BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Individual Project. Network diagnostic utility in C/C++

Created by Andriy Dutka

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Про мене

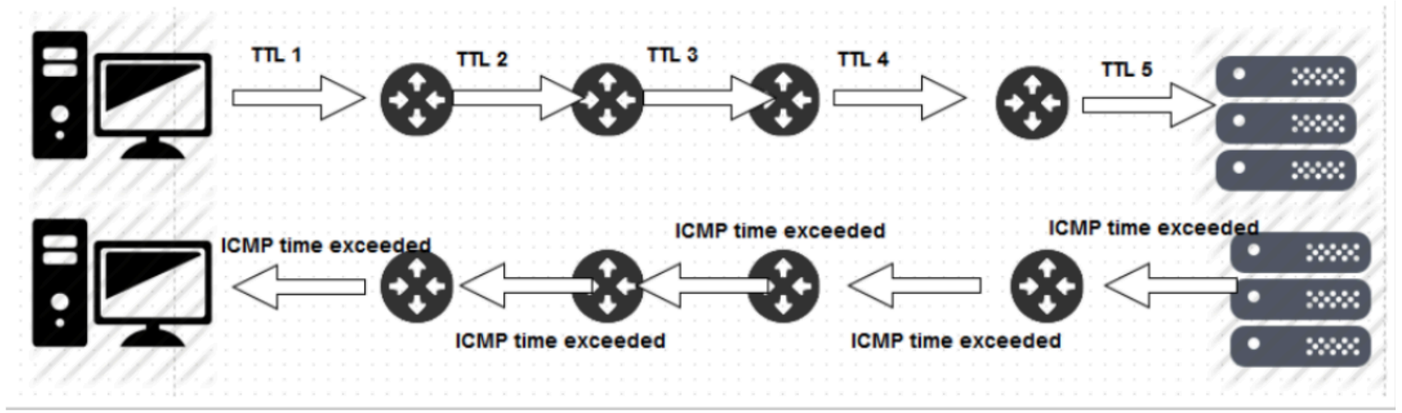
Андрій, 22 роки
Студент Львівської Політехніки
4 курс Кібербезпеки

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Netrouter (traceroute)

це утиліта командного рядка, яка використовується для діагностики мережі



BaseCamp
Linux/Networking

Netrouter (traceroute)

GlobalLogic EDUCATION

```
├── image
│   ├── result.txt
│   ├── traceroute.txt
│   └── usage.txt
├── include
│   ├── ascii_art.h
│   ├── logger_net.h
│   ├── print_logo.h
│   ├── print_table.h
│   ├── raw_socket.h
│   └── trace_route.h
├── Makefile
└── source
    ├── logger_net.cpp
    ├── print_logo.cpp
    ├── print_table.cpp
    ├── raw_socket.cpp
    ├── run_project.cpp
    └── trace_route.cpp
```

```cpp

int main(int argc, char* argv[]){
    PrintLogo printer;
    printer.addLogoByString(art_traceroute);
    // printer.addLogoByString(art_result); # | R E S U L T
    // printer.addLogoByString(art_usage); # | U S A G E
    printer.printLogoByIndexFromString(0); // traceroute.txt

    Answer result;
    switch (argc){
        case 2: {
            TraceRoute traceroute(argv[1]);
            result = traceroute.start();
            printResult(printer, result);
            break;
        }
        case 3: {
            TraceRoute traceroute(argv[1], argv[2]);
            result = traceroute.start();
            printResult(printer, result);
            break;
        }
        default: {
            cerr << "| Usage:" << endl;
            cerr << "|  $ " << argv[0] << " [ FQDN ]" << endl;
            cerr << "|  $ " << argv[0] << " [ FQDN ] [ YOUR_NICKNAME ]" << endl;
            break;
        }
    }

    return 0;
}

```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Class PrintLogo

```cpp
#ifndef PRINTLOGO_H
#define PRINTLOGO_H

#include <iostream>
#include <fstream>
#include <string>
#include <vector>

class PrintLogo{
private:
    std::vector<std::string> paths;
    std::vector<std::string> logos;
    void printFromFile(const std::string& path) const;

public:
    PrintLogo(){}

    void addLogoByPath(const std::string& path);
    void printLogoByIndexFromPath(size_t index) const;

    void addLogoByString(const std::string& logo);
    void printLogoByIndexFromString(size_t index) const;
};

#endif // PRINTLOGO_H
```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Class TraceRoute

```
#ifndef TRACEROUTE_H
#define TRACEROUTE_H

#include <string>
#include <netdb.h>
#include <stdexcept> // for invalid_argument
#include <arpa/inet.h>
#include <chrono>

#define MAX_HOPS 30
using Clock = std::chrono::high_resolution_clock;

struct Answer{
    bool isError;
    std::string time;
};


class TraceRoute{
private:
    std::string fqdn;
    std::string nickname;
    std::string dest_ip;

    void initArguments(const std::string &input_data);
    bool isValidFqdn(const std::string& fqdn);
    bool getIpFromFqdn(const std::string& fqdn);

public:
    TraceRoute(const std::string &input_fqdn);
    TraceRoute(const std::string &input_fqdn,
        const std::string &input_nickname);
    ~TraceRoute();

    Answer start();
};

#endif // TRACEROUTE_H
```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Class Raw Socket

```cpp
#ifndef RAW_SOCKET_H
#define RAW_SOCKET_H

#include <string>
#include <netinet/ip.h>
#include <netinet/ip_icmp.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <iostream>
#include <cstring>
#include <sys/socket.h>

class RawSocket{
private:
    int sockfd;
    bool setTTL(int ttl);
    unsigned short checkSum(const void* data, int length);

public:
    RawSocket();
    ~RawSocket();

    bool sendPacket(const std::string& destinationIP, int ttl);
    bool receivePacket(char* buffer, int bufferSize, std::string& senderIP, int& icmpType, int timeoutMs = 1000);

};

#endif // RAW_SOCKET_H
```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Class Print Table

```cpp
#ifndef PRINTTABLE_H
#define PRINTTABLE_H

#include <iostream>
#include <iomanip>
#include <vector>
#include <string>

class PrintTable{
private:
    std::vector<std::string> headers;
    std::vector<std::vector<std::string>> rows;
    std::vector<size_t> column_widths;

    void updateColumnWidths(const std::vector<std::string>& row);
    void printSeparator() const;
    void printRow(const std::vector<std::string>& row) const;

public:
    PrintTable(const std::vector<std::string>& headers);
    ~PrintTable();

    void addRow(const std::vector<std::string>& row);
    void showTable() const;
};

#endif // PRINTTABLE_H
```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Class Logger

```cpp
#ifndef LOGGER_H
#define LOGGER_H

#include <iostream>
#include <fstream>
#include <string>
#include <ctime>


class Logger{
    public:
    enum LogLevel{
        INFO,
        WARNING,
        ERROR
    };

    Logger(const std::string& filename);
    ~Logger();

    void log(const std::string& message, LogLevel level);

    private:
    std::ofstream log_file;
    std::string getCurrentTime();
};

#endif // LOGGER_H
```

```
2025-03-27 00:27:12 [ INFO ]    Початок роботи програми
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Пакет успішно відправлено
2025-03-27 00:27:12 [ INFO ]    Пакет успішно отримано
2025-03-27 00:27:12 [ INFO ]    Завершення роботи програми
```

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Makefile

```
# ========= Local Makefile ========= #

SRC_DIR = source
SRC_FILES = \
    $(SRC_DIR)/trace_route.cpp \
    $(SRC_DIR)/raw_socket.cpp \
    $(SRC_DIR)/print_logo.cpp \
    $(SRC_DIR)/print_table.cpp \
    $(SRC_DIR)/logger_net.cpp

INC_DIR = include
INC_FILES = \
    $(INC_DIR)/ascii_art.h \
    $(INC_DIR)/trace_route.h \
    $(INC_DIR)/raw_socket.h \
    $(INC_DIR)/print_logo.h \
    $(INC_DIR)/print_table.h \
    $(INC_DIR)/logger_net.h

OBJ_DIR = object
OBJ_FILES = \
    $(OBJ_DIR)/run_project.o \
    $(OBJ_DIR)/trace_route.o \
    $(OBJ_DIR)/raw_socket.o \
    $(OBJ_DIR)/print_logo.o \
    $(OBJ_DIR)/print_table.o \
    $(OBJ_DIR)/logger_net.o
```

```
CXX = g++
CXXFLAGS = -c -I$(INC_DIR)
LOG_DIR = log

start: $(OBJ_FILES)
    $(CXX) $^ -o netrouter

$(OBJ_DIR)/%.o: $(SRC_DIR)/%.cpp $(INC_DIR)/%.h
    $(CXX) $(CXXFLAGS) $< -o $@

# окреме правило для головного файлу run_project.cpp
$(OBJ_DIR)/run_project.o: $(SRC_DIR)/run_project.cpp
    mkdir -p $(OBJ_DIR)
    mkdir -p $(LOG_DIR)
    $(CXX) $(CXXFLAGS) $< -o $@

clean:
    rm -r netrouter
    rm -rf $(OBJ_DIR)
    rm -rf $(LOG_DIR)

restart: clean start

help:
    @echo "Welcome to the traceroute project"
    @echo "make start   - build the project"
    @echo "make clean   - remove all files"
    @echo "make restart - remove all files and build the project again"
    @echo "make help    - show this message"
```

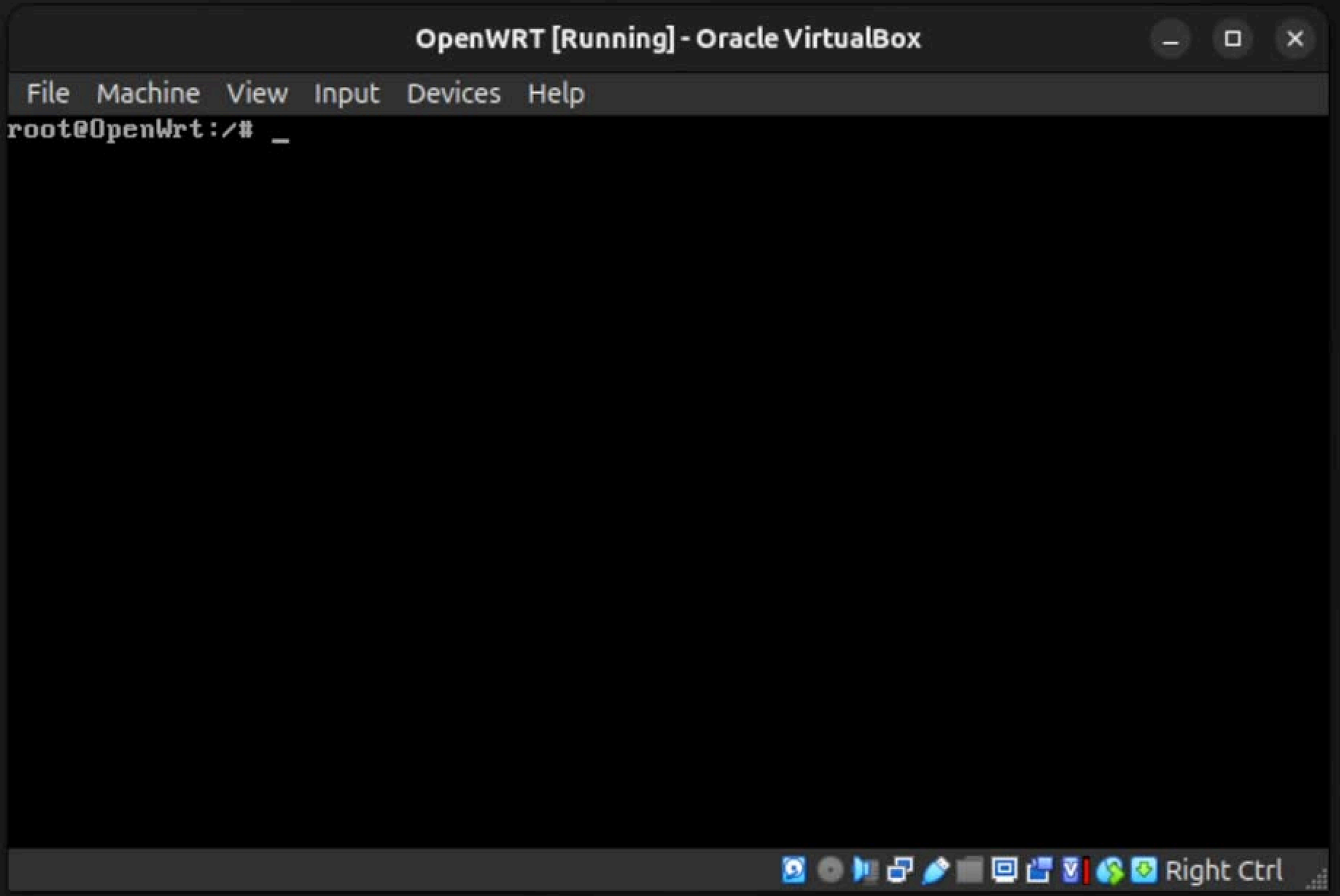
BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Running the Netrouter program

Record

Command:
netrouter [FQDN]
netrouter [IP] [NICKNAME]
netrouter [FAKE-FQDN]
netrouter [127.0.0.1]
netrouter


OpenWRT [Running] - Oracle VirtualBox
File Machine View Input Devices Help
root@OpenWrt:/# _
Right Ctrl

BaseCamp
Linux/Networking

GlobalLogic EDUCATION

Q&A

Created by Andriy Dutka