла реализована логика ветвления процессов подобно дереву бинарного поиска, также происходит получение команд {zmq\_recv()}. Удаление узлов происходит путем завершения процессов и их потомков, но из дерева в управляющем узле они не удаляются, а помечаются как недоступные. Команда проверки доступности узлов (heartbit) передает команду узлам в заданное время отправлять отчет о своей работоспособности.

**Код программы**

**binTree.h**

#include <iostream>

#include <map>

#include <zmq.h>

struct Node {

    int id;

    bool exist{true};

    // std::map<std::string, int> values;

    Node\* left;

    Node\* right;

    Node(int nodeId) : id(nodeId), left(nullptr), right(nullptr) {}

};

class BinarySearchTree {

private:

    Node\* root;

public:

    BinarySearchTree() : root(nullptr) {}

    ~BinarySearchTree() {

        destroyTree(root);

    }

    void printTree() {

        printTree(root, 0);

    }

    int findParentId(int nodeId) {

        return findParentId(root, nodeId);

    }

    int findInsert(int nodeId) {

        return findInsertionPoint(root, nodeId);

    }

    std::string createNode(int nodeId) {

        Node\* newNode = new Node(nodeId);

        if (root == nullptr) {

            root = newNode;

            return "Ok:";

        } else {

            Node\* currentNode = root;

            Node\* parentNode = nullptr;

            while (currentNode != nullptr) {

                parentNode = currentNode;

                if (nodeId < currentNode->id) {

                    currentNode = currentNode->left;

                } else if (nodeId > currentNode->id) {

                    currentNode = currentNode->right;

                } else {

                    delete newNode; // Узел с таким идентификатором уже существует

                    return "Error: Already exists";

                }

            }

            if (nodeId < parentNode->id) {

                parentNode->left = newNode;

            } else {

                parentNode->right = newNode;

            }

            return "Ok:";

        }

    }

    std::string findNode(int nodeId) {

        return findNode(root, nodeId);

    }

    int getSize() {

        return getSize(root);

    }

    void destroyTree(Node\* currentNode) {

        if (currentNode == nullptr) {

            return;

        }

        destroyTree(currentNode->left);

        destroyTree(currentNode->right);

        delete currentNode;

    }

    std::string killNodeAndChildren(int nodeId, void\* socket) {

        std::string answer = findNode(root, nodeId);

        if (answer == "Ok") {

            Node\* node = findNodePointer(root, nodeId);

            sendKillMessage(node, socket);

            return "Ok: All nodes killed";

        }

        return answer;

    }

private:

    void sendKillMessage(Node\* node, void\* socket) {

        if (node == nullptr || !node->exist) {

            return;

        }

        node->exist = false;

        std::string msg = "kill " + std::to\_string(node->id);

        zmq\_send(socket, msg.c\_str(), msg.length(), 0);

        sendKillMessage(node->left, socket);

        sendKillMessage(node->right, socket);

    }

    std::string findNode(Node\* currentNode, int nodeId) {

        if (currentNode == nullptr) {

            return "Error:"  + std::to\_string(nodeId) + ": Not found";

        }

        if (currentNode->id == nodeId) {

            if (currentNode->id == -1) {

                return "Error:" + std::to\_string(nodeId) + ": Not calculation node";

            } else {

                return "Ok";

            }

        } else {

            std::string leftResult = findNode(currentNode->left, nodeId);

            if (leftResult != "Error:"  + std::to\_string(nodeId) + ": Not found") {

                return leftResult;

            }

            std::string rightResult = findNode(currentNode->right, nodeId);

            return rightResult;

        }

    }

    Node\* findNodePointer(Node\* currentNode, int nodeId) {

        if (currentNode == nullptr) {

            return nullptr;

        }

        if (currentNode->id == nodeId) {

            return currentNode;

        }

        Node\* leftResult = findNodePointer(currentNode->left, nodeId);

        if (leftResult != nullptr) {

            return leftResult;

        }

        Node\* rightResult = findNodePointer(currentNode->right, nodeId);

        return rightResult;

    }

    int findParentId(Node\* currentNode, int nodeId) {

        if (currentNode == nullptr || currentNode->id == nodeId) {

            return -1;

        }

        if ((currentNode->left != nullptr && currentNode->left->id == nodeId) ||

            (currentNode->right != nullptr && currentNode->right->id == nodeId)) {

            return currentNode->id;

        }

        int leftResult = findParentId(currentNode->left, nodeId);

        if (leftResult != -1) {

            return leftResult;

        }

        int rightResult = findParentId(currentNode->right, nodeId);

        return rightResult;

    }

    int findInsertionPoint(Node\* currentNode, int nodeId) {

        if (currentNode == nullptr) {

            return -1;

        }

        if (nodeId < currentNode->id) {

            if (currentNode->left == nullptr) {

                return currentNode->id;

            } else {

                return findInsertionPoint(currentNode->left, nodeId);

            }

        } else if (nodeId > currentNode->id) {

            if (currentNode->right == nullptr) {

                return currentNode->id;

            } else {

                return findInsertionPoint(currentNode->right, nodeId);

            }

        } else {

            return -1;

        }

    }

    void printTree(Node\* currentNode, int indentLevel) {

        if (currentNode == nullptr) {

            return;

        }

        printTree(currentNode->right, indentLevel + 1);

        for (int i = 0; i < indentLevel; i++) {

            std::cout << "\t";

        }

        std::cout << "|----- " << currentNode->id << std::endl;

        printTree(currentNode->left, indentLevel + 1);

    }

    int getSize(Node\* currentNode) {

        if (currentNode == nullptr) {

            return 0;

        }

        int leftSize = getSize(currentNode->left);

        int rightSize = getSize(currentNode->right);

        return leftSize + rightSize + 1;

    }

};

**control.cpp**

#include <zmq.h>

#include "binTree.h"

#include <iostream>

#include <string>

#include <unistd.h>

#include <regex>

#include <chrono>

#include <thread>

int main() {

    void\* context = zmq\_ctx\_new();

    void\* socket = zmq\_socket(context, ZMQ\_PUB);

    zmq\_bind(socket, "tcp://0.0.0.0:5555");

    BinarySearchTree tree;

    tree.createNode(-1);

    std::regex createReg("^create ([0-9]+)$");

    std::regex execSaveReg("^exec ([0-9]+) ([a-zA-Z]+) ([0-9]+)$");

    std::regex execLoadReg("^exec ([0-9]+) ([A-Za-z0-9]+)$");

    std::regex killReg("^kill ([0-9]+)$");

    std::regex heartbitReg("^heartbit ([0-9]+)$");

    std::regex printReg("^print$");

    std::regex exitReg("^exit$");

    std::smatch matches;

    std::cout << "Started" << std::endl;

    int timebreak;

    std::string cmd;

    while (true){

        timebreak = 500;

        std::cout << "Enter command: ";

        std::getline(std::cin, cmd);

        //std::cout << '\n';

        //std::cout << "cmd: " << cmd << "" << std::endl;

        if (std::regex\_match(cmd, matches, createReg)) {

            int id = std::stoi(matches[1]);

            std::string answer = tree.createNode(id);

            int pos = tree.findParentId(id);

            //std::cout << "POS" << pos << std::endl;

            if(pos == -1){

                if(answer == "Ok:"){

                    pid\_t pid = fork();

                    if (pid == 0) {

                        execl("./calculate", "calculate", std::to\_string(id).c\_str(), NULL);

                        std::cout << "Error execl" << std::endl;

                        exit(1);

                    }

                    std::cout << answer << " " << pid << std::endl;

                }

                else{

                    std::cout << answer << std::endl;

                }

            }

            else{

                if(answer == "Ok:"){

                    //std::cout << "CREATE IN DAD" << std::endl;

                    std::string msg = "create " + std::to\_string(pos) + " " + std::to\_string(id);

                    zmq\_send(socket, msg.c\_str(), msg.length(), 0);

                }

                else{

                    std::cout << answer << std::endl;

                }

            }

        }

        else if (std::regex\_match(cmd, matches, execSaveReg)) {

            int id = std::stoi(matches[1]);

            std::string answer = tree.findNode(id);

            if(answer == "Ok"){

                std::string msg = "save " + matches[1].str() + " " + matches[2].str() + " " + matches[3].str();

                zmq\_send(socket, msg.c\_str(), msg.length(), 0);

            }

            else{

                std::cout << answer << std::endl;

            }

        }

        else if (std::regex\_match(cmd, matches, execLoadReg)) {

            int id = std::stoi(matches[1]);

            std::string answer = tree.findNode(id);

            if(answer == "Ok"){

                std::string msg = "load " + matches[1].str() + " " + matches[2].str();

                zmq\_send(socket, msg.c\_str(), msg.length(), 0);

            }

            else{

                std::cout << answer << std::endl;

            }

        }

        else if (std::regex\_match(cmd, matches, heartbitReg)) {

            int time = std::stoi(matches[1]);

            timebreak = time \*4;

            std::string msg = "heartbit " + std::to\_string(time);

            zmq\_send(socket, msg.c\_str(), msg.length(), 0);

        }

        else if (std::regex\_match(cmd, matches, killReg)) {

            int id = std::stoi(matches[1]);

            tree.killNodeAndChildren(id, socket);

        }

        else if (std::regex\_match(cmd, matches, printReg)) {

            tree.printTree();

        }

        else if (std::regex\_match(cmd, matches, exitReg)) {

            std::cout << "Exit" << std::endl;

            zmq\_send(socket, "exit", 4, 0);

            break;

        }

        else{

            std::cout << "Wrong command" << std::endl;

        }

        std::this\_thread::sleep\_for(std::chrono::milliseconds(timebreak));

    }

    zmq\_close(socket);

    zmq\_ctx\_destroy(context);

    return 0;

}

**calculate.cpp**

#include <zmq.h>

#include <iostream>

#include <unistd.h>

#include <string.h>

#include <map>

#include <sstream>

#include <thread>

#include <chrono>

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

    void\* context = zmq\_ctx\_new();

    void\* socket = zmq\_socket(context, ZMQ\_SUB);

    zmq\_connect(socket, "tcp://0.0.0.0:5555");

    zmq\_setsockopt(socket, ZMQ\_SUBSCRIBE, "", 0);

    int ID = std::stoi(argv[1]);

    //std::cout << "bEBRA" << ID << std::endl;

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

    char buffer[1024];

    while (true) {

        memset(buffer, 0, sizeof(buffer));

        //std::cout << "Not Received: "<< std::endl;

        zmq\_recv(socket, buffer, sizeof(buffer), 0);

        std::string message(buffer);

        //std::cout << "Received: " << ID << ":" << message << std::endl;

        std::istringstream iss(message);

        std::string command;

        iss >> command;

        if (command == "create") {

            int pos;

            iss >> pos;

            if(ID == pos){

                //std::cout << "Creating in " << ID <<  std::endl;

                std::string id;

                iss >> id;

                pid\_t pid = fork();

                if (pid == 0) {

                    //std::cout << "NEW FORK: " << std::endl;

                    execl("./calculate", "calculate", id.c\_str(), NULL);

                    std::cout << "Error execl" << std::endl;

                    exit(1);

                }

                std::cout << "Ok: " << pid << std::endl;

            }

        }

        else if (command == "save") {

            //std::cout << "Saving: " << ID << std::endl;

            int id;

            iss >> id;

            if(ID == id){

                std::string name;

                int value;

                iss >> name >> value;

                values[name] = value;

                std::cout << "Ok:" << id << std::endl;

            }

        }

        else if (command == "load") {

            //std::cout << "Loading: " << ID << std::endl;

            int id;

            iss >> id;

            if(ID == id){

                std::string name;

                iss >> name;

                if (values.count(name) > 0) {

                    int value = values[name];

                    std::cout << "Ok:" << id << ": " << value << std::endl;

                }

                else{

                    std::cout << "Error:" << id << ": " << name << std::endl;

                }

            }

        }

        else if (command == "heartbit") {

            int time;

            iss >> time;

            for(int i = 0; i < 4; i++){

                std::cout << "Heartbit: " << ID << std::endl;

                std::this\_thread::sleep\_for(std::chrono::milliseconds(time));

            }

        }

        else if (command == "kill") {

            int id;

            iss >> id;

            if(ID == id){

                std::cout << "Killed:" << ID << std::endl;

                break;

            }

        }

        else if (command == "exit") {

            break;

        }

    }

    // Закрытие сокета и освобождение ресурсов

    zmq\_close(socket);

    zmq\_ctx\_destroy(context);

    return 0;

}

**Протокол работы программы**

**Тестирование:**

maksim@caseofpeace:~/course2/sem3/os/lab5-7/src$ ./control

Started

Enter command: create 10

Ok: 11893

Enter command: create 12

Ok: 11934

Enter command: create 8

Ok: 11968

Enter command: create 11

Ok: 12001

Enter command: create 13

Ok: 12010

Enter command: create 13

Error: Already exists

Enter command: exec 14 naMe1

Error:14: Not found

Enter command: exec 13 naMe1

Error:13: naMe1

Enter command: exec 12 naMe1 10

Ok:12

Enter command: exec 12 naMe1

Ok:12: 10

Enter command: exec 12 naMe2 1

Ok:12

Enter command: exec 12 naMe2

Ok:12: 1

Enter command: heartbit 1000

Heartbit: 12

Heartbit: 10

Heartbit: 13

Heartbit: 8

Heartbit: 11

Heartbit: 12

Heartbit: 11

Heartbit: 13

Heartbit: 10

Heartbit: 8

Heartbit: 12

Heartbit: 11

Heartbit: 10

Heartbit: 13

Heartbit: 8

Heartbit: 12

Heartbit: 11

Heartbit: 8

Heartbit: 10

Heartbit: 13

Enter command: print

|----- 13

|----- 12

|----- 11

|----- 10

|----- 8

|----- -1

Enter command: kill 12

Ok: All nodes killed

Killed:12

Killed:11

Killed:13

Enter command: heartbit 2000

Heartbit: 10

Heartbit: 8

Heartbit: 8

Heartbit: 10

Heartbit: 8

Heartbit: 10

Heartbit: 8

Heartbit: 10

Enter command: exit

Exit

**STRACE:**

maksim@caseofpeace:~/course2/sem3/os/lab5-7/src$ strace -f ./control

execve("./control", ["./control"], 0x7fffd591a428 /\* 31 vars \*/) = 0

brk(NULL) = 0x7fffc3b08000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fffcb1be8b0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7ff410ba0000

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

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=23887, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 23887, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7ff410b5a000

close(3) = 0

openat(AT\_FDCWD, "/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\0\240\233\1\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=634936, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 636784, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7ff410ab0000

mmap(0x7ff410ac8000, 397312, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x18000) = 0x7ff410ac8000

**……………………………………………………………………………………………..**

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7ff4100c0910, parent\_tid=0x7ff4100c0910, exit\_signal=0, stack=0x7ff40f8c0000, stack\_size=0x7ffc80, tls=0x7ff4100c0640}, 88) = -1 ENOSYS (Function not implemented)

clone(child\_stack=0x7ff4100bfc70, flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTIDstrace: Process 14828 attached

, parent\_tid=[14828], tls=0x7ff4100c0640, child\_tidptr=0x7ff4100c0910) = 14828

[pid 14828] set\_robust\_list(0x7ff4100c0920, 24 <unfinished ...>

[pid 14827] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 14828] <... set\_robust\_list resumed>) = 0

[pid 14827] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 14828] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 14827] eventfd2(0, EFD\_CLOEXEC <unfinished ...>

[pid 14828] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 14827] <... eventfd2 resumed>) = 6

[pid 14828] rt\_sigprocmask(SIG\_BLOCK, ~[RTMIN RT\_1], <unfinished ...>

[pid 14827] fcntl(6, F\_GETFL <unfinished ...>

[pid 14828] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 14827] <... fcntl resumed>) = 0x2 (flags O\_RDWR)

[pid 14828] sched\_getparam(14828, <unfinished ...>

[pid 14827] fcntl(6, F\_SETFL, O\_RDWR|O\_NONBLOCK <unfinished ...>

**……………………………………………………………………………………………………..**

[pid 14827] recvmsg(9, {msg\_name={sa\_family=AF\_NETLINK, nl\_pid=0, nl\_groups=00000000}, msg\_namelen=12, msg\_iov=[{iov\_base=[{nlmsg\_len=20, nlmsg\_type=NLMSG\_DONE, nlmsg\_flags=NLM\_F\_MULTI, nlmsg\_seq=1704380934, nlmsg\_pid=14827}, 0], iov\_len=4096}], msg\_iovlen=1, msg\_controllen=0, msg\_flags=0}, 0) = 20

[pid 14827] close(9) = 0

**[pid 14827] socket(AF\_INET, SOCK\_STREAM|SOCK\_CLOEXEC, IPPROTO\_TCP) = 9**

**[pid 14827] setsockopt(9, SOL\_SOCKET, SO\_REUSEADDR, [1], 4) = 0**

**[pid 14827] bind(9, {sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("0.0.0.0")}, 16) = 0**

[pid 14827] listen(9, 100) = 0

[pid 14827] getsockname(9, {sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("0.0.0.0")}, [128 => 16]) = 0

[pid 14827] getsockname(9, {sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("0.0.0.0")}, [128 => 16]) = 0

[pid 14827] getpid() = 14827

[pid 14827] write(6, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=3283220064, u64=140736476608096}}], 256, -1) = 1

[pid 14827] getpid( <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 14827] <... getpid resumed>) = 14827

[pid 14829] <... getpid resumed>) = 14827

[pid 14827] write(8, "\1\0\0\0\0\0\0\0", 8 <unfinished ...>

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 14827] <... write resumed>) = 8

[pid 14829] <... poll resumed>) = 1 ([{fd=6, revents=POLLIN}])

[pid 14827] futex(0x7ff410aa9808, FUTEX\_WAKE\_PRIVATE, 2147483647 <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 14827] <... futex resumed>) = 0

[pid 14829] <... getpid resumed>) = 14827

[pid 14829] read(6, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 14829] mmap(NULL, 134217728, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_NORESERVE, -1, 0) = 0x7ff4070b0000

[pid 14829] munmap(0x7ff4070b0000, 16056320) = 0

[pid 14829] munmap(0x7ff40c000000, 51052544) = 0

[pid 14829] mprotect(0x7ff408000000, 135168, PROT\_READ|PROT\_WRITE) = 0

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_ADD, 9, {events=0, data={u32=134220656, u64=140686082968432}}) = 0

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=134220656, u64=140686082968432}}) = 0

[pid 14829] getpid() = 14827

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0) = 0 (Timeout)

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 14827] newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}, AT\_EMPTY\_PATH) = 0

[pid 14827] write(1, "Started\n", 8Started

) = 8

[pid 14827] write(1, "Enter command: ", 15Enter command: ) = 15

[pid 14827] newfstatat(0, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...}, AT\_EMPTY\_PATH) = 0

[pid 14827] read(0,

"create 10\n", 1024) = 10

[pid 14827] clone(child\_stack=NULL, flags=CLONE\_CHILD\_CLEARTID|CLONE\_CHILD\_SETTID|SIGCHLDstrace: Process 15520 attached

<unfinished ...>

[pid 15520] set\_robust\_list(0x7ff4100e0ca0, 24 <unfinished ...>

[pid 14827] <... clone resumed>, child\_tidptr=0x7ff4100e0c90) = 15520

[pid 15520] <... set\_robust\_list resumed>) = 0

[pid 14827] write(1, "Ok: 15520\n", 10Ok: 15520

) = 10

[pid 14827] clock\_nanosleep(CLOCK\_REALTIME, 0, {tv\_sec=0, tv\_nsec=500000000}, <unfinished ...>

[pid 15520] execve("./calculate", ["calculate", "10"], 0x7fffcb1bea88 /\* 31 vars \*/) = 0

[pid 15520] brk(NULL) = 0x7fffd8c6f000

[pid 15520] arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7fffe0359b60) = -1 EINVAL (Invalid argument)

[pid 15520] mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f5c2bfb0000

**……………………………………………………………………………………………**

[pid 15520] brk(0x7fffd8cb1000) = 0x7fffd8cb1000

[pid 15520] futex(0x7f5c2bec9808, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

[pid 15520] getpid() = 15520

[pid 15520] write(6, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 15520] getpid( <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=3637012064, u64=140736830400096}}], 256, -1) = 1

[pid 15520] <... getpid resumed>) = 15520

[pid 15522] getpid( <unfinished ...>

[pid 15520] write(8, "\1\0\0\0\0\0\0\0", 8 <unfinished ...>

[pid 15522] <... getpid resumed>) = 15520

[pid 15520] <... write resumed>) = 8

[pid 15522] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 15520] getpid( <unfinished ...>

[pid 15522] <... poll resumed>) = 1 ([{fd=6, revents=POLLIN}])

[pid 15520] <... getpid resumed>) = 15520

[pid 15522] getpid( <unfinished ...>

[pid 15520] poll([{fd=8, events=POLLIN}], 1, -1 <unfinished ...>

[pid 15522] <... getpid resumed>) = 15520

[pid 15520] <... poll resumed>) = 1 ([{fd=8, revents=POLLIN}])

[pid 15522] read(6, <unfinished ...>

[pid 15520] getpid( <unfinished ...>

[pid 15522] <... read resumed>"\1\0\0\0\0\0\0\0", 8) = 8

[pid 15520] <... getpid resumed>) = 15520

[pid 15522] mmap(NULL, 134217728, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_NORESERVE, -1, 0 <unfinished ...>

[pid 15520] read(8, <unfinished ...>

[pid 15522] <... mmap resumed>) = 0x7f5c224c0000

[pid 15520] <... read resumed>"\1\0\0\0\0\0\0\0", 8) = 8

[pid 15522] munmap(0x7f5c224c0000, 28573696 <unfinished ...>

[pid 15520] getpid( <unfinished ...>

[pid 15522] <... munmap resumed>) = 0

[pid 15520] <... getpid resumed>) = 15520

[pid 15522] munmap(0x7f5c28000000, 38535168 <unfinished ...>

[pid 15520] poll([{fd=8, events=POLLIN}], 1, 0 <unfinished ...>

[pid 15522] <... munmap resumed>) = 0

[pid 15520] <... poll resumed>) = 0 (Timeout)

[pid 15522] mprotect(0x7f5c24000000, 135168, PROT\_READ|PROT\_WRITE <unfinished ...>

[pid 15520] getpid( <unfinished ...>

[pid 15522] <... mprotect resumed>) = 0

[pid 15520] <... getpid resumed>) = 15520

[pid 15520] poll([{fd=8, events=POLLIN}], 1, -1 <unfinished ...>

**[pid 15522] socket(AF\_INET, SOCK\_STREAM|SOCK\_CLOEXEC, IPPROTO\_TCP) = 9**

[pid 15522] fcntl(9, F\_GETFL) = 0x80002 (flags O\_RDWR|O\_CLOEXEC)

[pid 15522] fcntl(9, F\_SETFL, O\_RDWR|O\_NONBLOCK|O\_CLOEXEC) = 0

**[pid 15522] connect(9, {sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("0.0.0.0")}, 16) = -1 EINPROGRESS (Operation now in progress)**

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134220656, u64=140686082968432}}], 256, -1) = 1

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_ADD, 9, {events=0, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] accept4(9, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... accept4 resumed>{sa\_family=AF\_INET, sin\_port=htons(54363), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16], SOCK\_CLOEXEC) = 10

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLOUT, data={u32=603984944, u64=140033717703728}} <unfinished ...>

**[pid 14829] setsockopt(10, SOL\_TCP, TCP\_NODELAY, [1], 4 <unfinished ...>**

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... setsockopt resumed>) = 0f

[pid 15522] getpid( <unfinished ...>

[pid 14829] getpeername(10, <unfinished ...>

[pid 15522] <... getpid resumed>) = 15520

[pid 14829] <... getpeername resumed>{sa\_family=AF\_INET, sin\_port=htons(54363), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 15522] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 14829] getsockname(10, <unfinished ...>

[pid 15522] <... poll resumed>) = 0 (Timeout)

[pid 14829] <... getsockname resumed>{sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 15522] epoll\_wait(7, <unfinished ...>

**…………………………………………………………………………………………………………….**

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN, data={u32=134220752, u64=140686082968528}} <unfinished ...>

[pid 15522] <... sendto resumed>) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=357603400}) = 0

[pid 15522] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=357691300}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134220752, u64=140686082968528}}], 256, 29997) = 1

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=603984944, u64=140033717703728}}], 256, 29997) = 1

[pid 14829] <... recvfrom resumed>"\3", 2, 0, NULL, NULL) = 1

[pid 15522] recvfrom(9, <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN|EPOLLOUT, data={u32=134220752, u64=140686082968528}} <unfinished ...>

[pid 15522] <... recvfrom resumed>"\3", 2, 0, NULL, NULL) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>0x7ff408001303, 53, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 15522] recvfrom(9, <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 15522] <... recvfrom resumed>0x7f5c24001bf3, 53, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=358270600}) = 0

[pid 15522] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=358364200}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=134220752, u64=140686082968528}}], 256, 29996) = 1

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] sendto(10, "\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, 0 <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=603984944, u64=140033717703728}}], 256, 29996) = 1

[pid 15522] sendto(9, "\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, 0 <unfinished ...>

[pid 14829] <... sendto resumed>) = 53

[pid 15522] <... sendto resumed>) = 53

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN, data={u32=134220752, u64=140686082968528}} <unfinished ...>

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 15522] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=358844300}) = 0

[pid 15522] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=358867200}) = 0

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134220752, u64=140686082968528}}], 256, 29996) = 1

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=603984944, u64=140033717703728}}], 256, 29996) = 1

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] recvfrom(9, <unfinished ...>

[pid 14829] <... recvfrom resumed>"\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, NULL) = 53

[pid 15522] <... recvfrom resumed>"\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, NULL) = 53

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN|EPOLLOUT, data={u32=134220752, u64=140686082968528}}) = 0

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>0x7ff408003b48, 8192, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 15522] recvfrom(9, <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 15522] <... recvfrom resumed>0x7f5c24003d18, 8192, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=359457100}) = 0

[pid 15522] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=359540800}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=134220752, u64=140686082968528}}], 256, 29995) = 1

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] sendto(10, "\4\31\5READY\vSocket-Type\0\0\0\3PUB", 27, 0, NULL, 0 <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=603984944, u64=140033717703728}}], 256, 29995) = 1

[pid 14829] <... sendto resumed>) = 27

[pid 15522] sendto(9, "\4\31\5READY\vSocket-Type\0\0\0\3SUB", 27, 0, NULL, 0 <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 15522] <... sendto resumed>) = 27

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=359833900}) = 0

[pid 15522] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... clock\_gettime resumed>{tv\_sec=7227, tv\_nsec=359915200}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN|EPOLLOUT, data={u32=134220752, u64=140686082968528}}], 256, 29995) = 1

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN, data={u32=134220752, u64=140686082968528}} <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLIN|EPOLLOUT, data={u32=603984944, u64=140033717703728}}], 256, 29995) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>"\4\31\5READY\vSocket-Type\0\0\0\3SUB", 8192, 0, NULL, NULL) = 27

[pid 15522] recvfrom(9, "\4\31\5READY\vSocket-Type\0\0\0\3PUB", 8192, 0, NULL, NULL) = 27

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN|EPOLLOUT, data={u32=134220752, u64=140686082968528}}) = 0

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN, data={u32=134220752, u64=140686082968528}} <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 15522] sendto(9, "\4\n\tSUBSCRIBE", 12, 0, NULL, 0 <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... sendto resumed>) = 12

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134220752, u64=140686082968528}}], 256, -1) = 1

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14829] recvfrom(10, <unfinished ...>

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=603984944, u64=140033717703728}}], 256, -1) = 1

[pid 14829] <... recvfrom resumed>"\4\n\tSUBSCRIBE", 8192, 0, NULL, NULL) = 12

[pid 15522] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=603984944, u64=140033717703728}} <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 15522] <... epoll\_ctl resumed>) = 0

[pid 15522] epoll\_wait(7, <unfinished ...>

[pid 14827] <... clock\_nanosleep resumed>0x7fffcb1be690) = 0

[pid 14827] write(1, "Enter command: ", 15Enter command: ) = 15

[pid 14827] read(0, create 12

"create 12\n", 1024) = 10

[pid 14827] getpid() = 14827

[pid 14827] poll([{fd=8, events=POLLIN}], 1, 0) = 1 ([{fd=8, revents=POLLIN}])

[pid 14827] getpid() = 14827

[pid 14827] read(8, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 14827] getpid() = 14827

[pid 14827] poll([{fd=8, events=POLLIN}], 1, 0) = 0 (Timeout)

[pid 14827] getpid() = 14827

[pid 14827] write(6, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=3283220064, u64=140736476608096}}], 256, -1) = 1

[pid 14827] clock\_nanosleep(CLOCK\_REALTIME, 0, {tv\_sec=0, tv\_nsec=500000000}, <unfinished ...>

[pid 14829] getpid() = 14827

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0) = 1 ([{fd=6, revents=POLLIN}])

[pid 14829] getpid() = 14827

[pid 14829] read(6, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 10, {events=EPOLLIN|EPOLLOUT, data={u32=134220752, u64=140686082968528}}) = 0

**[pid 14829] sendto(10, "\0\fcreate 10 12", 14, 0, NULL, 0) = 14**

[pid 15522] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=603984944, u64=140033717703728}}], 256, -1) = 1

[pid 14829] getpid() = 14827

[pid 15522] recvfrom(9, <unfinished ...>

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

**[pid 15522] <... recvfrom resumed>"\0\fcreate 10 12", 8192, 0, NULL, NULL) = 14**

[pid 14829] <... poll resumed>) = 0 (Timeout)

[pid 15522] getpid( <unfinished ...>

**……………………………………………………………………………………………**

[pid 16367] <... write resumed>) = 8

[pid 16369] <... poll resumed>) = 1 ([{fd=6, revents=POLLIN}])

[pid 16367] getpid( <unfinished ...>

[pid 16369] getpid( <unfinished ...>

[pid 16367] <... getpid resumed>) = 16367

[pid 16369] <... getpid resumed>) = 16367

[pid 16367] poll([{fd=8, events=POLLIN}], 1, -1 <unfinished ...>

[pid 16369] read(6, <unfinished ...>

[pid 16367] <... poll resumed>) = 1 ([{fd=8, revents=POLLIN}])

[pid 16369] <... read resumed>"\1\0\0\0\0\0\0\0", 8) = 8

[pid 16367] getpid( <unfinished ...>

[pid 16369] mmap(NULL, 134217728, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_NORESERVE, -1, 0 <unfinished ...>

[pid 16367] <... getpid resumed>) = 16367

[pid 16369] <... mmap resumed>) = 0x7f3188930000

[pid 16367] read(8, "\1\0\0\0\0\0\0\0", 8) = 8

[pid 16369] munmap(0x7f3188930000, 57475072 <unfinished ...>

[pid 16367] getpid( <unfinished ...>

[pid 16369] <... munmap resumed>) = 0

[pid 16367] <... getpid resumed>) = 16367

[pid 16369] munmap(0x7f3190000000, 9633792 <unfinished ...>

[pid 16367] poll([{fd=8, events=POLLIN}], 1, 0 <unfinished ...>

[pid 16369] <... munmap resumed>) = 0

[pid 16367] <... poll resumed>) = 0 (Timeout)

[pid 16369] mprotect(0x7f318c000000, 135168, PROT\_READ|PROT\_WRITE <unfinished ...>

[pid 16367] getpid( <unfinished ...>

[pid 16369] <... mprotect resumed>) = 0

[pid 16367] <... getpid resumed>) = 16367

[pid 16367] poll([{fd=8, events=POLLIN}], 1, -1 <unfinished ...>

**[pid 16369] socket(AF\_INET, SOCK\_STREAM|SOCK\_CLOEXEC, IPPROTO\_TCP) = 9**

[pid 16369] fcntl(9, F\_GETFL) = 0x80002 (flags O\_RDWR|O\_CLOEXEC)

[pid 16369] fcntl(9, F\_SETFL, O\_RDWR|O\_NONBLOCK|O\_CLOEXEC) = 0

**[pid 16369] connect(9, {sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("0.0.0.0")}, 16) = -1 EINPROGRESS (Operation now in progress)**

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134220656, u64=140686082968432}}], 256, -1) = 1

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_ADD, 9, {events=0, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] accept4(9, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... accept4 resumed>{sa\_family=AF\_INET, sin\_port=htons(60978), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16], SOCK\_CLOEXEC) = 11

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLOUT, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] setsockopt(11, SOL\_TCP, TCP\_NODELAY, [1], 4 <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... setsockopt resumed>) = 0

[pid 16369] getpid( <unfinished ...>

[pid 14829] getpeername(11, <unfinished ...>

[pid 16369] <... getpid resumed>) = 16367

[pid 14829] <... getpeername resumed>{sa\_family=AF\_INET, sin\_port=htons(60978), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 16369] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 14829] getsockname(11, <unfinished ...>

[pid 16369] <... poll resumed>) = 0 (Timeout)

[pid 14829] <... getsockname resumed>{sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] getpeername(11, <unfinished ...>

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=2348815408, u64=139850778940464}}], 256, -1) = 1

[pid 14829] <... getpeername resumed>{sa\_family=AF\_INET, sin\_port=htons(60978), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_DEL, 9, 0x7f318c001434 <unfinished ...>

[pid 14829] fcntl(11, F\_GETFL <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... fcntl resumed>) = 0x2 (flags O\_RDWR)

[pid 16369] getsockopt(9, SOL\_SOCKET, SO\_ERROR, <unfinished ...>

[pid 14829] fcntl(11, F\_SETFL, O\_RDWR|O\_NONBLOCK <unfinished ...>

[pid 16369] <... getsockopt resumed>[0], [4]) = 0

[pid 14829] <... fcntl resumed>) = 0

[pid 16369] setsockopt(9, SOL\_TCP, TCP\_NODELAY, [1], 4 <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 16369] <... setsockopt resumed>) = 0

[pid 14829] <... getpid resumed>) = 14827

[pid 16369] getsockname(9, <unfinished ...>

[pid 14829] write(6, "\1\0\0\0\0\0\0\0", 8 <unfinished ...>

[pid 16369] <... getsockname resumed>{sa\_family=AF\_INET, sin\_port=htons(60978), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 14829] <... write resumed>) = 8

[pid 16369] getpeername(9, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... getpeername resumed>{sa\_family=AF\_INET, sin\_port=htons(5555), sin\_addr=inet\_addr("127.0.0.1")}, [128 => 16]) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=3283220064, u64=140736476608096}}], 256, -1) = 1

[pid 16369] fcntl(9, F\_GETFL <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 16369] <... fcntl resumed>) = 0x80802 (flags O\_RDWR|O\_NONBLOCK|O\_CLOEXEC)

[pid 14829] <... getpid resumed>) = 14827

[pid 16369] fcntl(9, F\_SETFL, O\_RDWR|O\_NONBLOCK|O\_CLOEXEC <unfinished ...>

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 16369] <... fcntl resumed>) = 0

[pid 14829] <... poll resumed>) = 1 ([{fd=6, revents=POLLIN}])

[pid 16369] getpid( <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 16369] <... getpid resumed>) = 16367

[pid 14829] <... getpid resumed>) = 14827

[pid 16369] write(6, "\1\0\0\0\0\0\0\0", 8 <unfinished ...>

[pid 14829] read(6, <unfinished ...>

[pid 16369] <... write resumed>) = 8

[pid 14829] <... read resumed>"\1\0\0\0\0\0\0\0", 8) = 8

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_ADD, 11, {events=0, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=3315906144, u64=140736509294176}}], 256, -1) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] getpid( <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... getpid resumed>) = 16367

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=59953300}) = 0

[pid 16369] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... poll resumed>) = 1 ([{fd=6, revents=POLLIN}])

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] getpid( <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN|EPOLLOUT, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... getpid resumed>) = 16367

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] read(6, <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... read resumed>"\1\0\0\0\0\0\0\0", 8) = 8

[pid 14829] <... recvfrom resumed>0x7ff4080117a8, 12, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_ADD, 9, {events=0, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] getpid( <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... getpid resumed>) = 14827

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=60541600}) = 0

[pid 14829] <... poll resumed>) = 0 (Timeout)

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=60713800}) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=134287488, u64=140686083035264}}], 256, 29999) = 1

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] sendto(11, "\377\0\0\0\0\0\0\0\1\177", 10, 0, NULL, 0 <unfinished ...>

[pid 16369] <... recvfrom resumed>0x7f318c001be8, 12, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 14829] <... sendto resumed>) = 10

[pid 16369] getpid( <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... getpid resumed>) = 16367

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] poll([{fd=6, events=POLLIN}], 1, 0 <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... poll resumed>) = 0 (Timeout)

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=61285700}) = 0

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=61375400}) = 0

[pid 16369] epoll\_wait(7, [{events=EPOLLIN|EPOLLOUT, data={u32=2348815408, u64=139850778940464}}], 256, 29999) = 1

[pid 16369] sendto(9, "\377\0\0\0\0\0\0\0\1\177", 10, 0, NULL, 0) = 10

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134287488, u64=140686083035264}}], 256, 29998) = 1

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>"\377\0\0\0\0\0\0\0\1\177", 12, 0, NULL, NULL) = 10

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN|EPOLLOUT, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... recvfrom resumed>"\377\0\0\0\0\0\0\0\1\177", 12, 0, NULL, NULL) = 10

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>0x7ff4080117b2, 2, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... recvfrom resumed>0x7f318c001bf2, 2, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=62218300}) = 0

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=62303400}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=134287488, u64=140686083035264}}], 256, 29997) = 1

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] sendto(11, "\3", 1, 0, NULL, 0 <unfinished ...>

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=2348815408, u64=139850778940464}}], 256, 29998) = 1

[pid 14829] <... sendto resumed>) = 1

[pid 16369] sendto(9, "\3", 1, 0, NULL, 0 <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... sendto resumed>) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=62753300}) = 0

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=62861200}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134287488, u64=140686083035264}}], 256, 29997) = 1

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=2348815408, u64=139850778940464}}], 256, 29998) = 1

[pid 14829] <... recvfrom resumed>"\3", 2, 0, NULL, NULL) = 1

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN|EPOLLOUT, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... recvfrom resumed>"\3", 2, 0, NULL, NULL) = 1

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>0x7ff4080117b3, 53, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... recvfrom resumed>0x7f318c001bf3, 53, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=63464000}) = 0

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=63565200}) = 0

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=134287488, u64=140686083035264}}], 256, 29996) = 1

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] sendto(11, "\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, 0 <unfinished ...>

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLOUT, data={u32=2348815408, u64=139850778940464}}], 256, 29997) = 1

[pid 14829] <... sendto resumed>) = 53

[pid 16369] sendto(9, "\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, 0 <unfinished ...>

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN, data={u32=134287488, u64=140686083035264}} <unfinished ...>

[pid 16369] <... sendto resumed>) = 53

[pid 14829] <... epoll\_ctl resumed>) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=64042200}) = 0

[pid 16369] clock\_gettime(CLOCK\_MONOTONIC, <unfinished ...>

[pid 14829] epoll\_wait(7, <unfinished ...>

[pid 16369] <... clock\_gettime resumed>{tv\_sec=7756, tv\_nsec=64124700}) = 0

[pid 16369] epoll\_wait(7, <unfinished ...>

[pid 14829] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=134287488, u64=140686083035264}}], 256, 29995) = 1

[pid 16369] <... epoll\_wait resumed>[{events=EPOLLIN, data={u32=2348815408, u64=139850778940464}}], 256, 29996) = 1

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] recvfrom(9, <unfinished ...>

[pid 14829] <... recvfrom resumed>"\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, NULL) = 53

[pid 16369] <... recvfrom resumed>"\1NULL\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 53, 0, NULL, NULL) = 53

[pid 14829] epoll\_ctl(7, EPOLL\_CTL\_MOD, 11, {events=EPOLLIN|EPOLLOUT, data={u32=134287488, u64=140686083035264}}) = 0

[pid 16369] epoll\_ctl(7, EPOLL\_CTL\_MOD, 9, {events=EPOLLIN|EPOLLOUT, data={u32=2348815408, u64=139850778940464}} <unfinished ...>

[pid 14829] recvfrom(11, <unfinished ...>

[pid 16369] <... epoll\_ctl resumed>) = 0

[pid 14829] <... recvfrom resumed>0x7ff408013f18, 8192, 0, NULL, NULL) = -1 EAGAIN (Resource temporarily unavailable)

[pid 16369] recvfrom(9, <unfinished ...>

**………………………………………………………………………………………………………**

**Вывод**

Во время выполнения лабораторной работы разобрался в создание и отладке распределенной системы. Самое сложное было осознать задачу и логику взаимодействий между узлами, так как из-за не очень полного и достаточно размытого описания некоторых ключевых аспектов работы возникали трудности с планированием этапов разработки, которые создавали ощущение “неправильного пути”. С этим я смог справится, сильнее углубившись в работу брокера очередей и типы сокетов, а также изучив работу клиент-сервис систем (поверхностно, но эффект был получен). По итогу, научился создавать распределенную систему обмена данным, а главное понимать как это делать правильно и какие средства для этого использовать (брокер сообщений и разные типы сокетов).