Universidade Federal de Minas Gerais
Instituto de Ciéncias Exatas
Graduação em Ciéncia da Computação
Disciplina de Redes de Computadores

Trabalho Prático 0

Sistema de autenticação de alunos

cliente-servidor com socket

Aluna: Giulia Monteiro Silva Gomes Vieira
Matrícula: 2016006492

Data: 5 de abril de 2024

Conteúdo

1 Visão geral do projeto

Para o trabalho prático 0 da disciplina de redes de computadores, implementei um sistema de cliente-servidor utilizando a biblioteca socket da linguagem python. Como definido nas especificações do projeto, as mensagens implementadas são em binário e se organizam nos seguintes formatos:

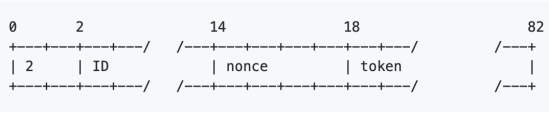
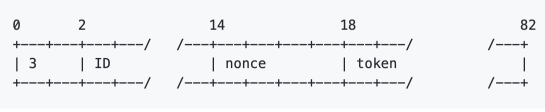
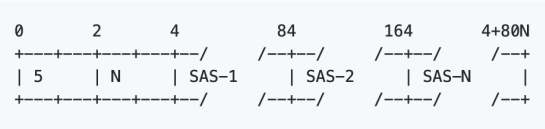
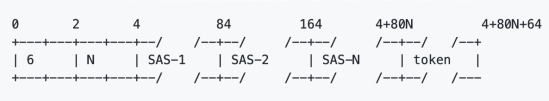
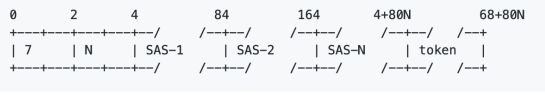
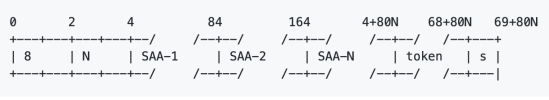
Client	**Server**
Individual Token Request 	Individual Token Response 
Individual Token Verification Request 	Individual Token Verification Response 
Group Token Request 	Group Token Request Response 
Group Token Verification Request 	Group Token Verification Response 

Tabela 1: Client-Server Interaction

O programa também avalia cinco tipos de erro, INVALID_MESSAGE_CODE, INCORRECT_MESAGE_LENGTH, INVALID_PARAMETER, INVALID_SINGLE_TOKEN e ASCII_DECODE_ERROR.

Na implementação também adicionei um timeout para a conexão do lado do cliente, mas enquanto o timeout não acaba, o cliente reenvia a mensagem ao servidor intermitentemente até receber uma resposta.

Após ler novamente as especificações, percebi que a implementação do servidor era desnecessária, visto que deveríamos testar o cliente com o servidor disponibilizado pelo professor. Contudo, escrever o servidor me ajudou a compreender o problema, portanto mantive os arquivos na submissão deste projeto.

2 Testes em servidor do professor

Neste projeto, para além de testar localmente, deveríamos testar as funções no servidor disponibilizado pelo professor, no endereço *slardar.snes.2advanced.dev*. Meus testes se comportaram da seguinte maneira

```
# Get token for nonce 01

python client.py slardar.snes.2advanced.dev. 51001 itr 2016006492 1

2016006492:1:
f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3
```

```
# Get token id for nonce 02

python client.py slardar.snes.2advanced.dev. 51001 itr 2016006492 2

2016006492:2:
8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6
```

```
# Verify token for nonce 01

python3 client.py slardar.snes.2advanced.dev. 51001 itv 2016006492:1:
f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3

0
```

```
# Verify token for nonce 02

python3 client.py slardar.snes.2advanced.dev. 51001 itv 2016006492:2:
8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6

0
```

```
# Get group token for nonce 01 and nonce 02

python3 client.py slardar.snes.2advanced.dev. 51001 gtr 2 2016006492:1:f
8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3
    2016006492:2:
8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6

2016006492:1:
f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3+
2016006492:2:
8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6+
c29d489ca989ecf98917d3739995b741443af8ad239898c8081594ca8283649b
```

```
# Verify group token for nonce 01 and nonce 02

python3 client.py slardar.snes.2advanced.dev. 51001 gtv 2 2016006492:1:f
8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3+
2016006492:2:
8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6+
c29d489ca989ecf98917d3739995b741443af8ad239898c8081594ca8283649b

0
```

Terminei este projeto adiantado, portanto quanto o entreguei a primeira vez o servidor *slardar.snes.2advanced.dev* ainda estava de pé e eu executei as funções nele. Contudo, hoje descobri que esse servidor não existe mais, e agora o endereço é *pugna.snes.dcc.ufmg.br.* Portanto, para fins de completude, executei as mesmas funções em ipv4 neste servidor (visto que o professor informou que o ipv6 dele não está funcionando). Os resultados foram exatamente iguais aos da execussão no servidor anterior, portanto anexarei aqui apenas o print do meu terminal com as execussões.

```
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 itr
 2016006492 1
2016006492:1:f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 itr
 2016006492 2
2016006492:2:8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 itv
 2016006492:2:8332afd72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6
0
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 itv
 2016006492:1:f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3
0
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 gtr
 2 2016006492:1:f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3 2016006492:2:8332afd
72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6

2016006492:1:f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3+2016006492:2:8332afd728
42f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6+c29d489ca989ecf98917d3739995b741443af8ad239898
c8081594ca8283649b
giuliavieira@Giulias-MacBook-Air-2 authenticator % python3 client.py pugna.snes.dcc.ufmg.br 51001 gtv
 2 2016006492:1:f8f86bb2eccb020597c5f2b5200209daf29b6d03aef4dfe06f33495e76212aa3+2016006492:2:8332afd
72842f143a44d59e115759416010452c94e7af1ce7f98c95844047bb6+c29d489ca989ecf98917d3739995b741443af8ad239
898c8081594ca8283649b
0
```

No arquivo anexo chamado **cli-tests.txt** especifico todas as chamadas do cliente contra os dois servidores do professor e meu próprio, assim como as saídas recebidas em cada caso.

3 Vulnerabilidades

Algumas vulnerabilidades potenciais do sitema são:

- **Ataques de Negação de Serviço (DoS):** Como o UDP é sem conexão e não requer um handshake como o TCP, ele é vulnerável a ataques de inundação UDP, onde um atacante inunda o servidor com um grande número de pacotes UDP, fazendo com que ele fique sobrecarregado e incapaz de responder a solicitações legítimas. Isso inclusive aconteceu enquanto eu estava testando no servidor e não tinha implementado um limite para os reenvios do cliente.

- **Falsificação de Pacotes:** Como o UDP não fornece mecanismos embutidos para autenticação de pacotes, um atacante pode falsificar pacotes para se passar por um aluno, potencialmente copiando seu token. Nós inclusive podemos saber os tokens dos demais alunos, dado que o número de matrícula é público.

- **Ataques de Homem no Meio (MITM):** Sem mecanismos de criptografia ou autenticação, um atacante poderia interceptar e modificar mensagens entre o cliente e o servidor,por exemplo descobrindo o nonce escolhido pelo aluno.

- **Manipulação de Mensagens:** Sem verificações de integridade ou autenticação de mensagens, um atacante poderia modificar o conteúdo dos pacotes UDP em trânsito, por exemplo trocando o número de matrícula ou nonce do aluno.