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: Голубев Т.Д.

Преподаватель: Миронов Е.С.

Оценка: ____________________

Дата: 28.12.2023

Москва, 2023

## Постановка задачи

**Вариант 6.**

Реализовать распределенную систему по асинхронной обработке запросов.

1. Топология 3 -- Все вычислительные узлы хранятся в бинарном дереве поиска. [parent] — является необязательным параметром.
2. Набор команд 4 (поиск подстроки в строке)
3. Команда проверки 1 (pingall)

## Общий метод и алгоритм решения

Использованные системные вызовы:

- int socket(int domain, int type, int protocol);  -- создает конечную точку соединения
- int connect(int sockfd, const struct sockaddr *serv_addr, socklen_t addrlen); -- инициирует соединение на сокете
- int bind(int sockfd, struct sockaddr *my_addr, socklen_t addrlen); -- привязать имя к сокету
- ssize_t send(int s, const void *msg, size_t len, int flags); -- отправляет сообщения в сокет
- int recv(int s, void *buf, size_t len, int flags); -- получить сообщение из сокета
- int setsockopt(int s, int level, int optname, const void *optval, socklen_t optlen); -- получить или установить флаги на сокете

Для реализации распределённой системы использовалась библиотека ZeroMQ.

## Код программы

### binary_tree.h

```
#pragma once

#include <compare>
#include <stdexcept>
#include <vector>

namespace mysys {
    template <class T>
    requires std::three_way_comparable<T>
    class BinaryTree {
    public:
        BinaryTree() : _root{nullptr} {}

        BinaryTree(const BinaryTree& other) {
            _root = _copy_tree(other->_root);
        }

        BinaryTree(BinaryTree&& other) noexcept {
            _root = other._root;
        }

        ~BinaryTree() noexcept {
            _delete_tree(_root);
        }

        BinaryTree& operator=(const BinaryTree& rhs) {
            _root = _copy_tree(rhs._root);
            return *this;
        }

        BinaryTree& operator=(BinaryTree&& rhs) noexcept {
            _root = rhs._root;
            return *this;
        }


        bool search(T key) const {
            _Node* found = _search(_root, key);

            if (found == nullptr) return false;

            return true;
        }

        void insert(T key) {
            if (search(key)) throw std::logic_error("Already exists");
            _root = _insert(_root, key);
        }

        std::vector<T> get_tops() {
            std::vector<T> tops;
            _get_tops(_root, tops);
            return tops;
        }

        std::vector<T> get_children(T key) {
            _Node* found = _search(_root, key);
            if (found == nullptr) throw std::logic_error("Key not found");
            std::vector<T> tops;
            _get_tops(found, tops);
            return tops;
        }

        T get_parent(T key) {
            _Node* found = _search_parent(_root, nullptr, key);
            if (found == nullptr) return 0;
            return found->key;
        }

    private:
        struct _Node {
            T key;
            _Node* left;
            _Node* right;
        };

        _Node* _root;

        _Node* _new_node(T item) {
            _Node* temp = new _Node;
            temp->key = item;
```

```cpp
            temp->left = temp->right = nullptr;
            return temp;
        }

        _Node* _insert(_Node* node, T key) {
            if (node == nullptr)
                return _new_node(key);

            if (key < node->key)
                node->left = _insert(node->left, key);
            else if (key > node->key)
                node->right = _insert(node->right, key);

            return node;
        }

        void _delete_tree(_Node* node) noexcept {
            if (node == nullptr) return;
            _Node* left = node->left;
            _Node* right = node->right;
            delete node;
            _delete_tree(left);
            _delete_tree(right);
        }

        _Node* _create_tree(_Node* left, _Node* right, T key) {
            _Node* root = _new_node(key);
            root->left = left;
            root->right = right;
            return root;
        }

        _Node* _copy_tree(_Node* other) {
            if (other == nullptr) return nullptr;
            _Node* root = _new_node(other->key);
            root->left = _copy_tree(other->left);
            root->right = _copy_tree(other->right);
            return root;
        }

        _Node* _search(_Node* root, int key) const {
            if (root == nullptr || root->key == key)
                return root;

            if (root->key < key)
                return _search(root->right, key);

            return _search(root->left, key);
        }

        _Node* _search_parent(_Node* root, _Node* prev, int key) const {
            if (root == nullptr || root->key == key)
                return prev;
            if (root->key < key)
                return _search_parent(root->right, root, key);
            return _search_parent(root->left, root, key);
        }

        void _get_tops(_Node* root, std::vector<T>& tops) {
            if (root == nullptr) return;
            _get_tops(root->left, tops);
            _get_tops(root->right, tops);
            tops.push_back(root->key);
        }
    };
}
```

## calculating_node.h

```
#pragma once

#include <zmq.hpp>
#include <string>
#include <vector>
#include <map>
#include <array>
#include "message_type.h"

namespace mysys {
    class CalculatingNode {
    public:
        CalculatingNode(int id, int base_port);
        CalculatingNode(const CalculatingNode& other) = delete;
        CalculatingNode(CalculatingNode&& other) noexcept;
        ~CalculatingNode() noexcept;
        void connect_child(int child_id);
        std::vector<int> ping_children();      // returns id of unavailable child
        MyMessage get_child_msg(zmq::socket_t* child);
        MyMessage get_parent_msg();
        bool req(zmq::socket_t* child, const MyMessage& msg);
        void reply(const MyMessage& msg);
        int id() const;
        std::array<std::pair<int, zmq::socket_t*>, 2> children() const;
        std::vector<int> _string_to_vector(const std::string& str);
        std::string exec(const std::string& text, const std::string& pattern);
        zmq::socket_t* get_less_child() const;
        zmq::socket_t* get_greater_child() const;
    private:
        zmq::context_t _context;
        zmq::socket_t _s_parent;     // like server (to parent)
        std::array<std::pair<int, zmq::socket_t*>, 2> _s_children;
        int _base_port;
        int _id;

        void _msg_to_string(const zmq::message_t& msg, std::string& str);
    };
}
```

## calculating_node.cpp

```
#include "calculating_node.h"

using namespace mysys;

std::vector<int> PrefixFunction(const std::string& s) {
        unsigned int n = s.size();
        std::vector<int> p(n);
        for (int i = 1; i < n; ++i) {
                p[i] = p[i - 1];
                while (p[i] > 0 and s[i] != s[p[i]]) {
                        p[i] = p[p[i] - 1];
                }
                if (s[i] == s[p[i]]) {
                        ++p[i];
                }
        }
        return p;
}

std::vector<int> KMPWeak(const std::string& text, const std::string& pattern) {
        std::vector<int> p = PrefixFunction(pattern);
        int m = pattern.size();
        int n = text.size();
        int i = 0;
        std::vector<int> ans;
        if (m > n) {
                return ans;
        }
        while (i < n - m + 1) {
                int j = 0;
                while (j < m and pattern[j] == text[i + j]) {
                        ++j;
                }
                if (j == m) {
                        ans.push_back(i);
                } else {
                        if (j > 0 and j > p[j - 1]) {
                                i = i + j - p[j - 1] - 1;
                        }
                }
                ++i;
        }
        return ans;
}

CalculatingNode::CalculatingNode(int id, int base_port) :
_id{id},
_base_port{base_port},
_s_parent(_context, zmq::socket_type::pair) {
        _s_parent.set(zmq::sockopt::sndtimeo, 3000);
        std::string addr = "tcp://localhost:" + std::to_string(_base_port + _id);
        _s_parent.connect(addr);
        _s_children[0] = std::make_pair(-1, nullptr);
        _s_children[1] = std::make_pair(-1, nullptr);
}

CalculatingNode::CalculatingNode(CalculatingNode&& other) noexcept {
        _context = std::move(other._context);
        _s_parent = std::move(other._s_parent);
        _s_children = std::move(other._s_children);
        _base_port = std::move(_base_port);
        _id = std::move(other._id);
}

CalculatingNode::~CalculatingNode() noexcept {
        for (auto& p : _s_children) {
                if (p.second == nullptr) continue;
                delete p.second;
        }
}

bool CalculatingNode::req(zmq::socket_t* child, const MyMessage& msg) {
        zmq::message_t message_type(std::to_string(msg.type));
        if (msg.type == MessageType::ping) {
                auto res = child->send(message_type, zmq::send_flags::dontwait);
                if (!res) {
                        return false;
                }
                return true;
        }
        auto res = child->send(message_type, zmq::send_flags::sndmore);
        zmq::message_t message_text(msg.text);
        res = child->send(message_text, zmq::send_flags::none);
        return true;
}

void CalculatingNode::reply(const MyMessage& msg) {
        zmq::message_t message_type(std::to_string((int) msg.type));
        auto res = _s_parent.send(message_type, zmq::send_flags::sndmore);
        zmq::message_t message_text(msg.text);
        res = _s_parent.send(message_text, zmq::send_flags::none);
}

void CalculatingNode::connect_child(int child_id) {
        zmq::socket_t* child = new zmq::socket_t(_context, zmq::socket_type::pair);
        std::string addr = "tcp://*:" + std::to_string(_base_port + child_id);
        child->bind(addr);
        if (child_id < _id) {
                _s_children[0] = std::make_pair(child_id, child);
        } else {
                _s_children[1] = std::make_pair(child_id, child);
        }
}

void CalculatingNode::_msg_to_string(const zmq::message_t& msg, std::string& str)
{
        str.resize(msg.size() / sizeof(char));
    std::memcpy(str.data(), msg.data(), msg.size());
}

MyMessage CalculatingNode::get_child_msg(zmq::socket_t* child) {
        MyMessage msg;
        zmq::message_t msg_type;
        auto res = child->recv(msg_type);
        std::string buf;
        _msg_to_string(msg_type, buf);
        msg.type = (MessageType) std::stoi(buf);
        zmq::message_t msg_text;
        res = child->recv(msg_text);
        _msg_to_string(msg_text, buf);
        msg.text = buf;

        return msg;
}

MyMessage CalculatingNode::get_parent_msg() {
        MyMessage msg;
        zmq::message_t msg_type;
        auto res = _s_parent.recv(msg_type);
        std::string buf;
        _msg_to_string(msg_type, buf);
        msg.type = (MessageType) std::stoi(buf);
        if (msg.type == MessageType::ping || msg.type == MessageType::shutdown) re-
turn msg;
        zmq::message_t msg_text;
        res = _s_parent.recv(msg_text);
        _msg_to_string(msg_text, buf);
        msg.text = buf;

```

```cpp
        return msg;
}

std::vector<int> CalculatingNode::_string_to_vector(const std::string& str) {
    std::stringstream ss(str);
    std::vector<int> vec;
    int num;
    while (ss >> num) {
        vec.push_back(num);
    }
    return vec;
}

std::vector<int> CalculatingNode::ping_children() {
        std::string s = "";
        std::vector<int> ids;
        for (auto& p : _s_children) {
                if (p.second == nullptr) continue;
                MyMessage msg;
                msg.type = MessageType::ping;
                bool res = req(p.second, msg);
                if (!res) {
                        s += std::to_string(p.first) + ' ';
                        continue;
                }
                msg = get_child_msg(p.second);
                s += msg.text + ' ';
        }
        ids = _string_to_vector(s);
        return ids;
}

int CalculatingNode::id() const {
        return _id;
}

std::array<std::pair<int, zmq::socket_t*>, 2> CalculatingNode::children() const {
        return _s_children;
}

std::string CalculatingNode::exec(const std::string& text, const std::string& pat-
tern) {
        std::vector<int> idxs = KMPWeak(text, pattern);
        std::string res = "";
        for (auto idx : idxs) {
                res += std::to_string(idx) + ' ';
        }
        return res;
}

zmq::socket_t* CalculatingNode::get_less_child() const {
        return _s_children[0].second;
}

zmq::socket_t* CalculatingNode::get_greater_child() const {
        return _s_children[1].second;
}
```

## control_node.h

```cpp
#pragma once

```

```cpp
#include <zmq.hpp>
#include <unistd.h>
#include <vector>
#include <string>
#include <fstream>
#include "message_type.h"
#include "binary_tree.h"

namespace mysys {
    class ControlNode {
    public:
        ControlNode(int base_port = 5000);
        ControlNode(const ControlNode& other) = delete;
        ControlNode(ControlNode&& other) noexcept;
        ~ControlNode() noexcept;
        pid_t new_node(int id);
        std::vector<int> pingall();
        std::string exec(int id, const std::string& text, const std::string& pat-
tern);
        MyMessage get_message(zmq::recv_flags flags = zmq::recv_flags::none);
        bool send_message(const MyMessage& msg);
    private:
        zmq::context_t _context;
        zmq::socket_t _s_request;
        int _base_port;
        BinaryTree<int> _topology;
        bool _has_child;

        std::vector<int> _string_to_vector(const std::string& str);
        void _msg_to_string(const zmq::message_t& msg, std::string& str);
    };
}
```

control_node.cpp

```cpp
#include "control_node.h"
#include <iostream>

using namespace mysys;

void ControlNode::_msg_to_string(const zmq::message_t& msg, std::string& str) {
    str.resize(msg.size() / sizeof(char));
    std::memcpy(str.data(), msg.data(), msg.size());
}

template<typename Item>
void concat(std::vector<Item> &a, std::vector<Item> &b) {
    a.reserve(a.size() + b.size());
    a.insert(
        a.end(),
        std::make_move_iterator(b.begin()),
        std::make_move_iterator(b.end())
    );
}

ControlNode::ControlNode(int base_port) : _base_port{base_port},
    _s_request(_context, zmq::socket_type::pair), _has_child{false} {
    _s_request.set(zmq::sockopt::sndtimeo, 3000);
}

ControlNode::ControlNode(ControlNode&& other) noexcept {
    _context = std::move(other._context);
    _s_request = std::move(other._s_request);
    _base_port = std::move(other._base_port);
    _topology = std::move(other._topology);
}

ControlNode::~ControlNode() noexcept {}

pid_t ControlNode::new_node(int id) {
    if (id == 0) throw std::logic_error("id 0 is reserved for server");
    _topology.insert(id);
    int parent = _topology.get_parent(id);
    pid_t pid = fork();
    if (pid == 0) {
        execl("./lab5-7_calc", "./lab5-7_calc", std::to_string(id).c_str(), std::to_string(_base_port).c_str());
    } else {
        if (!_has_child) {
            std::string addr = "tcp://*:" + std::to_string(_base_port + id);
            _s_request.bind(addr);
            _has_child = true;
        } else {
            MyMessage bind;
            bind.type = MessageType::bind_node;
            bind.text = std::to_string(parent) + " " + std::to_string(id);
            send_message(bind);
        }
        return pid;
    }
}

std::vector<int> ControlNode::_string_to_vector(const std::string& str) {
    std::stringstream ss(str);
    std::vector<int> vec;
    int num;
    while (ss >> num) {
        vec.push_back(num);
    }
    return vec;
}

MyMessage ControlNode::get_message(zmq::recv_flags flags) {
    MyMessage msg;
    std::string buf;
    zmq::message_t rec;
    auto res = _s_request.recv(rec, flags);
    _msg_to_string(rec, buf);
    msg.type = (MessageType) std::stoi(buf);
    res = _s_request.recv(rec);
    _msg_to_string(rec, buf);
    msg.text = buf;
    return msg;
}

bool ControlNode::send_message(const MyMessage& msg) {
    zmq::message_t msg_type(std::to_string(msg.type));
    if (msg.type == MessageType::ping || msg.type == MessageType::shutdown) {
        auto res = _s_request.send(msg_type, zmq::send_flags::dontwait);
        if (!res) return false;
        return true;
    }
    _s_request.send(msg_type, zmq::send_flags::sndmore);
    zmq::message_t msg_text(msg.text);
    _s_request.send(msg_text, zmq::send_flags::none);
    return true;
```

```cpp
}

std::vector<int> ControlNode::pingall() {
    MyMessage msg;
    msg.type = MessageType::ping;
    if (!send_message(msg)) {
        std::vector<int> ids = _topology.get_tops();
        return ids;
    }
    msg = get_message();
    std::vector<int> ids = _string_to_vector(msg.text);
    std::vector<int> tops;
    for (auto el : ids) {
        std::vector<int> children = _topology.get_children(el);
        concat<int>(tops, children);
    }
    return tops;
}

std::string ControlNode::exec(int id, const std::string& text, const std::string&
pattern) {
    MyMessage msg;
    msg.type = MessageType::exec;
    msg.text = std::to_string(id) + ' ' + text + ' ' + pattern;
    send_message(msg);
    msg = get_message();
    if (msg.type != MessageType::exec_result) throw std::runtime_error("Wrong mes-
sage type");
    return msg.text;
}
```

## message_type.h

```cpp
#pragma once

#include <string>
#include <zmq.hpp>

namespace mysys {
    enum MessageType {
        exec,
        ping,
        bind_node,
        exec_result,
        ping_result,
        error,
        shutdown
    };

    struct MyMessage {
        MessageType type;
        std::string text;
    };
}
```

## calc_main.cpp

```cpp
#include "calculating_node.h"

using namespace mysys;

std::string vector_to_string(const std::vector<int>& v) {
    std::string str = "";
    for (auto el : v) {
        str += std::to_string(el) + ' ';
    }
    return str;
}

int main(int argc, char** argv) {
    CalculatingNode node(std::stoi(argv[1]), std::stoi(argv[2]));
    while (true) {
        MyMessage msg = node.get_parent_msg();
        if (msg.type == MessageType::ping) {
            std::vector<int> ids = node.ping_children();
            MyMessage rep;
            rep.type = MessageType::ping_result;
            rep.text = vector_to_string(ids);
            node.reply(rep);
        } else if (msg.type == MessageType::bind_node) {
            int parent, id;
            std::stringstream ss(msg.text);
            ss >> parent >> id;
            if (parent == node.id()) {
                node.connect_child(id);
            } else if (parent < node.id()) {
                node.req(node.children()[0].second, msg);
            } else {
                node.req(node.children()[1].second, msg);
            }
        } else if (msg.type == MessageType::exec) {
            int id;
            std::stringstream ss(msg.text);
            ss >> id;
            if (id == node.id()) {
                std::string text, pattern;
                ss >> text >> pattern;
                MyMessage reply;
                reply.text = node.exec(text, pattern);
                reply.type = MessageType::exec_result;
                node.reply(reply);
            } else {
                MyMessage next;
                next.type = MessageType::exec;
                next.text = msg.text;
                if (id < node.id()) {
                    node.req(node.get_less_child(), next);
                    next = node.get_child_msg(node.get_less_child());
                } else {
                    node.req(node.get_greater_child(), next);
                    next = node.get_child_msg(node.get_greater_child());
                }
                node.reply(next);
            }
        } else if (msg.type == MessageType::exec_result) {
            MyMessage next;
            next.type = MessageType::exec_result;
            next.text = msg.text;
            node.reply(next);
        } else if (msg.type == MessageType::shutdown) {
            MyMessage next;
            next.type = MessageType::shutdown;
            if (node.get_less_child() != nullptr) node.req(node.get_less_child(),
next);
            if (node.get_greater_child() != nullptr)
node.req(node.get_greater_child(), next);
```

```
                break;
            }
        }

        return 0;
    }
```

main.cpp

```
#include <iostream>
#include <sstream>
#include "control_node.h"

using namespace mysys;

std::vector<int> string_to_vector(const std::string& str) {
    std::stringstream ss(str);
    std::vector<int> vec;
    int num;
    while (ss >> num) {
        vec.push_back(num);
    }
    return vec;
}

int main() {
    ControlNode ctrl;
    std::cout << "Starting Control Node" << std::endl;
    std::string cmd;
    while(true) {
        std::cout << "> ";
        std::cin >> cmd;
        if (cmd == "create") {
            int id;
            std::cin >> id;
            pid_t pid;
            try {
                pid = ctrl.new_node(id);
            } catch (std::exception& e) {
                std::cout << e.what() << std::endl;
                continue;
            }
            std::cout << "Ok: " << pid << std::endl;
        } else if (cmd == "pingall") {
            std::vector<int> ids = ctrl.pingall();
            std::cout << "Ok: ";
            if (ids.size() == 0) {
                std::cout << "-1";
            }
            for (auto id : ids) {
                std::cout << id << ";";
            }
            std::cout << std::endl;
        } else if (cmd == "q") {
            MyMessage msg;
            msg.type = MessageType::shutdown;
            ctrl.send_message(msg);
            break;
        } else if (cmd == "exec") {
            int id;
            std::cin >> id;
            std::string text_str, pattern_str;
            std::cin >> text_str >> pattern_str;
            std::string res = ctrl.exec(id, text_str, pattern_str);
            std::cout << res << std::endl;
        } else {
            std::cout << "Wrong command!" << std::endl;
        }
    }

    return 0;
}
```

## Протокол работы программы

**Strace:**

```
execve("./lab5-7_exe", ["./lab5-7_exe"], 0x7ffda5c591e0 /* 27 vars */) = 0

brk(NULL)                               = 0x562f286dd000

arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe5d1aae50) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 
0x7f3ded59b000

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=32515, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 32515, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f3ded593000

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\0\0\0\0\0\0\0\0"..., 832) = 
832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=583920, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 585912, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded503000

mmap(0x7f3ded51b000, 364544, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 
3, 0x18000) = 0x7f3ded51b000

mmap(0x7f3ded574000, 90112, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 
0x71000) = 0x7f3ded574000

mmap(0x7f3ded58a000, 36864, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 
3, 0x86000) = 0x7f3ded58a000

close(3)                                = 0

openat(AT_FDCWD, "/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\0\0\0\0\0\0\0\0"..., 832) = 
832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2260296, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 2275520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded2d7000

mprotect(0x7f3ded371000, 1576960, PROT_NONE) = 0
```

```
     mmap(0x7f3ded371000, 1118208, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9a000) = 0x7f3ded371000

     mmap(0x7f3ded482000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1ab000) = 0x7f3ded482000

     mmap(0x7f3ded4f2000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x21a000) = 0x7f3ded4f2000

     mmap(0x7f3ded500000, 10432, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f3ded500000

     close(3)                                = 0

     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\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded2b7000

     mmap(0x7f3ded2ba000, 94208, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x3000) = 0x7f3ded2ba000

     mmap(0x7f3ded2d1000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1a000) = 0x7f3ded2d1000

     mmap(0x7f3ded2d5000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1d000) = 0x7f3ded2d5000

     close(3)                                = 0

     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\0P\237\2\0\0\0\0\0"..., 832) =
832

     pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784,
64) = 784

     pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48,
848) = 48

     pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0
=\340\2563\265?\356\25x\261\27\313A#\350"..., 68, 896) = 68

     newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2216304, ...}, AT_EMPTY_PATH) = 0

     pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784,
64) = 784

     mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded08f000

     mmap(0x7f3ded0b7000, 1658880, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f3ded0b7000

     mmap(0x7f3ded24c000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1bd000) = 0x7f3ded24c000

     mmap(0x7f3ded2a4000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x214000) = 0x7f3ded2a4000

     mmap(0x7f3ded2aa000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f3ded2aa000

     close(3)                                = 0
     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libunwind.so.8", 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\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=63744, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 109264, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded074000

     mmap(0x7f3ded076000, 40960, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x2000) = 0x7f3ded076000

     mmap(0x7f3ded080000, 12288, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xc000)
= 0x7f3ded080000

     mmap(0x7f3ded083000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xe000) = 0x7f3ded083000

     mmap(0x7f3ded085000, 39632, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f3ded085000

     close(3)                                = 0

     openat(AT_FDCWD, "/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\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=355040, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 357440, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3ded01c000

     mprotect(0x7f3ded028000, 303104, PROT_NONE) = 0

     mmap(0x7f3ded028000, 229376, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xc000) = 0x7f3ded028000

     mmap(0x7f3ded060000, 69632, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x44000) = 0x7f3ded060000

     mmap(0x7f3ded072000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x55000) = 0x7f3ded072000

     close(3)                                = 0

     mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3ded01a000

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libpgm-5.3.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\0\340L\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=310264, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 329808, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decfc9000

     mmap(0x7f3decfcd000, 172032, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4000) = 0x7f3decfcd000

     mmap(0x7f3decff7000, 118784, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x2e000) = 0x7f3decff7000

     mmap(0x7f3ded014000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4a000) = 0x7f3ded014000

     mmap(0x7f3ded016000, 14416, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f3ded016000
```

```
     close(3)                                = 0

     openat(AT_FDCWD, "/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\0 \255\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=497824, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 1223168, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3dece9e000

     mprotect(0x7f3decea8000, 446464, PROT_NONE) = 0

     mmap(0x7f3decea8000, 286720, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xa000) = 0x7f3decea8000

     mmap(0x7f3deceee000, 155648, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x50000) = 0x7f3deceee000

     mmap(0x7f3decf15000, 16384, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x76000) = 0x7f3decf15000

     mmap(0x7f3decf19000, 719360, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f3decf19000

     close(3)                                = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgssapi_krb5.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\0\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=338648, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 340960, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3dece4a000

     mprotect(0x7f3dece55000, 282624, PROT_NONE) = 0

     mmap(0x7f3dece55000, 229376, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xb000) = 0x7f3dece55000

     mmap(0x7f3dece8d000, 49152, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x43000) = 0x7f3dece8d000

     mmap(0x7f3dece9a000, 16384, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4f000) = 0x7f3dece9a000

     close(3)                                = 0

     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\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decd63000

     mmap(0x7f3decd71000, 507904, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xe000) = 0x7f3decd71000

     mmap(0x7f3decded000, 372736, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x8a000) = 0x7f3decded000

     mmap(0x7f3dece48000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xe4000) = 0x7f3dece48000

     close(3)                                = 0
     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/liblzma.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\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=170456, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 172296, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decd38000

     mmap(0x7f3decd3b000, 110592, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x3000) = 0x7f3decd3b000

     mmap(0x7f3decd56000, 45056, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1e000) = 0x7f3decd56000

     mmap(0x7f3decd61000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x28000) = 0x7f3decd61000

     close(3)                                = 0

     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\0\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=21448, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3decd36000

     mmap(NULL, 16424, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decd31000

     mmap(0x7f3decd32000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x1000) = 0x7f3decd32000

     mmap(0x7f3decd33000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000)
= 0x7f3decd33000

     mmap(0x7f3decd34000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x2000) = 0x7f3decd34000

     close(3)                                = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libkrb5.so.3", 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\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=827936, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 830576, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc66000

     mprotect(0x7f3decc87000, 634880, PROT_NONE) = 0

     mmap(0x7f3decc87000, 380928, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x21000) = 0x7f3decc87000

     mmap(0x7f3decce4000, 249856, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x7e000) = 0x7f3decce4000

     mmap(0x7f3decd22000, 61440, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xbb000) = 0x7f3decd22000

     close(3)                                = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libk5crypto.so.3", 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\0"..., 832) =
832
```

```
    newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=182864, ...}, AT_EMPTY_PATH) = 0

    mmap(NULL, 188472, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc37000

    mprotect(0x7f3decc3b000, 163840, PROT_NONE) = 0

    mmap(0x7f3decc3b000, 110592, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4000) = 0x7f3decc3b000

    mmap(0x7f3decc56000, 49152, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1f000) = 0x7f3decc56000

    mmap(0x7f3decc63000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x2b000) = 0x7f3decc63000

    mmap(0x7f3decc65000, 56, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -
1, 0) = 0x7f3decc65000

    close(3)                                = 0

    openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libcom_err.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\0\0\0\0\0\0\0\0\0"..., 832) =
832

    newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=18504, ...}, AT_EMPTY_PATH) = 0

    mmap(NULL, 20552, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc31000

    mmap(0x7f3decc33000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x2000) = 0x7f3decc33000

    mmap(0x7f3decc34000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000)
= 0x7f3decc34000

    mmap(0x7f3decc35000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x3000) = 0x7f3decc35000

    close(3)                                = 0

    openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libkrb5support.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\0\0\0\0\0\0\0\0\0"..., 832) =
832

    newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=52016, ...}, AT_EMPTY_PATH) = 0

    mmap(NULL, 54224, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc23000

    mprotect(0x7f3decc26000, 36864, PROT_NONE) = 0

    mmap(0x7f3decc26000, 24576, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x3000) = 0x7f3decc26000

    mmap(0x7f3decc2c000, 8192, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9000)
= 0x7f3decc2c000

    mmap(0x7f3decc2f000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xb000) = 0x7f3decc2f000

    close(3)                                = 0

    openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libkeyutils.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\0"..., 832) =
832

    newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=22600, ...}, AT_EMPTY_PATH) = 0
```

```
     mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3decc21000

     mmap(NULL, 24592, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc1a000

     mmap(0x7f3decc1c000, 8192, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x2000) = 0x7f3decc1c000

     mmap(0x7f3decc1e000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x4000)
= 0x7f3decc1e000

     mmap(0x7f3decc1f000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x4000) = 0x7f3decc1f000

     close(3)                                = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libresolv.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\0\0\0\0\0\0\0\0\0"..., 832) =
832

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=68552, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 80456, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3decc06000

     mmap(0x7f3decc09000, 40960, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x3000) = 0x7f3decc09000

     mmap(0x7f3decc13000, 12288, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xd000)
= 0x7f3decc13000

     mmap(0x7f3decc16000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0xf000) = 0x7f3decc16000

     mmap(0x7f3decc18000, 6728, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS,
-1, 0) = 0x7f3decc18000

     close(3)                                = 0

     mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3decc04000

     mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3decc01000

     arch_prctl(ARCH_SET_FS, 0x7f3decc019c0) = 0

     set_tid_address(0x7f3decc01c90)         = 13203

     set_robust_list(0x7f3decc01ca0, 24)     = 0

     rseq(0x7f3decc02360, 0x20, 0, 0x53053053) = 0

     mprotect(0x7f3ded2a4000, 16384, PROT_READ) = 0

     mprotect(0x7f3decc16000, 4096, PROT_READ) = 0

     mprotect(0x7f3decc1f000, 4096, PROT_READ) = 0

     mprotect(0x7f3decc2f000, 4096, PROT_READ) = 0

     mprotect(0x7f3decc35000, 4096, PROT_READ) = 0

     mprotect(0x7f3decc63000, 4096, PROT_READ) = 0

     mprotect(0x7f3decd22000, 53248, PROT_READ) = 0

     mprotect(0x7f3decd34000, 4096, PROT_READ) = 0
     mprotect(0x7f3decd61000, 4096, PROT_READ) = 0

     mprotect(0x7f3dece48000, 4096, PROT_READ) = 0

     mprotect(0x7f3dece9a000, 8192, PROT_READ) = 0

     mprotect(0x7f3ded2d5000, 4096, PROT_READ) = 0

     mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f3decbff000

     mprotect(0x7f3ded4f2000, 45056, PROT_READ) = 0

     mprotect(0x7f3decf15000, 12288, PROT_READ) = 0

     mprotect(0x7f3ded014000, 4096, PROT_READ) = 0

     mprotect(0x7f3ded072000, 4096, PROT_READ) = 0

     mprotect(0x7f3ded083000, 4096, PROT_READ) = 0

     mprotect(0x7f3ded58a000, 32768, PROT_READ) = 0

     mprotect(0x562f26c92000, 4096, PROT_READ) = 0

     mprotect(0x7f3ded5d5000, 8192, PROT_READ) = 0

     prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

     munmap(0x7f3ded593000, 32515)           = 0

     getrandom("\xc7\x7b\x4f\x88\xaa\xce\xa4\xba", 8, GRND_NONBLOCK) = 8

     brk(NULL)                               = 0x562f286dd000

     brk(0x562f286fe000)                     = 0x562f286fe000

     futex(0x7f3ded50077c, FUTEX_WAKE_PRIVATE, 2147483647) = 0

     openat(AT_FDCWD, "/sys/devices/system/cpu/online", O_RDONLY|O_CLOEXEC) = 3

     read(3, "0-15\n", 1024)                 = 5

     close(3)                                = 0

     openat(AT_FDCWD, "/sys/devices/system/cpu", O_RDONLY|O_NONBLOCK|O_CLOEXEC|O_DIRECTORY)
= 3

     newfstatat(3, "", {st_mode=S_IFDIR|0755, st_size=0, ...}, AT_EMPTY_PATH) = 0

     getdents64(3, 0x562f286eeee0 /* 31 entries */, 32768) = 896

     getdents64(3, 0x562f286eeee0 /* 0 entries */, 32768) = 0

     close(3)                                = 0

     getpid()                                = 13203

     sched_getaffinity(13203, 128, [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15])
= 32

     newfstatat(AT_FDCWD, "/etc/nsswitch.conf", {st_mode=S_IFREG|0644, st_size=510, ...},
0) = 0

     newfstatat(AT_FDCWD, "/", {st_mode=S_IFDIR|0755, st_size=4096, ...}, 0) = 0

     openat(AT_FDCWD, "/etc/nsswitch.conf", O_RDONLY|O_CLOEXEC) = 3

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=510, ...}, AT_EMPTY_PATH) = 0
     read(3, "# /etc/nsswitch.conf\n#\n# Example"..., 4096) = 510

     read(3, "", 4096)                          = 0

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=510, ...}, AT_EMPTY_PATH) = 0

     close(3)                                   = 0

     openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=32515, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 32515, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f3ded593000

     close(3)                                   = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v3/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v3", 0x7ffe5d1a7e50,
0) = -1 ENOENT (No such file or directory)

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v2/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v2", 0x7ffe5d1a7e50,
0) = -1 ENOENT (No such file or directory)

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64/x86_64/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -
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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/tls", 0x7ffe5d1a7e50, 0) = -1 ENOENT (No
such file or directory)

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/x86_64/x86_64/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64-linux-gnu", {st_mode=S_IFDIR|0755,
st_size=36864, ...}, 0) = 0

     openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v3/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v3",
0x7ffe5d1a7e50, 0) = -1 ENOENT (No such file or directory)

     openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v2/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/glibc-hwcaps/x86-64-v2",
0x7ffe5d1a7e50, 0) = -1 ENOENT (No such file or directory)

     openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/tls/x86_64/x86_64/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/tls/x86_64/x86_64", 0x7ffe5d1a7e50, 0)
= -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/tls", 0x7ffe5d1a7e50, 0) = -1 ENOENT
(No such file or directory)

     openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/x86_64/x86_64/libnss_db.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -
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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu", {st_mode=S_IFDIR|0755,
st_size=36864, ...}, 0) = 0

     openat(AT_FDCWD, "/lib/glibc-hwcaps/x86-64-v3/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -
1 ENOENT (No such file or directory)
```

```
     newfstatat(AT_FDCWD, "/lib/glibc-hwcaps/x86-64-v3", 0x7ffe5d1a7e50, 0) = -1 ENOENT (No
such file or directory)

     openat(AT_FDCWD, "/lib/glibc-hwcaps/x86-64-v2/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -
1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/glibc-hwcaps/x86-64-v2", 0x7ffe5d1a7e50, 0) = -1 ENOENT (No
such file or directory)

     openat(AT_FDCWD, "/lib/tls/x86_64/x86_64/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -1
ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/lib/tls/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/tls", 0x7ffe5d1a7e50, 0) = -1 ENOENT (No such file or
directory)

     openat(AT_FDCWD, "/lib/x86_64/x86_64/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -1 ENOENT
(No such file or directory)

     newfstatat(AT_FDCWD, "/lib/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/lib", {st_mode=S_IFDIR|0755, st_size=12288, ...}, 0) = 0

     openat(AT_FDCWD, "/usr/lib/glibc-hwcaps/x86-64-v3/libnss_db.so.2", O_RDONLY|O_CLOEXEC)
= -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/glibc-hwcaps/x86-64-v3", 0x7ffe5d1a7e50, 0) = -1 ENOENT
(No such file or directory)

     openat(AT_FDCWD, "/usr/lib/glibc-hwcaps/x86-64-v2/libnss_db.so.2", O_RDONLY|O_CLOEXEC)
= -1 ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/glibc-hwcaps/x86-64-v2", 0x7ffe5d1a7e50, 0) = -1 ENOENT
(No such file or directory)
```

```
     openat(AT_FDCWD, "/usr/lib/tls/x86_64/x86_64/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -1
ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/tls/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/tls/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/tls", 0x7ffe5d1a7e50, 0) = -1 ENOENT (No such file or
directory)

     openat(AT_FDCWD, "/usr/lib/x86_64/x86_64/libnss_db.so.2", O_RDONLY|O_CLOEXEC) = -1
ENOENT (No such file or directory)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib/x86_64", 0x7ffe5d1a7e50, 0) = -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)

     newfstatat(AT_FDCWD, "/usr/lib", {st_mode=S_IFDIR|0755, st_size=12288, ...}, 0) = 0

     munmap(0x7f3ded593000, 32515)           = 0

     openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=32515, ...}, AT_EMPTY_PATH) = 0

     mmap(NULL, 32515, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f3ded593000

     close(3)                                = 0

     openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libnss_db-2.35.so", O_RDONLY|O_CLOEXEC) = -1
ENOENT (No such file or directory)

     openat(AT_FDCWD, "/usr/lib/x86_64-linux-gnu/libnss_db-2.35.so", O_RDONLY|O_CLOEXEC) =
-1 ENOENT (No such file or directory)

     openat(AT_FDCWD, "/lib/libnss_db-2.35.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such
file or directory)

     openat(AT_FDCWD, "/usr/lib/libnss_db-2.35.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No
such file or directory)
```

```
     munmap(0x7f3ded593000, 32515)           = 0

     openat(AT_FDCWD, "/etc/protocols", O_RDONLY|O_CLOEXEC) = 3

     newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2932, ...}, AT_EMPTY_PATH) = 0

     lseek(3, 0, SEEK_SET)                   = 0

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

     read(3, "", 4096)                       = 0

     close(3)                                = 0

     eventfd2(0, EFD_CLOEXEC)                = 3

     fcntl(3, F_GETFL)                       = 0x2 (flags O_RDWR)

     fcntl(3, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     fcntl(3, F_GETFL)                       = 0x802 (flags O_RDWR|O_NONBLOCK)

     fcntl(3, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     getpid()                                = 13203

     getpid()                                = 13203

     getrandom("\x6c\x5c\xe2\x32\x30\xe6\xf2\x1b\xdc\x2d\xac\xd8\x40\xe0\x20\x73", 16, 0) =
16

     getrandom("\xfb\x4c\x41\xdf\x15\xf6\x1d\x18\x9b\x44\x17\x2f\xd9\x93\xd8\x6b", 16, 0) =
16

     eventfd2(0, EFD_CLOEXEC)                = 4

     fcntl(4, F_GETFL)                       = 0x2 (flags O_RDWR)

     fcntl(4, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     fcntl(4, F_GETFL)                       = 0x802 (flags O_RDWR|O_NONBLOCK)

     fcntl(4, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     getpid()                                = 13203

     epoll_create1(EPOLL_CLOEXEC)            = 5

     epoll_ctl(5, EPOLL_CTL_ADD, 4, {events=0, data={u32=678359648, u64=94760541811296}}) =
0

     epoll_ctl(5, EPOLL_CTL_MOD, 4, {events=EPOLLIN, data={u32=678359648,
u64=94760541811296}}) = 0

     getpid()                                = 13203

     rt_sigaction(SIGRT_1, {sa_handler=0x7f3ded120870, sa_mask=[],
sa_flags=SA_RESTORER|SA_ONSTACK|SA_RESTART|SA_SIGINFO, sa_restorer=0x7f3ded0d1520}, NULL, 8)
= 0

     rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0

     mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7f3dec3fe000

     mprotect(0x7f3dec3ff000, 8388608, PROT_READ|PROT_WRITE) = 0

     rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
     clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|C
LONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, child_tid=0x7f3decbfe910,
parent_tid=0x7f3decbfe910, exit_signal=0, stack=0x7f3dec3fe000, stack_size=0x7ffc80,
tls=0x7f3decbfe640} => {parent_tid=[13204]}, 88) = 13204

     rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 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

     getpid()                                = 13203

     epoll_create1(EPOLL_CLOEXEC)            = 7

     epoll_ctl(7, EPOLL_CTL_ADD, 6, {events=0, data={u32=678363568, u64=94760541815216}}) =
0

     epoll_ctl(7, EPOLL_CTL_MOD, 6, {events=EPOLLIN, data={u32=678363568,
u64=94760541815216}}) = 0

     mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7f3debbfd000

     mprotect(0x7f3debbfe000, 8388608, PROT_READ|PROT_WRITE) = 0

     rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0

     clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|C
LONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, child_tid=0x7f3dec3fd910,
parent_tid=0x7f3dec3fd910, exit_signal=0, stack=0x7f3debbfd000, stack_size=0x7ffc80,
tls=0x7f3dec3fd640} => {parent_tid=[13205]}, 88) = 13205

     rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

     eventfd2(0, EFD_CLOEXEC)                = 8

     fcntl(8, F_GETFL)                       = 0x2 (flags O_RDWR)

     fcntl(8, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     fcntl(8, F_GETFL)                       = 0x802 (flags O_RDWR|O_NONBLOCK)

     fcntl(8, F_SETFL, O_RDWR|O_NONBLOCK)    = 0

     getpid()                                = 13203

     newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...},
AT_EMPTY_PATH) = 0

     write(1, "Starting Control Node\n", 22Starting Control Node

     ) = 22

     write(1, "> ", 2> )                       = 2

     newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...},
AT_EMPTY_PATH) = 0

     read(0, create 1

     "create 1\n", 1024)             = 9
```

```
      clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
child_tidptr=0x7f3decc01c90) = 13218

      getpid()                                = 13203

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

      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(5001), sin_addr=inet_addr("0.0.0.0")}, 16)
= 0

      listen(9, 100)                          = 0

      getsockname(9, {sa_family=AF_INET, sin_port=htons(5001),
sin_addr=inet_addr("0.0.0.0")}, [128 => 16]) = 0

      getsockname(9, {sa_family=AF_INET, sin_port=htons(5001),
sin_addr=inet_addr("0.0.0.0")}, [128 => 16]) = 0

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      write(1, "Ok: 13218\n", 10Ok: 13218

      )              = 10

      write(1, "> ", 2> )                       = 2

      read(0, pingall

      "pingall\n", 1024)              = 8

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203

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

      getpid()                                = 13203
```

```
write(6, "\1\0\0\0\0\0\0\0", 8)         = 8
write(1, "Ok: -1\n", 7Ok: -1
)                 = 7
write(1, "> ", 2> )                       = 2
read(0, q
"q\n", 1024)                    = 2
getpid()                                = 13203
poll([{fd=8, events=POLLIN}], 1, 0)     = 0 (Timeout)
getpid()                                = 13203
write(6, "\1\0\0\0\0\0\0\0", 8)         = 8
getpid()                                = 13203
write(4, "\1\0\0\0\0\0\0\0", 8)         = 8
getpid()                                = 13203
getpid()                                = 13203
write(8, "\1\0\0\0\0\0\0\0", 8)         = 8
futex(0x562f286ef388, FUTEX_WAKE_PRIVATE, 1) = 1
getpid()                                = 13203
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=13218, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---
poll([{fd=3, events=POLLIN}], 1, -1)    = 1 ([{fd=3, revents=POLLIN}])
getpid()                                = 13203
read(3, "\1\0\0\0\0\0\0\0", 8)          = 8
getpid()                                = 13203
write(6, "\1\0\0\0\0\0\0\0", 8)         = 8
close(7)                                = 0
close(6)                                = 0
close(5)                                = 0
close(4)                                = 0
close(3)                                = 0
lseek(0, -1, SEEK_CUR)                  = -1 ESPIPE (Illegal seek)
exit_group(0)                           = ?
+++ exited with 0 +++
```

**Тестирование:**

```
cat_mood@nuclear-box:~/programming/mai-os-labs/lab5-7/build$ ./lab5-7_exe
Starting Control Node
> create 4
```

```
Ok: 7751

> create 2

Ok: 7784

> create 1

Ok: 7787

> create 3

Ok: 7802

>

pingall

Ok: -1

> create 3

Already exists

> pingall

Ok: -1

> exec 1 abcabcabc abc

0 3 6

> exec 4 abcabcabc abc

0 3 6

terminal_2> kill -9 7784

> pingall

Ok: 1;3;2;

> q
```

## Вывод

В ходе лабораторной работы я получил опыт разработки распределённой системы. Научился работать с сокетами и очередями сообщений.