International Institute of Information Technology Hyderabad

System and Network Security (CS5470)

Lab Assignment 1:
Secure file transfer
using a client-server programming model
with 3DES with three keys cryptosystem
Deadline: February 3, 2020 (Monday), 23:59 PM
Total Marks: 100

Note:- *It is strongly recommended that no student is allowed to copy programs from others. No assignment will be taken after deadline. Name your programs as rollno_assign_1_client.c and rollno_assign_1_server.c for the client and server, respectively. Upload your only rollno_assign_1_client.c and rollno_assign_1_server.c files along with a README file in a zip file (rollno_assign_1.zip) to course portal (moodle).* ***You are allowed to use any programming language implementation (for example, C, C++, Java, Python).***

Problem Description

Suppose there are n_c users A (clients) want to securely access the files stored in the database of a user B (server).

At first, each client will make a connection using *socket* with the server B and then will establish three *symmetric keys*, say K_1, K_2 and K_3 which will be used for encryption and decryption using the **3DES with three keys** symmetric cryptosystem.

Part 1 (Key Establishment using Diffie-Hellman Key Exchange Protocol)

For establishment of a shared session key $K_{A,B}$ between the client A and the server B, you will use the Diffie-Hellman key exchange protocol, which is described below:

- *Global Public Elements*
  - q: a sufficiently large prime, such that it is intractable to compute the discrete logarithms in Z_q^*.

  - α: $\alpha < q$ and α a primitive root of q.

- *User A Key Generation*

  - Select private X_A such that $X_A < q$
  - Calculate public Y_A such that $Y_A = \alpha^{X_A} \mod q$
    $A \rightarrow B : \{Y_A, q, \alpha\}$
    Here $A \rightarrow B : M$ denotes party A sends a message M to party B.

- *User B Key Generation*

  - Select private X_B such that $X_B < q$
  - Calculate public Y_B such that $Y_B = \alpha^{X_B} \mod q$
    $B \rightarrow A : \{Y_B\}$

- *Generation of secret key by User A*

  - Compute the shared key with B as $K_{A,B} = (Y_B)^{X_A} \mod q$

- *Generation of secret key by User B*

  - Compute the shared key with A as $K_{B,A} = (Y_A)^{X_B} \mod q = K_{A,B}$

Part 2 (Secure Files Transfer)

- Assume that each client A has already established three symmetric keys K_1, K_2 and K_3 using *Part 1*.

- Both client A and server B will use the encoding technique for all printable ASCII characters (0 to 127) represented in seven binary bits (see the attached ASCII table).

- A client (A) sends a request message REQSERV for requesting the service (transferring a requested file) from the server (B).

- Suppose the server B is allowed to transmit at a time a maximum of 1024 bytes to a client (A). After receiving the message REQERV, if the requested file is not present at the server B, an appropriate message (DISCONNECT) will be sent by the server B to the client A indicating the 'no such file exists'. Otherwise, the server B will send the encrypted messages ENCMSG on the file to the client A at a time a maximum of 1024 bytes until the entire file is transferred.

- Finally, a completion message REQCOM will be sent from the server B to the client A to indicate that the whole file has been successfully transmitted followed by DISCONNECT messages.

The protocol messages to be used in implementation are provided below.

Table 1: Protocol messages to be used in implementation

Opcode	*Message*	*Description*
10	PUBKEY	Public key Y_A/Y_B sent to the server/client by the client/server
20	REQSERV	Request for service (transferring a requested file) from the server to the client
30	ENCMSG	Sending encrypted message(s) for the file from the server to the client
40	REQCOM	Request completion message from the server to the client
50	DISCONNET	Disconnect request message sent from the client to the server

The communication messages for this problem are given below:

Table 2: Communication message flow to be used for implementation

Client (User A)	Server (User B)
Diffie-Hellman key exchange algorithm	
$\xrightarrow{PUBKEY}$	
	$\xleftarrow{PUBKEY}$
Secure file transfer using 3DES with three keys cryptosystem	
$\xrightarrow{REQSERV}$	
	$\xleftarrow{ENCMSG}$
	$\xleftarrow{ENCMSG}$
	$\vdots$
	$\xleftarrow{ENCMSG}$
	$\xleftarrow{REQCOM}$
$\xrightarrow{DISCONNET}$	
	$\xleftarrow{DISCONNET}$

Data structure to be used for implementation

```
/* Header of a general message */
typedef struct {
    int opcode; /* opcode for a message */
    int s_addr; /* source address */
    int d_addr; /* destination address */
} Hdr;
```

```
/* Public key $Y_A = \alpha^{X_A} \mod q$ or $Y_B = \alpha^{X_B} \mod q$ */
typedef struct {
    long $q$; /* large prime */
    long $\alpha$; /* primitive root */
    long $Y$; /* public key */
} PubKey;
```

Similarly, define other message structures clearly. Finally, the generalized message will contain the Hdr and union of all the other defined messages defined. For example,

```
/*A general message */
typedef struct {
    Hdr hdr; /* Header for a message */
    typedef union {
    PubKey pubkey;
    ReqServ reqserv;
    ReqCom reqcom;
    EncMsg encmsg;
    Disconnect disconnect;
    } AllMsg;
} Msg;
```

Instructions:

1. Write your server.c program in server directory, say Server. Run the program as: ./a.out
2. Write your client1.c, client2.c, client3.c, . . . programs in the clients directories, say Client1, Client2, Client3, ...
   Run the program as: ./a.out 127.0.0.1 (for local host loop back)
3. Display your all messages at both client and server sides.
4. Each student must register in course portal (moodle) for online submissions.

All the best!!!

ASCII table: An overview of all ASCII codes

Bin.	Hex.	Dec.	ASCII Symbol	Explanation	Group
0000000	0	0	NUL	The null character prompts the device to do nothing	Control Character
0000001	1	1	SOH	Initiates a header (Start of Heading)	
0000010	2	2	STX	Ends the header and marks the beginning of a message. (start of text)	
0000011	3	3	ETX	Indicates the end of the message (end of text)	
0000100	4	4	EOT	Marks the end of a completes transmission (End of Transmission)	
0000101	5	5	ENQ	A request that requires a response (Enquiry)	
0000110	6	6	ACK	Gives a positive answer to the request (Acknowledge)	
0000111	7	7	BEL	Triggers a beep (Bell)	
0001000	8	8	BS	Lets the cursor move back one step (Backspace)	
0001001	9	9	TAB (HT)	A horizontal tab that moves the cursor within a row to the next predefined position (Horizontal Tab)	
0001010	A	10	LF	Causes the cursor to jump to the next line (Line Feed)	
0001011	B	11	VT	The vertical tab lets the cursor jump to a predefined line (Vertical Tab)	
0001100	C	12	FF	Requests a page break (Form Feed)	

0001101	D	13	CR	Moves the cursor back to the first position of the line (Carriage Return)	
0001110	E	14	SO	Switches to a special presentation (Shift Out)	
0001111	F	15	SI	Switches the display back to the normal state (Shift In)	
0010000	10	16	DLE	Changes the meaning of the following characters (Data Link Escape)	
0010001	11	17	DC1	Control characters assigned depending on the device used (Device Control)	
0010010	12	18	DC2		
0010011	13	19	DC3		
0010100	14	20	DC4		
0010101	15	21	NAK	Negative response to a request (Negative Acknowledge)	
0010110	16	22	SYN	Synchronizes a data transfer, even if no signals are transmitted (Synchronous Idle)	
0010111	17	23	ETB	Marks the end of a transmission block (End of Transmission Block)	
0011000	18	24	CAN	Makes it clear that a transmission was faulty and the data must be discarded (Cancel)	
0011001	19	25	EM	Indicates the end of the storage medium (End of Medium)	
0011010	1A	26	SUB	Replacement for a faulty sign (Substitute)	
0011011	1B	27	ESC	Initiates an escape sequence and thus gives the following characters a special meaning (Escape)	

0011100	1C	28	FS	Marks the separation of logical data blocks and is hierarchically ordered: file as the largest unit, file as the smallest unit.(File Separator, Group Separator, Record Separator, Unit Separator)	
0011101	1D	29	GS		
0011110	1E	30	RS		
0011111	1F	31	US		
0100000	20	32	SP	Blank space (Space)	Special Character
0100001	21	33	!	Exclamation mark	
0100010	22	34	"	Only quotes above	
0100011	23	35	#	Pound sign	
0100100	24	36	$	Dollar sign	
0100101	25	37	%	Percentage sign	
0100110	26	38	&	Commericial and	
0100111	27	39	'	Apostrophe	
0101000	28	40	(	Left bracket	
0101001	29	41	)	Right bracket	
0101010	2A	42	*	Asterisk	
0101011	2B	43	+	Plus symbol	
0101100	2C	44	,	Comma	
0101101	2D	45	-	Dash	
0101110	2E	46	.	Full stop	
0101111	2F	47	/	Forward slash	
0110000	30	48	0		Numbers
0110001	31	49	1		

0110010	32	50	2		
0110011	33	51	3		
0110100	34	52	4		
0110101	35	53	5		
0110110	36	54	6		
0110111	37	55	7		
0111000	38	56	8		
0111001	39	57	9		
0111010	3A	58	:	Colon	Special characters
0111011	3B	59	;	Semicolon	
0111100	3C	60	<	Small than bracket	
0111101	3D	61	=	Equals sign	
0111110	3E	62	>	Bigger than symbol	
0111111	3F	63	?	Question mark	
1000000	40	64	@	At symbol	
1000001	41	65	A		Capital letters
1000010	42	66	B		
1000011	43	67	C		
1000100	44	68	D		
1000101	45	69	E		
1000110	46	70	F		
1000111	47	71	G		

1001000	48	72	H		
1001001	49	73	I		
1001010	4A	74	J		
1001011	4B	75	K		
1001100	4C	76	L		
1001101	4D	77	M		
1001110	4E	78	N		
1001111	4F	79	O		
1010000	50	80	P		
1010001	51	81	Q		
1010010	52	82	R		
1010011	53	83	S		
1010100	54	84	T		
1010101	55	85	U		
1010110	56	86	V		
1010111	57	87	W		
1011000	58	88	X		
1011001	59	89	Y		
1011010	5A	90	Z		
1011011	5B	91	[	Left square bracket	Special character
1011100	5C	92	\	Inverse/backward slash	
1011101	5D	93	]	Right square bracket	

1011110	5E	94	^	Circumflex	
1011111	5F	95	_	Underscore	
1100000	60	96	`	Gravis (backtick)	
1100001	61	97	a		Lowercase letters
1100010	62	98	b		
1100011	63	99	c		
1100100	64	100	d		
1100101	65	101	e		
1100110	66	102	f		
1100111	67	103	g		
1101000	68	104	h		
1101001	69	105	i		
1101010	6A	106	j		
1101011	6B	107	k		
1101100	6C	108	l		
1101101	6D	109	m		
1101110	6E	110	n		
1101111	6F	111	o		
1110000	70	112	p		
1110001	71	113	q		
1110010	72	114	r		
1110011	73	115	s		

1110100	74	116	t		
1110101	75	117	u		
1110110	76	118	v		
1110111	77	119	w		
1111000	78	120	x		
1111001	79	121	y		
1111010	7A	122	z		
1111011	7B	123	{	Left curly bracket	Special characters
1111100	7C	124	|	Vertical line	
1111101	7D	125	}	Right curly brackets	
1111110	7E	126	~	Tilde	
1111111	7F	127	DEL	Deletes a character. Since this control character consists of the same number on all positions, during the typewriter era it was possible to invalidate another character by punching out all the positions (Delete)	Control characters