Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Jakub Šimko

SIP PROXY - Dokumentácia

Mobilné technológie a aplikácie – Zadanie 1

Cvičiaci: Ing. Matej Janeba
Čas cvičení: Streda, 16:00 - 17:50

Meno: Jakub Šimko
AIS ID: 103146

1. Využitá knižnica

Knižnica ktorú som využil:
https://github.com/tirfil/PySipFullProxy?fbclid=IwAR0IDiXSy9Ph4iJHALXnZABQrjGLIeagGZM2C9MgdzbeaHK3DNDIHHidQ-k

2. Spôsob implementácie a sprevádzkovania SIP servera

Pri implementácii som využil vyššie spomenutú knižnicu, ktorú odporučil Ing. Marek Galinski, PhD. Knižnica ale mala niekoľko chýb, ktoré bolo nutné opraviť. Najprv som musel opraviť všetky chyby spojené s prevodom kódu z Python 2 na Python 3. Následne veľký problém nad ktorým som strávil veľa času bol ten, že mi nešlo zložiť hovor na iných zariadeniach než na tom na ktorom bežal SIP server. BYE signál sa nenachádzal ani vo wiresharku. Debuggovaním som sa nakoniec dopracoval k oprave a teda chyba sa nachádzala v riadku kde bola volaná funkcia gethostbyname(), ktorá z nejakého dôvodu vracia nesprávnu IP adresu, preto je v main funkcii teraz využívaná len premenná HOST, ktorá obsahuje IP adresu servera.

```
ipaddress = socket.gethostbyname(hostname)
```

```
HOST, PORT = "147.175.191.134", 5060
```

Knižnica obsahovala všetky základné a väčšinu extra bodov zo zadania. Chýbala len implementácia denníka hovorov a úprava stavových kódov, ktoré som doprogramoval spolu s vlastnou main funkciou kde som funkcie knižnice owrapoval. V programe fungujú všetky základné aj extra funkcionality.

Pri testovaní jednotlivých funkcionalít som využil klient **Linphone**.

Meno: Jakub Šimko
AIS ID: 103146

2.1. Denník hovorov

Vrámci doimplementovania denníka hovorov som upravil funkcie *processRequest* a *processCode*. Spôsob implementácie je vysvetlený v komentároch v kóde.

```
def processRequest(self):
    # print "processRequest"
    if len(self.data) > 0:
        request_uri = self.data[0]
        if rx_register.search(request_uri):
            self.processRegister()
        elif rx_invite.search(request_uri):
            self.processInvite()
            logging.info("%s vola %s" % (self.getOrigin(), self.getDestination()))
        elif rx_ack.search(request_uri):
            self.processAck()
        elif rx_bye.search(request_uri):
            self.processNonInvite()
            logging.info("Hovor medzi %s a %s bol ukonceny" % (self.getOrigin(), self.getDestination()))
```

```
def processCode(self):
```

.

.

```
if "180" in data[0]:
    # nedokazem len na zaklade 200 urcit ci bol hovor prijaty
    # pretoze kod 200 je spracovavany aj vramci register
    # preto ak pred kodom 200 bolo 180 ringing tak viem ze hovor bol prijaty
    # pretoze sa jedna o odpoved na INVITE
    last_ringing = True
else:
    if "486" in data[0]:
        logging.info("Hovor medzi %s a %s nebol prijaty - busy" % (self.getOrigin(), self.getDestination()))
        # logging.info("Hovor medzi %s a %s bol ukonceny" % (self.getOrigin(), self.getDestination()))
    elif "603" in data[0]:
        logging.info("Hovor medzi %s a %s bol odmietnuty" % (self.getOrigin(), self.getDestination()))
        # logging.info("Hovor medzi %s a %s bol ukonceny" % (self.getOrigin(), self.getDestination()))
    elif last_ringing and ("200" in data[0]):
        logging.info("Hovor medzi %s a %s bol prijaty" % (self.getOrigin(), self.getDestination()))

    last_ringing = False
```

Denník hovorov:

```
21:40:09: mobil@147.175.191.134 vola pocitac@147.175.191.134
21:40:11: Hovor medzi mobil@147.175.191.134 a pocitac@147.175.191.134 bol prijaty
21:40:14: Hovor medzi pocitac@147.175.191.134 a mobil@147.175.191.134 bol ukonceny
21:40:17: mobil@147.175.191.134 vola pocitac@147.175.191.134
21:40:19: Hovor medzi mobil@147.175.191.134 a pocitac@147.175.191.134 bol odmietnuty
```

Meno: Jakub Šimko
AIS ID: 103146

2.2. Úprava SIP stavových kódov

Server je schopný posielať SIP stavové kódy, ktoré sú zadané v slovníku response_codes_dict.

```
response_codes_dict = {
    "180": "180 Zvonim",
    "200": "200 OK VYBAVENE",

    "400": "400 Bad Request",
    "406": "406 Not Acceptable",
    "480": "480 Temporarily Unavailable",
    "486": "486 Obsadené",
    "488": "488 Not Acceptable Here",

    "500": "500 Server Internal Error",
}
```

Funkcionalitu som doprogramoval do knižnice vo funkcii *processCode*:

```
351     def processCode(self):
352         global last_ringing
353 
354         origin = self.getOrigin()
355         if len(origin) > 0:
356             logging.debug("origin %s" % origin)
357             if origin in registrar:
358                 socket, claddr = self.getSocketInfo(origin)
359                 self.data = self.removeRouteHeader()
360                 data = self.removeTopVia()
361                 text = "\r\n".join(data)
362 
363                 # uprava akehokolvek sip stavoveho kodu ktory posle sip server
364                 for key, value in response_codes_dict.items():
365                     if key in data[0]:
366                         data[0] = f"SIP/2.0 {value}"
367 
```

Vo viacerých častiach knižnice kde sa odosielajú stavové kódy bola taktiež nutná úprava:

```
self.sendResponse(response_codes_dict["200"])
```

Meno: Jakub Šimko
AIS ID: 103146

3. Wireshark

Všetky pcap súbory so scénarmi zo zadania sa nachádzajú v priečinku PCAP SUBORY. Pri všetkych komunikáciach boli využívané upravené stavové kódy, ktoré sa dajú zmeniť v programe.

4. Link na Github repozitár s PCAP súbormi a kódom

https://github.com/JSimko4/xsimkoj2