Vysoké učení technické v Brně
Fakulta informačních technologií


Síťové aplikace a správa sítí

# Manual pro project:
# "Reverse-engineering neznámeho protokolu"

Autor: Kozhevnikov Dmitrii (xkozhe00)

Brno
15.11.2021

## Obsah

## Cíl projektu

Cilem tohoto projektu bylo vytvořit komunikující aplikaci podle konkrétní vybrané specifikace pomocí síťové knihovny BSD. Projekt byl vytvořen v jazyce C++. Projekt se sklada z nekolika častí:

- Zachycení protokolové komunikáce
- Implementování Wireshark dissektoru
- Implementování kompatibilního klienta

## Zachycení protokolové komunikáce

K zadaní projektu byl dan virtuální stroj s referenčním klientem a serverem. Bylo nutné vytvořit spojení mezi klientem a serverem a poté pomocí Wiresharku zachytit tuto komunikaci a analyzovat ji. Výsledek zachycené komunikace je v souboru ***isa.pcap***.

Byl nám chycen neznámý protokol TCP, který obsahoval pakety, které měly nešifrované data jak při odesílání na server, tak při přijímání odpovědi z neho.

Nešifrované údaje jsou odesílány zpravidla ve formátu: ***(příkaz ["data"...])***

A odpoveď ve formatu: ***(odpoveď “informace”)***

Druh zprávy závisí na příkazu, který pošle klient.

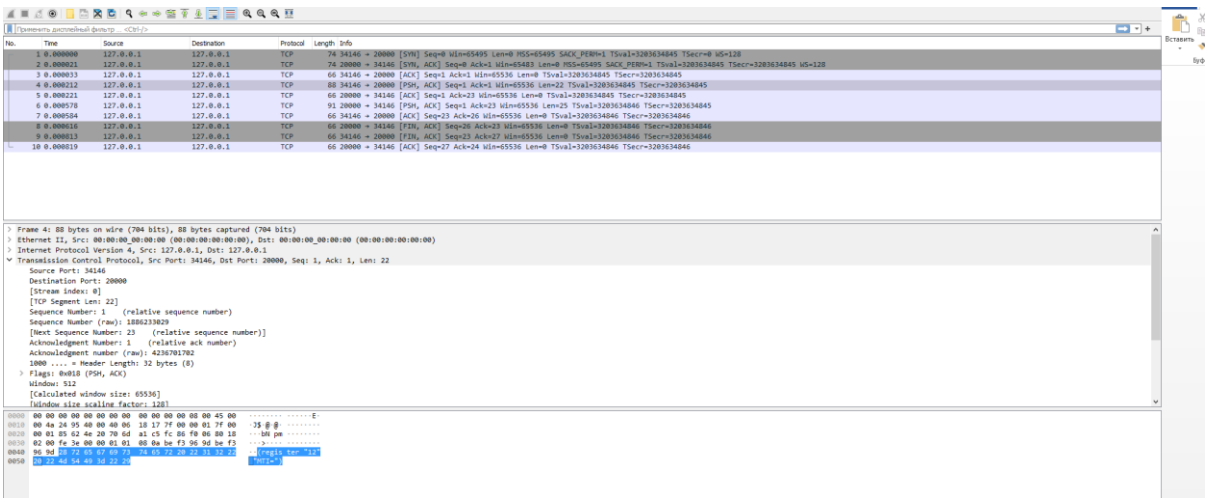
Příklad komandy *register:*

```
·(regis ter "12"
 "MTI=")
```

Client message

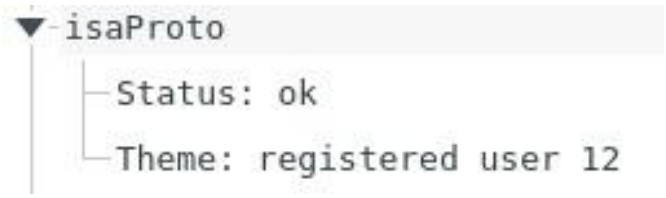
```
·(ok "r egistere
d user 1 2")
```

Server response

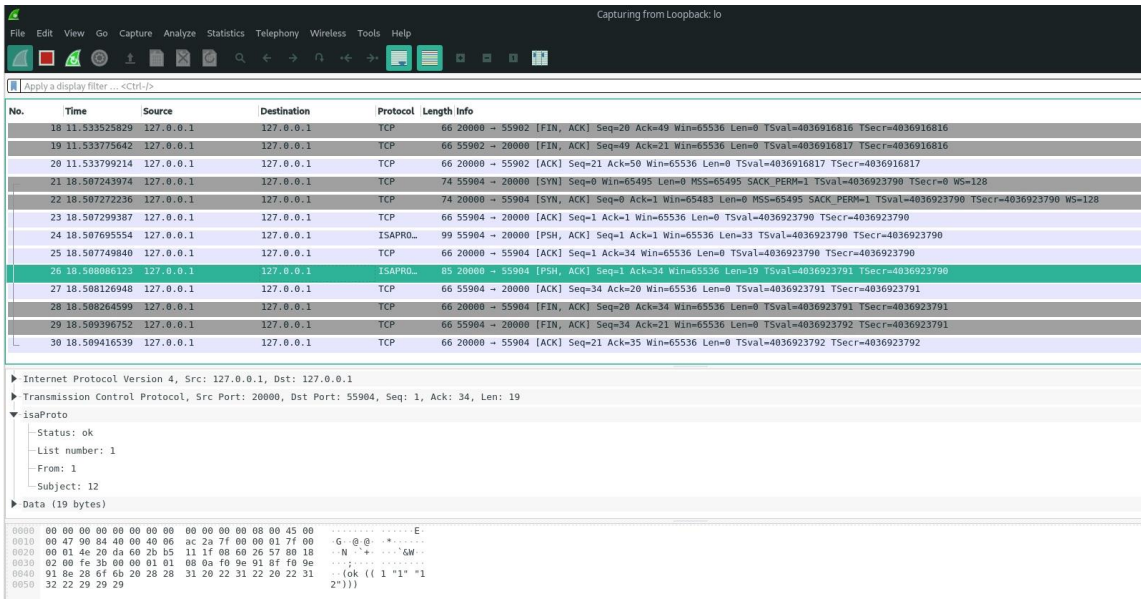


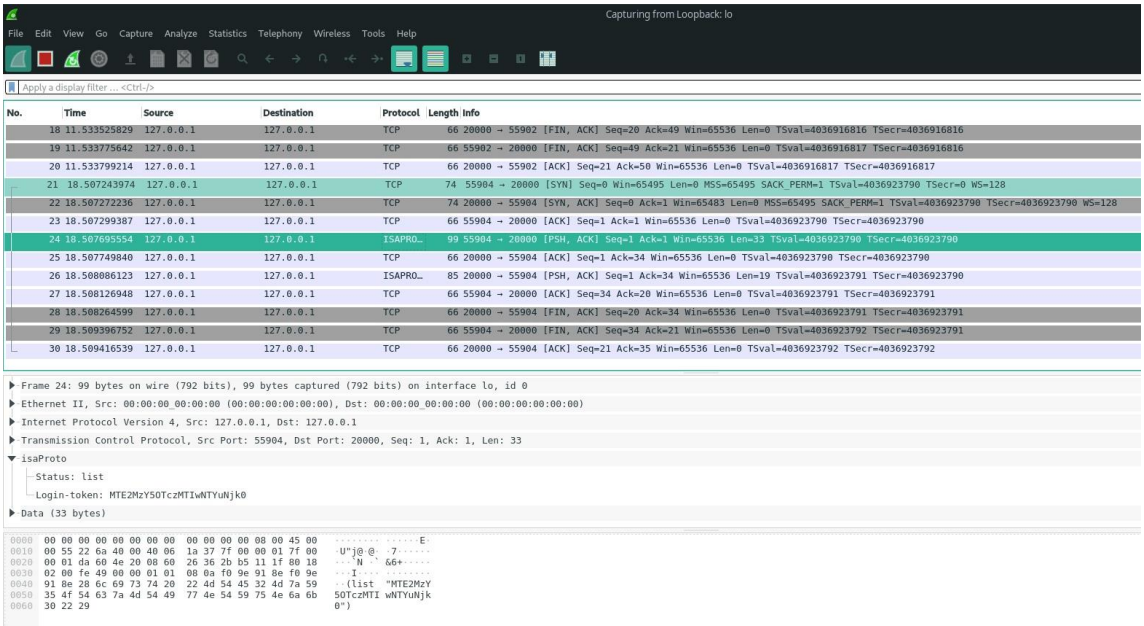
## Implementování Wireshark dissektoru

Pro podporu Wiresharku byl vztvořen vlastní disektor, který implementovan v jazyce Lua. Disektory umožňují prezentovat data protokolu v uživatelsky přívětivější podobě. Já jsem vytvořil disektor nazvaný *isaProto* pro neznámý protokol TCP, který poskytuje data o něm v uživatelsky přívětivém formátu.


Pro tento protokol tento disektor dává podrobné informace jak o požadavku klienta, tak o odpovědi ze serveru.



## Implementování kompatibilního klienta

Největší část projektu, která měla vytvořit klienta pro komunikaci se stávajícím serverem. Tento klient měl být co nejvíce podobný původnímu klientu. Nový klient byl vytvořen v jazyce C++. Tento klient má stejné funkce jako původní.

Také pro tohoto klienta je vytvořen **Makefile**, který provádí potřebné k sestavení, a který je třeba provést pomocí příkazu "*make*" před prvním spuštěním projektu.

Spuštění klienta lze provést z příkazového řádku pomocí příkazu:

***./myClient [ <option> ... ] <command> [<args>]...***

<*option*> je:

- *-a <addr>, --address <addr>* - název serveru nebo adresa, ke které se chcete připoji
- *-p <port>, --port <port>* - port serveru pro připojení k zadanemu portu
- *-help, -h* - zobrazit tuto nápovědu

<*command*> je:

- *register <username> <password>* - registruje uživatele s daným jmenem a heslem
- *login <username> <password>* - umožňuje přihlášení s daným přihlašovacím jménem a heslem pro uživatele je vytvořeno šifrované heslo v kódování ***base64***, které je uloženo do souboru ***token-login***.
- *list* - zobrazí seznam odeslaných zpráv
- *send <recipient> <subject> <body>* - odeslat zprávu pro příjemce, která má předmět zpravy a tělo
- *fetch <id>* - vrácí zprávu podle čísla
- *logout* - odhlásit se

```
[isa@isa isa]$ ./myClient -p 20000 -a 127.0.0.1 login 1 2
ERROR: incorrect password
[isa@isa isa]$ ./myClient -p 20000 -a 127.0.0.1 login 1 1
SUCCESS: user logged in
[isa@isa isa]$ ./myClient -p 20000 -a 127.0.0.1 send 1 12 123
SUCCESS: message sent
[isa@isa isa]$ ./myClient -p 20000 -a 127.0.0.1 list
SUCCESS:
1:
  From: 1
  Subject: 12
```

## Závěr

Tento projekt mi umožnil rozšířit moje znalosti o pochopení fungování sítí. Lépe jsem pochopit, jak funguje Wireshark. Také byl prostudován materiál na zpracování paketu dat a práce dissektorů a možnost jejich implementace pro určité protokoly.

Také jsem byl schopen zvládnout implementace klienta, který rozšířil můj pohled na interakci *"client-server"*, zlepšil jsem znalostí práce s knihovnou BSD socket a naučil jsem se zpracovávat komunikaci.

## Seznam literatury


1. https://gitlab.com/wireshark/wireshark/-/wikis/Lua
2. https://gitlab.com/wireshark/wireshark/-/wikis/Lua/Dissectors
3. https://gitlab.com/wireshark/wireshark/-/wikis/Lua/Examples
4. https://en.wikipedia.org/
5. https://en.wikiversity.org/wiki/Internet_Protocol_Analysis/Internet_Layer_IPv4
6. https://support.sas.com/documentation/onlinedoc/sasc/doc/lr2/lrv2ch15.htm
7. https://www.wireshark.org/
8. https://habr.com/
9. https://stackoverflow.com/questions/342409/how-do-i-base64-encode-decode-in-c
10. https://mika-s.github.io/wireshark/lua/dissector/2017/11/04/creating-a-wireshark-dissector-in-lua-1.html
11. https://linux.die.net/man/3/getopt_long
12. Přednášky z předmetů IPK a ISA