



FACULDADE DE CIÊNCIAS E TECNOLOGIA DA UNIVERSIDADE DE COIMBRA



**DEEC**

DEPARTAMENTO DE ENGENHARIA  
ELECTROTÉCNICA E DE COMPUTADORES

## ***REDES DE COMPUTADORES***

TPC

### **SetEvents**

**Trabalho realizado por:**

Filipe João Abrantes Soares da Conceição, 2014196660

João Carlos da Costa Barreiros, 2014196880

## ***Objetivo***

O objetivo deste trabalho é desenvolver um programa (em C / C++) de computador cliente-servidor (SetEvent) que informa os clientes de uma lista de eventos e que permite que estes escolham e se registem num desses eventos. A lista de eventos é guardada num ficheiro de texto (*events.txt*) e os registos dos vários clientes são guardados noutros ficheiros de texto, um ficheiro por evento. Cada utilizador apenas se pode registrar uma vez em cada evento. Os registos repetidos no mesmo evento não são aceites, não sendo reconhecidos.

## ***Programa desenvolvido***

Neste trabalho foi desenvolvido um programa cliente-servidor que está contido em dois ficheiros, o client e o server.

Para iniciar o programa servidor deve ser colocado o número de porto e uma variável booleana (0 ou 1) que vai permitir controlar os registos dos clientes. No servidor é criada uma socket TCP com o objetivo de se poder comunicar com um determinado número de utilizadores que corram o programa cliente. Este tem como finalidade, fornecer os serviços pretendidos pelo cliente. Caso o cliente pretenda conhecer os eventos que estão disponíveis (opção 3- “Get List of events”), é o servidor que lê do ficheiro de eventos (*events.txt*) a informação sobre cada evento relativa ao número do evento, à localização deste, a sua data de acontecimento, a duração expectável do mesmo e o número de vagas ainda disponíveis de inscrição. Seguidamente, envia ao cliente toda esta informação relativa a todos os eventos, por meio da socket criada. Caso o cliente pretenda registar-se num desses eventos (opção 4 – “Make registration”), o servidor tem como função registar esse cliente (com o respetivo nome de utilizador e palavra chave) no ficheiro *users.txt*, um ficheiro que contém todos os pares username + password para cada utilizador. Para além disso, o servidor também tem a opção de não aceitar registos de novos utilizadores, isto é, apenas permitir o início de sessão (login) de utilizadores que já estejam registados, ou seja, que se encontrem no ficheiro referido. Para tal é necessário verificar se o utilizador e a palavra-chave introduzida correspondem. O servidor também permite que a palavra-chave seja encriptada, tal encriptação é feita com base no nome de utilizador (username) respetivo.

O programa cliente permite que um utilizador contacte com o servidor de forma a serem pedidos os serviços anteriormente descritos. No entanto, é necessário configurar primeiro o programa cliente com o endereço IP do servidor (opção 1 – “Set event server”) e o número do porto respetivo à socket (opção 2 – “Set port”), que tem de pertencer a um intervalo específico. Uma última opção que o utilizador pode escolher é a saída do programa (opção 0 – “Exit”), após uma confirmação adequada.

## ***Guia do utilizador***

Quando um determinado utilizador corre o programa cliente é-lhe, de imediato, facultado um menu com 6 opções, informação sobre o endereço IP e o número de porto do servidor atual. Para selecionar uma opção o utilizador necessita de introduzir o número respetivo da opção e premir a tecla “Enter”. Se o utilizador introduzir uma opção que não é válida, é apresentada uma mensagem no ecrã a informar que o número/caracter que introduziu não é válido. Cada uma das opções permite ao utilizador o seguinte:

**0** - Encerrar o programa cliente, após uma mensagem de confirmação adequada de y/n (sim ou não). Caso o utilizador não introduza nenhum dos caracteres relativos à pergunta de confirmação, estes não são aceites, sendo a confirmação da saída de novo inquerida.

**1**- É pedido ao utilizador o endereço IP do hospedeiro que corre o programa servidor, isto é, do programa que se pretende que forneça os serviços que o utilizador pretende.

**2**- É pedido ao utilizador o número do porto a ser usado no hospedeiro que corre o programa servidor. O intervalo dos números dos portos disponíveis é indicado no ecrã e, caso o utilizador introduza um número fora desse intervalo, este é alertado do mesmo, sendo pedido um novo número de porto.

**3**- Inicia a conexão com o programa servidor e obtém a lista de eventos que se encontram disponíveis (que o servidor pode fornecer) no momento atual. No entanto, é necessário que o utilizador se registe primeiro, caso seja a primeira vez que está a correr o programa, ou inicie a respetiva sessão (login), caso já se tenha registado no passado. Na eventualidade de o servidor não aceitar registos de novos clientes, apenas é possível iniciar a sessão (introdução do nome de utilizador-username e palavra chave-password respetiva), no caso de se tratar de um cliente previamente registado.

**4**- Se a sessão não estiver iniciada, permite que o utilizador se registe ou inicie a respetiva sessão, de forma a poder usufruir dos serviços prestados pelo servidor, através da introdução do nome de utilizador e palavra-chave respetiva. Se o utilizador já se encontrar logado, é perguntado a este se pretende registar-se em algum dos eventos fornecidos pelo servidor (caso semelhante à opção 3), esperando pela respetiva resposta y/n. Em caso afirmativo, é pedido ao utilizador para introduzir o número do respetivo evento desejado e o número de reservas para tal. Caso o registo seja bem sucedido (isto é, não seja um registo repetido do mesmo, utilizado no mesmo evento ou o número de vagas disponíveis para o evento seja inferior ao número de vagas que o utilizado solicitou) essa informação é enviada para o servidor.

## ***Resultados***

As seguintes captura do Wireshark apresentadas ilustram a troca de mensagens entre o programa cliente o servidor, no ato de autenticação do utilizador do programa cliente.

146	2.820566248	127.0.0.1	127.0.0.1	TCP	68	39890 → 50000 [PSH, ACK] Seq=3 Ack=1 Win=43776 Len=2 TSval=95586 TSecr=95301
147	2.820622630	127.0.0.1	127.0.0.1	TCP	66	50000 → 39890 [ACK] Seq=1 Ack=5 Win=43776 Len=0 TSval=95586 TSecr=95586
148	2.831948700	127.0.0.1	127.0.0.1	TCP	74	43100 → 4101 [SYN] Seq=0 Win=0 MSS=65495 SACK_PERM=1 TSval=95589 TSecr=0

▶ Frame 146: 68 bytes on wire (544 bits). 68 bytes captured (544 bits) on interface 0  
 ▶ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)  
 ▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
 ▼ Transmission Control Protocol, Src Port: 39890, Dst Port: 50000, Seq: 3, Ack: 1, Len: 2

Source Port: 39890  
 Destination Port: 50000  
 [Stream index: 58]  
 [TCP Segment Len: 2]  
 Sequence number: 3 (relative sequence number)  
 [Next sequence number: 5 (relative sequence number)]  
 Acknowledgment number: 1 (relative ack number)  
 Header Length: 32 bytes  
 ▶ Flags: 0x018 (PSH, ACK)  
 Window size value: 342  
 [Calculated window size: 43776]  
 [Window size scaling factor: 128]  
 Checksum: 0xfe2a [unverified]  
 [Checksum Status: Unverified]  
 Urgent pointer: 0  
 ▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps  
 ▶ [SEQ/ACK analysis]

▼ Data (2 bytes)  
 Data: 6a63  
 [Length: 2]

Figura 1: Envio do nome de utilizador para o programa servidor.

0000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 45 00	.....E.
0010	00 36 1c 7e 40 00 40 06 20 42 7f 00 00 01 7f 00	.6..@. B.....
0020	00 01 9b d2 c3 50 2c 2f 68 41 3d 42 b1 f2 80 18	....P,/ hA=B....
0030	01 56 fe 2a 00 00 01 01 08 0a 00 01 75 62 00 01	.V*.....ub..
0040	74 45 6a 63	tE

  

172	3.819123767	127.0.0.1	127.0.0.1	TCP	70	39890 → 50000 [PSH, ACK] Seq=5 Ack=1 Win=43776 Len=4 TSval=95835 TSecr=95586
173	3.819140456	127.0.0.1	127.0.0.1	TCP	66	50000 → 39890 [ACK] Seq=1 Ack=9 Win=43776 Len=0 TSval=95835 TSecr=95835
174	3.819217623	127.0.0.1	127.0.0.1	TCP	321	50000 → 39890 [PSH, ACK] Seq=1 Ack=9 Win=43776 Len=255 TSval=95835 TSecr=95835
175	3.819339745	127.0.0.1	127.0.0.1	TCP	321	50000 → 39890 [FIN, PSH, ACK] Seq=256 Ack=9 Win=43776 Len=255 TSval=95835 TSecr=95835
176	3.819358257	127.0.0.1	127.0.0.1	TCP	66	39890 → 50000 [ACK] Seq=9 Ack=256 Win=44800 Len=0 TSval=95835 TSecr=95835
177	3.831664325	127.0.0.1	127.0.0.1	TCP	74	43132 → 4101 [SYN] Seq=0 Win=0 MSS=65495 SACK_PERM=1 TSval=95838 TSecr=0 WS=128

▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
 ▼ Transmission Control Protocol, Src Port: 39890, Dst Port: 50000, Seq: 5, Ack: 1, Len: 4

Source Port: 39890  
 Destination Port: 50000  
 [Stream index: 58]  
 [TCP Segment Len: 4]  
 Sequence number: 5 (relative sequence number)  
 [Next sequence number: 9 (relative sequence number)]  
 Acknowledgment number: 1 (relative ack number)  
 Header Length: 32 bytes  
 ▶ Flags: 0x018 (PSH, ACK)  
 Window size value: 342  
 [Calculated window size: 43776]  
 [Window size scaling factor: 128]  
 Checksum: 0xfe2c [unverified]  
 [Checksum Status: Unverified]  
 Urgent pointer: 0  
 ▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps  
 ▶ [SEQ/ACK analysis]

▼ Data (4 bytes)  
 Data: 6a636363

0000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 45 00	.....E.
0010	00 36 1c 7f 40 00 40 06 20 3f 7f 00 00 01 7f 00	.8..@. ?.....
0020	00 01 9b d2 c3 50 2c 2f 68 43 3d 42 b1 f2 80 18	....P,/ hC=B....
0030	01 56 fe 2c 00 00 01 01 08 0a 00 01 76 5b 00 01	.V*.....v[...
0040	75 62 31 32 33 34	ub

Figura 2: Envio da palavra chave para o programa servidor.