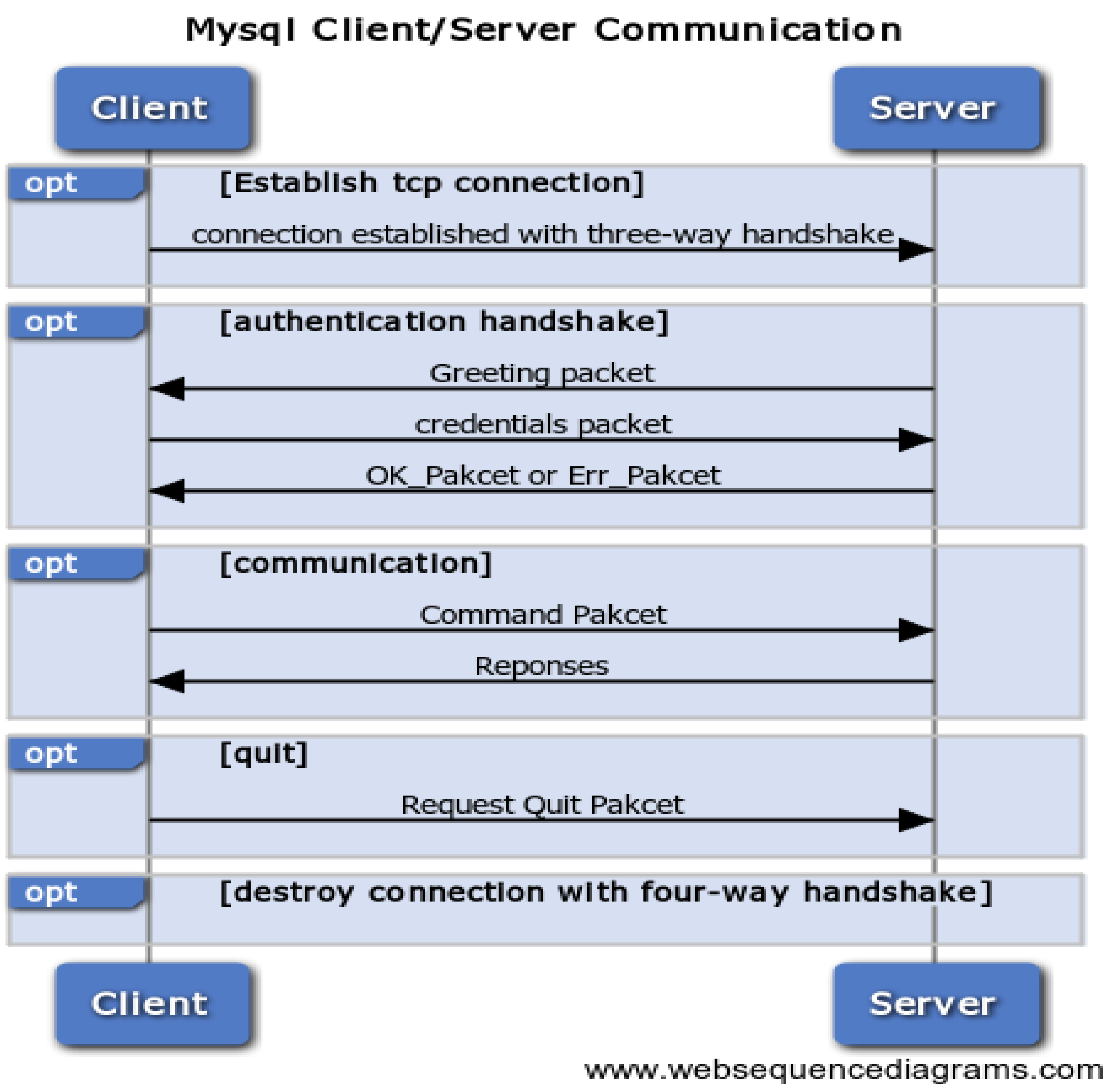
Mysql Client/Server Protocol

(based on 5.6.17)

Overview


Mysql Client/Server Communication
Client
Server
opt
[Establish tcp connection]
connection established with three-way handshake
opt
[authentication handshake]
Greeting packet
credentials packet
OK_Pakcet or Err_Pakcet
opt
[communication]
Command Pakcet
Reponses
opt
[quit]
Request Quit Pakcet
opt
[destroy connection with four-way handshake]
Client
Server
www.websequencediagrams.com

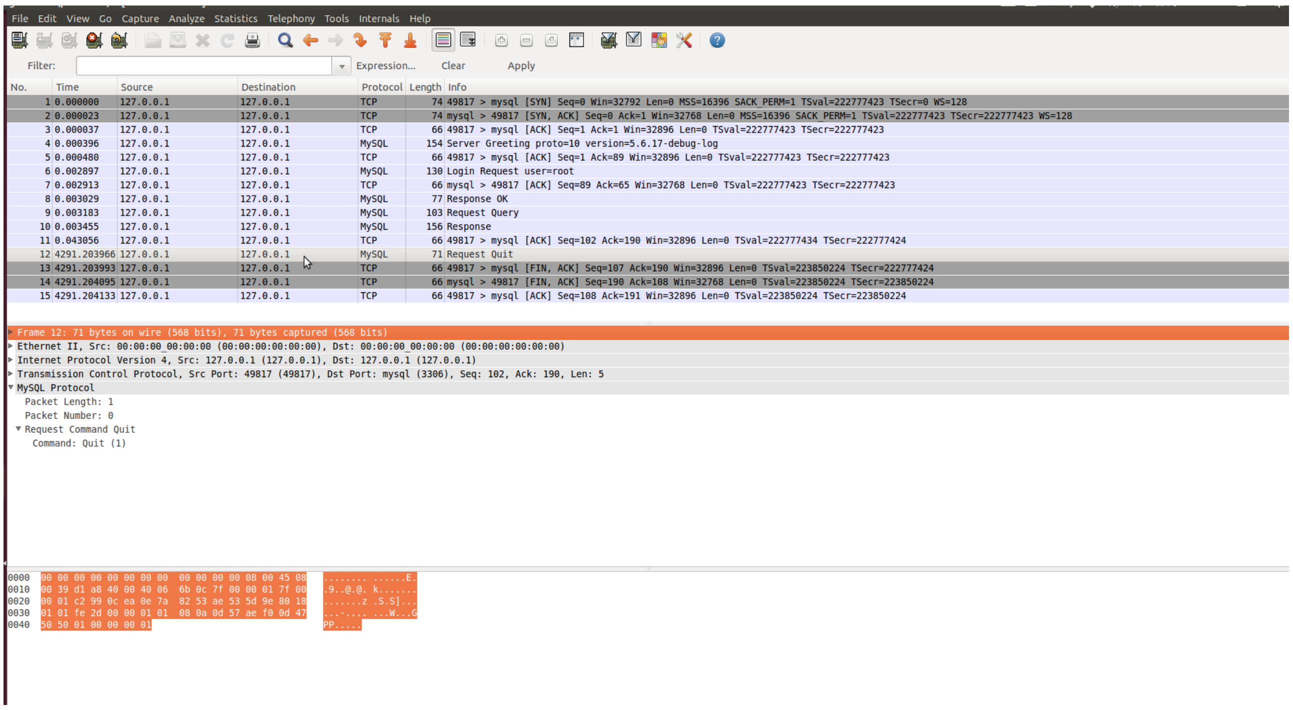
The server listens for connections on a TCP/IP port or a local socket.

When a client connects, a handshake and authentication are performed. If successful,the session begins.

The client sends a command,and the server responds with a data set or a message appropriate for the type of command that was sent.

When the client is finished,it sends a special command telling the server it is done,and the session is terminated.

TCP Dump


File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Clear
Apply

```
No.  Time         Source     Destination  Protocol Length Info
   1 0.000000     127.0.0.1  127.0.0.1    TCP         74 49817 > mysql [SYN] Seq=0 Win=32792 Len=0 MSS=16396 SACK_PERM=1 TSval=222777423 TSecr=0 WS=128
   2 0.000023     127.0.0.1  127.0.0.1    TCP         74 mysql > 49817 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=16396 SACK_PERM=1 TSval=222777423 TSecr=222777423 WS=128
   3 0.000037     127.0.0.1  127.0.0.1    TCP         66 49817 > mysql [ACK] Seq=1 Ack=1 Win=32896 Len=0 TSval=222777423 TSecr=222777423
   4 0.000396     127.0.0.1  127.0.0.1    MySQL      154 Server Greeting proto=10 version=5.6.17-debug-log
   5 0.000480     127.0.0.1  127.0.0.1    TCP         66 49817 > mysql [ACK] Seq=1 Ack=89 Win=32896 Len=0 TSval=222777423 TSecr=222777423
   6 0.002897     127.0.0.1  127.0.0.1    MySQL      130 Login Request user=root
   7 0.002913     127.0.0.1  127.0.0.1    TCP         66 mysql > 49817 [ACK] Seq=89 Ack=65 Win=32768 Len=0 TSval=222777423 TSecr=222777423
   8 0.003029     127.0.0.1  127.0.0.1    MySQL       77 Response OK
   9 0.003183     127.0.0.1  127.0.0.1    MySQL      103 Request Query
  10 0.003455     127.0.0.1  127.0.0.1    MySQL      156 Response
  11 0.043056     127.0.0.1  127.0.0.1    TCP         66 49817 > mysql [ACK] Seq=102 Ack=190 Win=32896 Len=0 TSval=222777434 TSecr=222777424
  12 4291.203966  127.0.0.1  127.0.0.1    MySQL       71 Request Quit
  13 4291.203993  127.0.0.1  127.0.0.1    TCP         66 49817 > mysql [FIN, ACK] Seq=107 Ack=190 Win=32896 Len=0 TSval=223850224 TSecr=222777424
  14 4291.204095  127.0.0.1  127.0.0.1    TCP         66 mysql > 49817 [FIN, ACK] Seq=190 Ack=108 Win=32768 Len=0 TSval=223850224 TSecr=223850224
  15 4291.204133  127.0.0.1  127.0.0.1    TCP         66 49817 > mysql [ACK] Seq=108 Ack=191 Win=32896 Len=0 TSval=223850224 TSecr=223850224

▶ Frame 12: 71 bytes on wire (568 bits), 71 bytes captured (568 bits)
▶ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: 49817 (49817), Dst Port: mysql (3306), Seq: 102, Ack: 190, Len: 5
▼ MySQL Protocol
    Packet Length: 1
    Packet Number: 0
  ▼ Request Command Quit
      Command: Quit (1)

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 39 d1 a8 40 00 40 06  6b 0c 7f 00 00 01 7f 00   .9..@.@. k.......
0020  00 01 c2 99 0c ea 0e 7a  82 53 ae 53 5d 9e 80 18   .......z .S.S]...
0030  01 01 fe 2d 00 00 01 01  08 0a 0d 57 ae f0 0d 47   ...-.... ...W...G
0040  50 50 01 00 00 00 01                               PP.....
```

Process flow

Step1:Establish tcp connection

1. Establish tcp connection with three-way handshake

Step2:Mysql:Authentication Handshake

1. Server->Client:Handshake initialization packet(greeting packet)
2. Client->Server:Response with a credentials packet
3. Server->Client: accepts connection with OK_Packet or reject it with ERR_Packet

Step 3:Mysql:Communication between client and server

1. Client->Server:Command packet
2. Server->Client:Server Responses including:
   - ➢ OK Packet
   - ➢ Error Pakcet
   - ➢ EOF Pakcet
   - ➢ Result Set Pakcets

Step 4:mysql Quit

Client->Server: Request Quit

Step 5:Destroy Connection

Destroy Connection with four-way handshake

Pakcet Format

Categories

Two types of packets depends on the capabilities during handshake stage

- ➢ compressed
- ➢ noncompressed

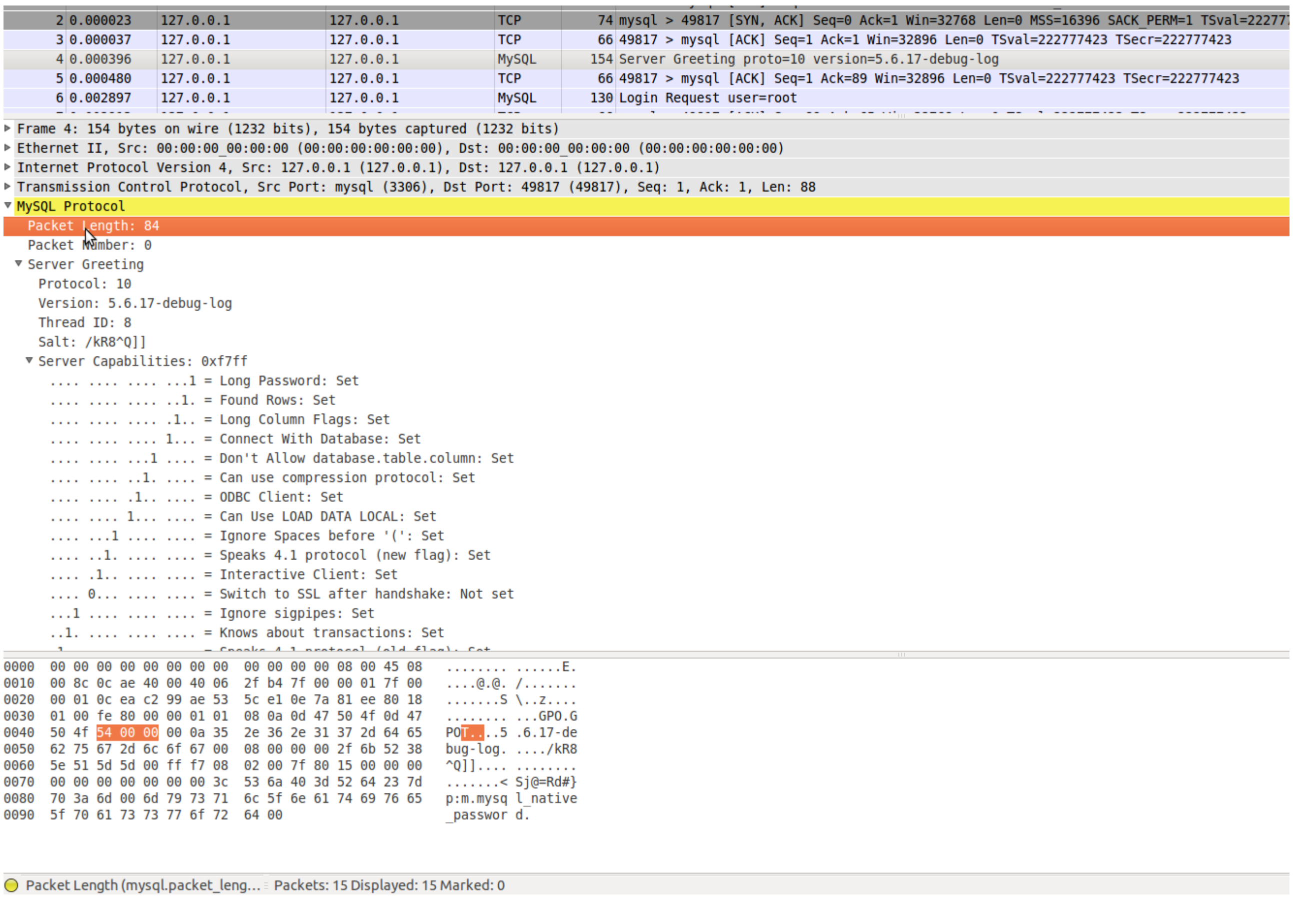
Two categories divided by sender

- ➢ commands sent by client
- ➢ responses returned by the server, divided into four categories:
  - data packets,
  - end-of-data-stream packets
  - success report(OK) packets
  - Error message packets

Common Header

Common four-byte header for uncompressed packets:

Offset	Length	Description
0	3	Packet body length stored(16MB)
3	1	Packet sequence number. The sequence numbers are rest with each new command.



```
    2 0.000023  127.0.0.1        127.0.0.1        TCP       74 mysql > 49817 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=16396 SACK_PERM=1 TSval=22277
    3 0.000037  127.0.0.1        127.0.0.1        TCP       66 49817 > mysql [ACK] Seq=1 Ack=1 Win=32896 Len=0 TSval=222777423 TSecr=222777423
    4 0.000396  127.0.0.1        127.0.0.1        MySQL    154 Server Greeting proto=10 version=5.6.17-debug-log
    5 0.000480  127.0.0.1        127.0.0.1        TCP       66 49817 > mysql [ACK] Seq=1 Ack=89 Win=32896 Len=0 TSval=222777423 TSecr=222777423
    6 0.002897  127.0.0.1        127.0.0.1        MySQL    130 Login Request user=root
```

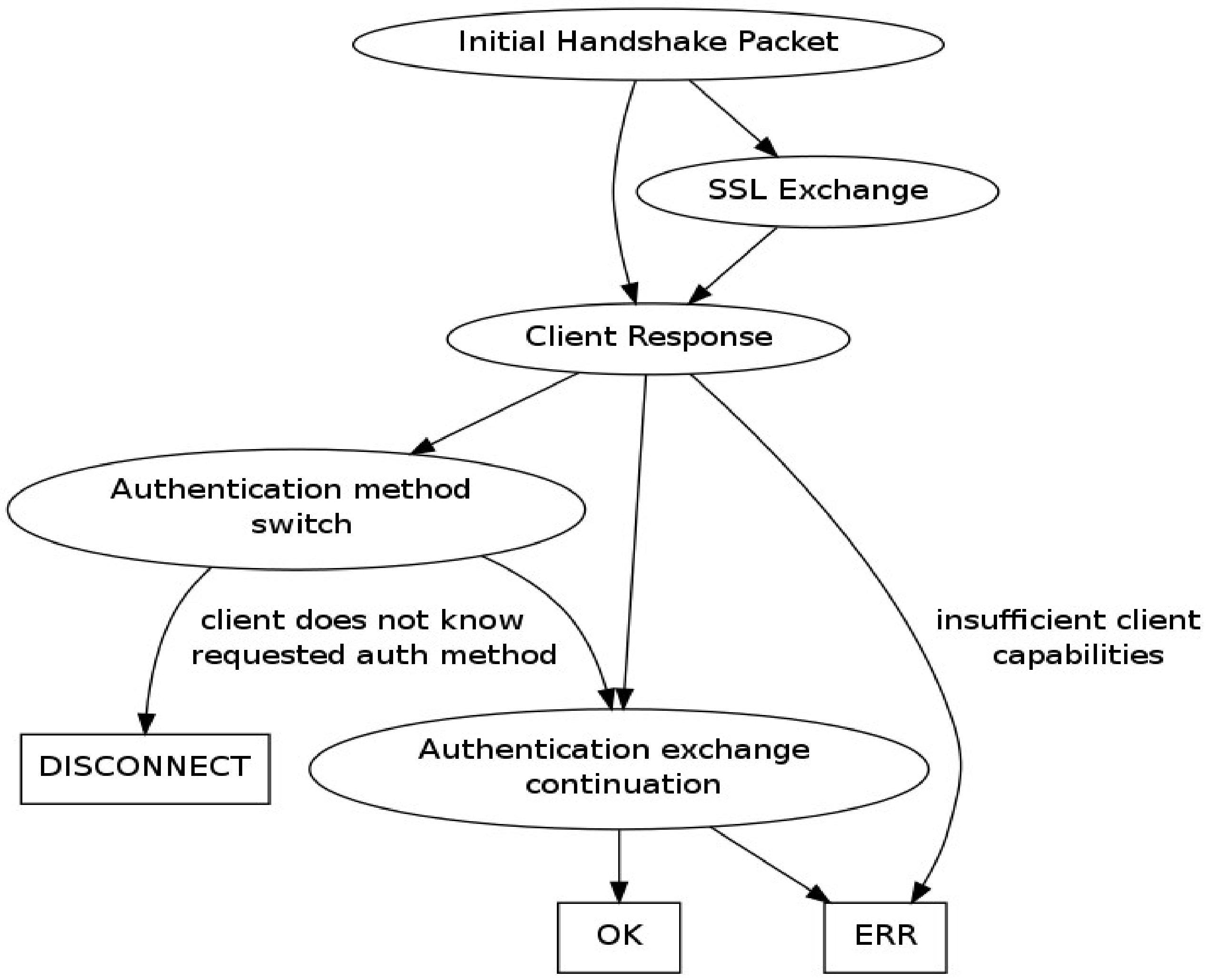
```
▶ Frame 4: 154 bytes on wire (1232 bits), 154 bytes captured (1232 bits)
▶ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▶ Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 49817 (49817), Seq: 1, Ack: 1, Len: 88
▼ MySQL Protocol
    Packet Length: 84
    Packet Number: 0
  ▼ Server Greeting
      Protocol: 10
      Version: 5.6.17-debug-log
      Thread ID: 8
      Salt: /kR8^Q]]
    ▼ Server Capabilities: 0xf7ff
        .... .... .... ...1 = Long Password: Set
        .... .... .... ..1. = Found Rows: Set
        .... .... .... .1.. = Long Column Flags: Set
        .... .... .... 1... = Connect With Database: Set
        .... .... ...1 .... = Don't Allow database.table.column: Set
        .... .... ..1. .... = Can use compression protocol: Set
        .... .... .1.. .... = ODBC Client: Set
        .... .... 1... .... = Can Use LOAD DATA LOCAL: Set
        .... ...1 .... .... = Ignore Spaces before '(': Set
        .... ..1. .... .... = Speaks 4.1 protocol (new flag): Set
        .... .1.. .... .... = Interactive Client: Set
        .... 0... .... .... = Switch to SSL after handshake: Not set
        ...1 .... .... .... = Ignore sigpipes: Set
        ..1. .... .... .... = Knows about transactions: Set
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 8c 0c ae 40 00 40 06  2f b4 7f 00 00 01 7f 00   ....@.@. /.......
0020  00 01 0c ea c2 99 ae 53  5c e1 0e 7a 81 ee 80 18   .......S \..z....
0030  01 00 fe 80 00 00 01 01  08 0a 0d 47 50 4f 0d 47   ........ ...GPO.G
0040  50 4f 54 00 00 00 0a 35  2e 36 2e 31 37 2d 64 65   POT.....5 .6.17-de
0050  62 75 67 2d 6c 6f 67 00  08 00 00 00 2f 6b 52 38   bug-log. ..../kR8
0060  5e 51 5d 5d 00 ff f7 08  02 00 7f 80 15 00 00 00   ^Q]].... ........
0070  00 00 00 00 00 00 00 3c  53 6a 40 3d 52 64 23 7d   .......< Sj@=Rd#}
0080  70 3a 6d 00 6d 79 73 71  6c 5f 6e 61 74 69 76 65   p:m.mysq l_native
0090  5f 70 61 73 73 77 6f 72  64 00                     _passwor d.
```

Packet Length (mysql.packet_leng... Packets: 15 Displayed: 15 Marked: 0

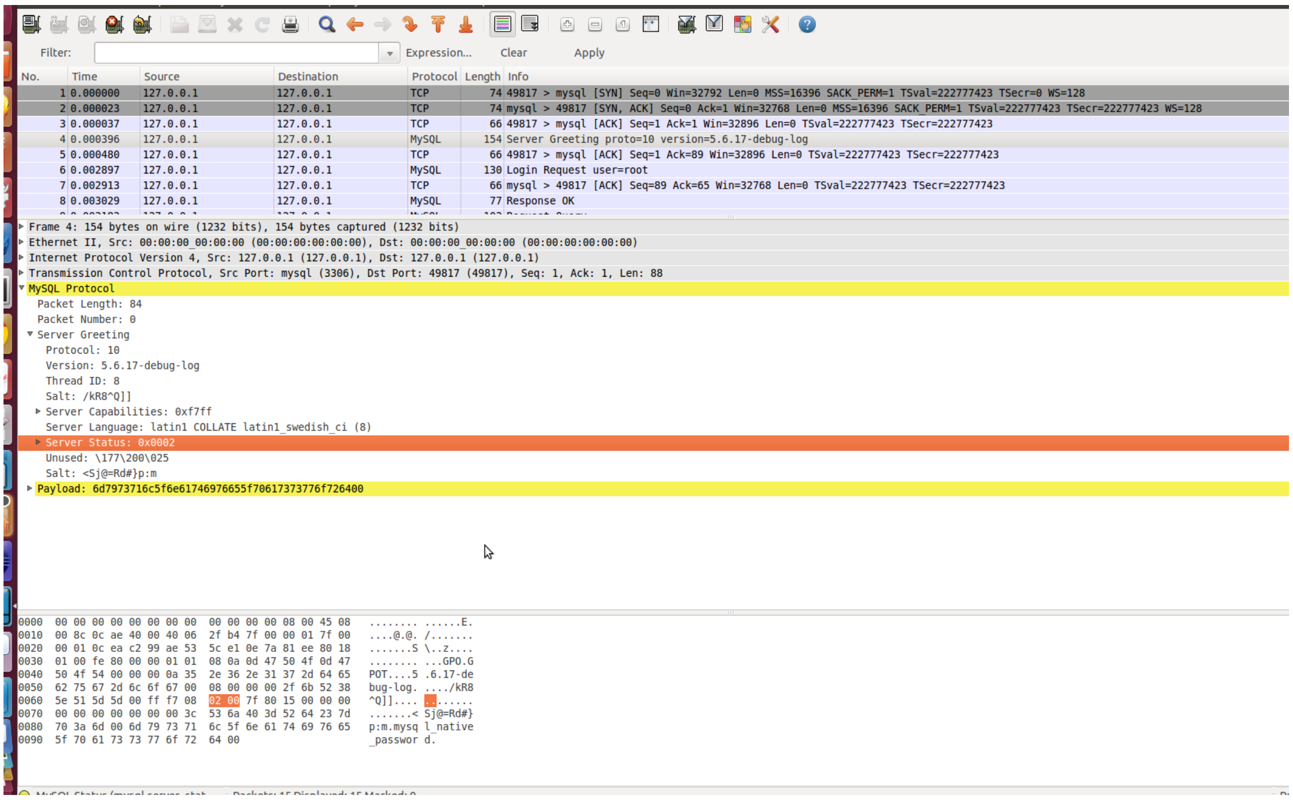
A compressed packet will have an additional 3-byte field,containing the length of the compressed packet body part that follows.

Authentication Handshake


Initial Handshake Packet
SSL Exchange
Client Response
Authentication method switch
client does not know requested auth method
insufficient client capabilities
DISCONNECT
Authentication exchange continuation
OK
ERR

The session between a client and a server begins with an authenticating handshake.
Before it can begin, the server checks whether the host that the client is connecting from is even allowed to connect to this server. If it is not, an error message packet is sent to the client notifying it that the host is not allowed to connect.

Greeting Packet



```
No.  Time      Source     Destination  Protocol Length Info
   1 0.000000  127.0.0.1  127.0.0.1    TCP        74 49817 > mysql [SYN] Seq=0 Win=32792 Len=0 MSS=16396 SACK_PERM=1 TSval=222777423 TSecr=0 WS=128
   2 0.000023  127.0.0.1  127.0.0.1    TCP        74 mysql > 49817 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=16396 SACK_PERM=1 TSval=222777423 TSecr=222777423 WS=128
   3 0.000037  127.0.0.1  127.0.0.1    TCP        66 49817 > mysql [ACK] Seq=1 Ack=1 Win=32896 Len=0 TSval=222777423 TSecr=222777423
   4 0.000396  127.0.0.1  127.0.0.1    MySQL     154 Server Greeting proto=10 version=5.6.17-debug-log
   5 0.000480  127.0.0.1  127.0.0.1    TCP        66 49817 > mysql [ACK] Seq=1 Ack=89 Win=32896 Len=0 TSval=222777423 TSecr=222777423
   6 0.002897  127.0.0.1  127.0.0.1    MySQL     130 Login Request user=root
   7 0.002913  127.0.0.1  127.0.0.1    TCP        66 mysql > 49817 [ACK] Seq=89 Ack=65 Win=32768 Len=0 TSval=222777423 TSecr=222777423
   8 0.003029  127.0.0.1  127.0.0.1    MySQL      77 Response OK
```

```
▶ Frame 4: 154 bytes on wire (1232 bits), 154 bytes captured (1232 bits)
▶ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 49817 (49817), Seq: 1, Ack: 1, Len: 88
▼ MySQL Protocol
    Packet Length: 84
    Packet Number: 0
  ▼ Server Greeting
      Protocol: 10
      Version: 5.6.17-debug-log
      Thread ID: 8
      Salt: /kR8^Q]]
    ▶ Server Capabilities: 0xf7ff
      Server Language: latin1 COLLATE latin1_swedish_ci (8)
    ▶ Server Status: 0x0002
      Unused: \177\200\025
      Salt: <Sj@=Rd#}p:m
  ▶ Payload: 6d7973716c5f6e61746976655f70617373776f726400
```

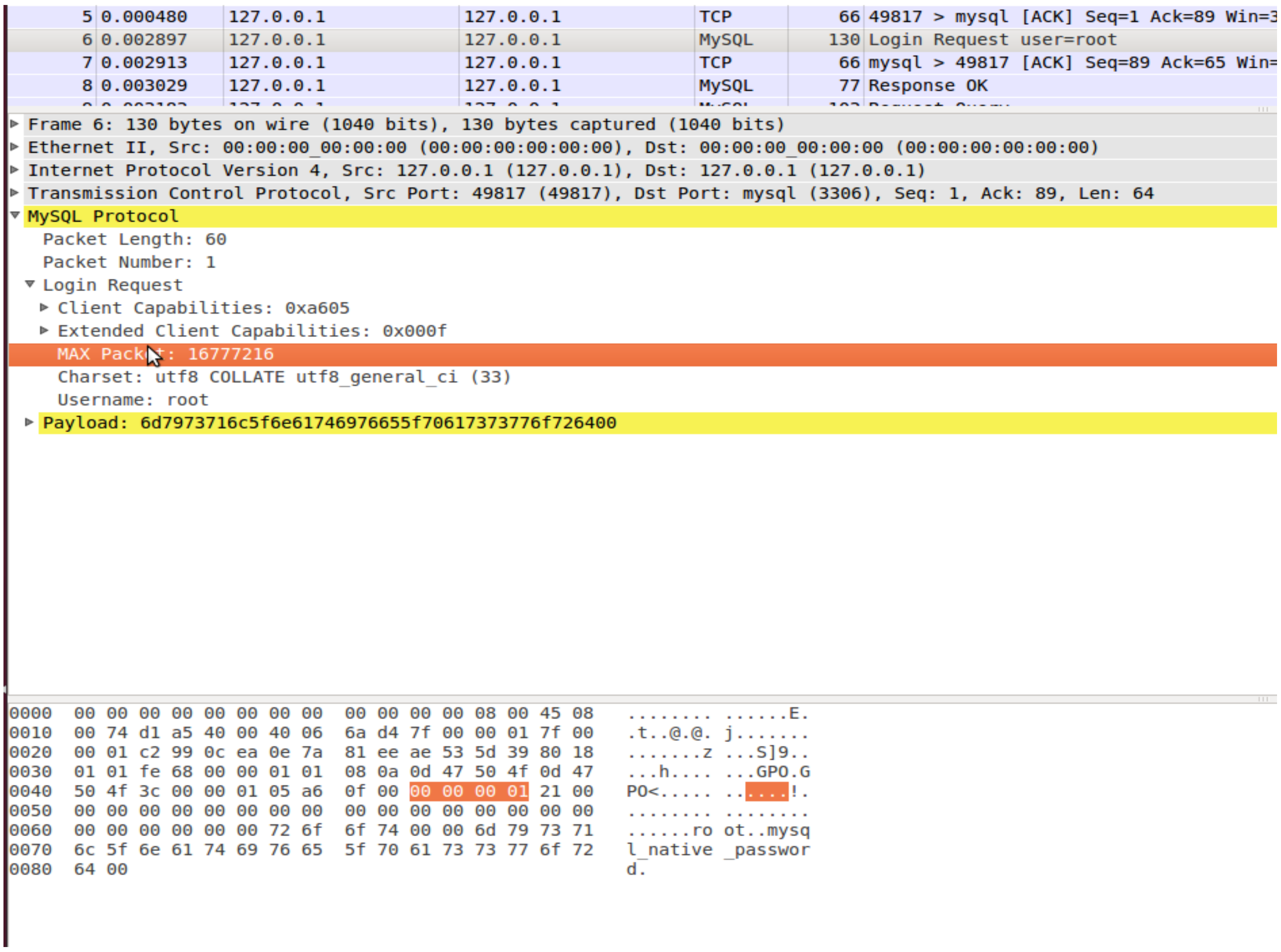
```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 8c 0c ae 40 00 40 06  2f b4 7f 00 00 01 7f 00   ....@.@. /.......
0020  00 01 0c ea c2 99 ae 53  5c e1 0e 7a 81 ee 80 18   .......S \..z....
0030  01 00 fe 80 00 00 01 01  08 0a 0d 47 50 4f 0d 47   ........ ...GPO.G
0040  50 4f 54 00 00 00 0a 35  2e 36 2e 31 37 2d 64 65   POT....5 .6.17-de
0050  62 75 67 2d 6c 6f 67 00  08 00 00 00 2f 6b 52 38   bug-log. ..../kR8
0060  5e 51 5d 5d 00 ff f7 08  02 00 7f 80 15 00 00 00   ^Q]].... ........
0070  00 00 00 00 00 00 00 3c  53 6a 40 3d 52 64 23 7d   .......< Sj@=Rd#}
0080  70 3a 6d 00 6d 79 73 71  6c 5f 6e 61 74 69 76 65   p:m.mysq l_native
0090  5f 70 61 73 73 77 6f 72  64 00                     _passwor d.
```

In the case of successful host verification, the server sends a greeting packet with the standard 4-byte header, the packet sequence number set to 0, and the body in the format shown bellow:

offset in the body	Length	Description
0	1	Protocol version number.
1	ver_len = strlen(server_version) + 1	Zero-terminated server version string.
ver_len+1	4	Thread ID. Internal Mysql ID of the thread that is handling this connection.
ver_len + 5	9	Salt. Zero-terminated string: the first 8 bytes of the 20-byte rand seed string(caused by the version compatibility)
ver_len+14	2	Server Capabilities
ver_len+16	1	Server Languate.Default character set code,or more precisely,the code of the default collation
ver_len+17	2	Server Status.Reports whether the server is in

		transaction or autocommit mode.if there are additional results from a multisatement query,or if a good index(or some index) was used for query optimization...
ver_len+19	13	Unused.Reserved for future use
ver_len + 32	13	Salt:the rest of the random seed string terminated with a zero byte.

Credentials packet



```
    5 0.000480   127.0.0.1        127.0.0.1        TCP      66 49817 > mysql [ACK] Seq=1 Ack=89 Win=3
    6 0.002897   127.0.0.1        127.0.0.1        MySQL   130 Login Request user=root
    7 0.002913   127.0.0.1        127.0.0.1        TCP      66 mysql > 49817 [ACK] Seq=89 Ack=65 Win=
    8 0.003029   127.0.0.1        127.0.0.1        MySQL    77 Response OK
```

```
▸ Frame 6: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits)
▸ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▸ Transmission Control Protocol, Src Port: 49817 (49817), Dst Port: mysql (3306), Seq: 1, Ack: 89, Len: 64
▾ MySQL Protocol
   Packet Length: 60
   Packet Number: 1
  ▾ Login Request
   ▸ Client Capabilities: 0xa605
   ▸ Extended Client Capabilities: 0x000f
     MAX Packet: 16777216
     Charset: utf8 COLLATE utf8_general_ci (33)
     Username: root
  ▸ Payload: 6d7973716c5f6e61746976655f70617373776f726400
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 74 d1 a5 40 00 40 06  6a d4 7f 00 00 01 7f 00   .t..@.@. j.......
0020  00 01 c2 99 0c ea 0e 7a  81 ee ae 53 5d 39 80 18   .......z ...S]9..
0030  01 01 fe 68 00 00 01 01  08 0a 0d 47 50 4f 0d 47   ...h.... ...GPO.G
0040  50 4f 3c 00 00 01 05 a6  0f 00 00 00 00 01 21 00   PO<..... ......!.
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 72 6f  6f 74 00 00 6d 79 73 71   ......ro ot..mysq
0070  6c 5f 6e 61 74 69 76 65  5f 70 61 73 73 77 6f 72   l_native _passwor
0080  64 00                                              d.
```

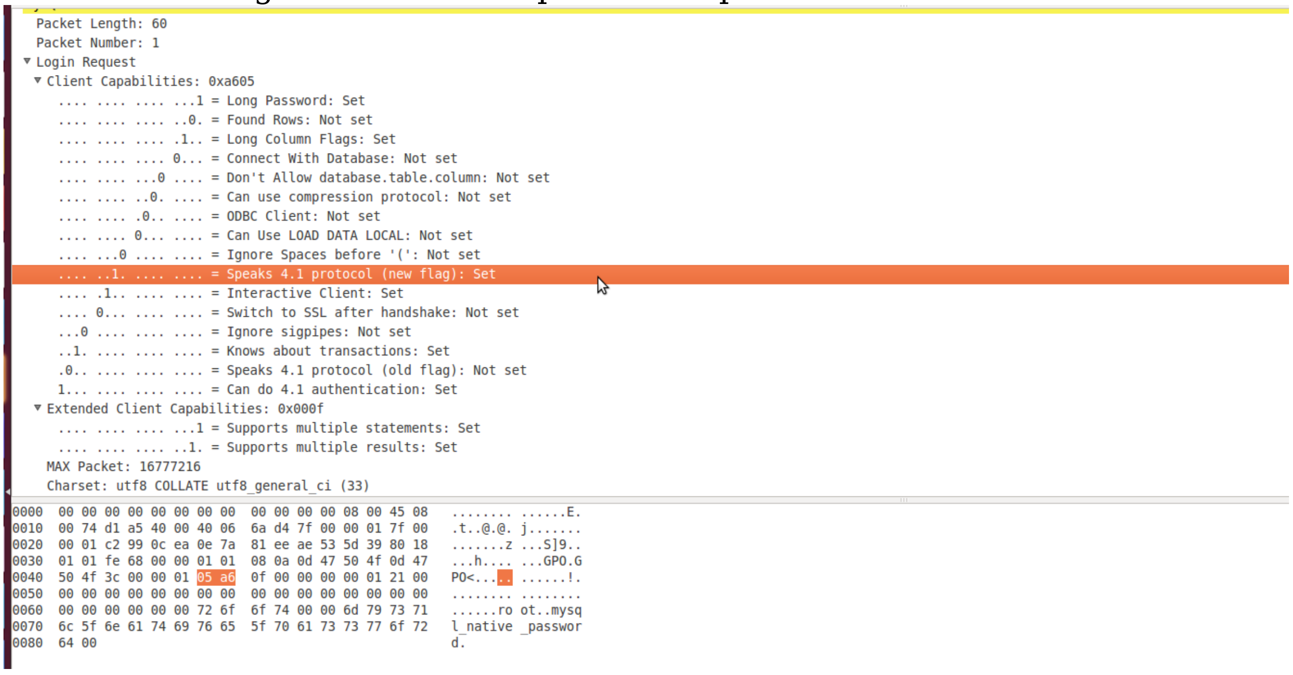
The client responds with a credentials packet

Offset in the body	Length	Description
0	2	Client capabilities

3	2	Extended Client Capabilities.
5	4	Max Packet. Maximum packet length that the client is willing to send or receive.Zero values means the client imposes no restrictions of its own in addition to what is already there in protocol
10	1	Charset,default character set (or more precisely,collation) code of the client
12		Username,name of the SQL account which client wants to log in. this string should be interpreted using the character set indicated by character set field

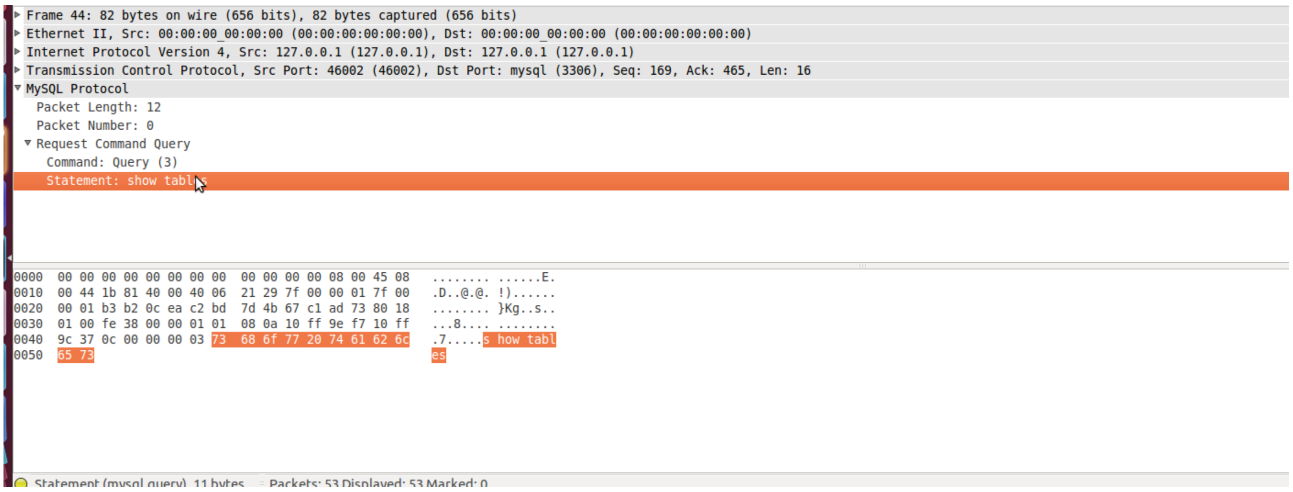
Protocol Capabilities Bit Mask

During the authentication handshake,the client and the server exchange information on what the other is able or willing to do.This enables them to adjust their expectations of their peer and not send the data in some unsupported format. The exchange of information is accomplished through fields containing the bit mask of protocol capabilities.



```
  Packet Length: 60
  Packet Number: 1
▼ Login Request
  ▼ Client Capabilities: 0xa605
      .... .... .... ...1 = Long Password: Set
      .... .... .... ..0. = Found Rows: Not set
      .... .... .... .1.. = Long Column Flags: Set
      .... .... .... 0... = Connect With Database: Not set
      .... .... ...0 .... = Don't Allow database.table.column: Not set
      .... .... ..0. .... = Can use compression protocol: Not set
      .... .... .0.. .... = ODBC Client: Not set
      .... .... 0... .... = Can Use LOAD DATA LOCAL: Not set
      .... ...0 .... .... = Ignore Spaces before '(': Not set
      .... ..1. .... .... = Speaks 4.1 protocol (new flag): Set
      .... .1.. .... .... = Interactive Client: Set
      .... 0... .... .... = Switch to SSL after handshake: Not set
      ...0 .... .... .... = Ignore sigpipes: Not set
      ..1. .... .... .... = Knows about transactions: Set
      .0.. .... .... .... = Speaks 4.1 protocol (old flag): Not set
      1... .... .... .... = Can do 4.1 authentication: Set
  ▼ Extended Client Capabilities: 0x000f
      .... .... .... ...1 = Supports multiple statements: Set
      .... .... .... ..1. = Supports multiple results: Set
    MAX Packet: 16777216
    Charset: utf8 COLLATE utf8_general_ci (33)

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 74 d1 a5 40 00 40 06  6a d4 7f 00 00 01 7f 00   .t..@.@. j.......
0020  00 01 c2 99 0c ea 0e 7a  81 ee ae 53 5d 39 80 18   .......z ...S]9..
0030  01 01 fe 68 00 00 01 01  08 0a 0d 47 50 4f 0d 47   ...h.... ...GPO.G
0040  50 4f 3c 00 00 01 05 a6  0f 00 00 00 00 01 21 00   PO<..... ......!.
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 72 6f  6f 74 00 00 6d 79 73 71   ......ro ot..mysq
0070  6c 5f 6e 61 74 69 76 65  5f 70 61 73 73 77 6f 72   l_native _passwor
0080  64 00                                              d.
```

Command packet



```
▶ Frame 44: 82 bytes on wire (656 bits), 82 bytes captured (656 bits)
▶ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: 46002 (46002), Dst Port: mysql (3306), Seq: 169, Ack: 465, Len: 16
▼ MySQL Protocol
    Packet Length: 12
    Packet Number: 0
  ▼ Request Command Query
      Command: Query (3)
      Statement: show tables

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 44 1b 81 40 00 40 06  21 29 7f 00 00 01 7f 00   .D..@.@. !)......
0020  00 01 b3 b2 0c ea c2 bd  7d 4b 67 c1 ad 73 80 18   ........ }Kg..s..
0030  01 00 fe 38 00 00 01 01  08 0a 10 ff 9e f7 10 ff   ...8.... ........
0040  9c 37 0c 00 00 00 03 73  68 6f 77 20 74 61 62 6c   .7.....s how tabl
0050  65 73                                              es

Statement (mysql query), 11 bytes    Packets: 53 Displayed: 53 Marked: 0
```

Once the authentication is complete,the client begins sending commands to the server using command packets,the body of a command packet is documented as following:

Offset in the body	Length	Description
0	1	Command code
1	For the noncompressed pakcet,total packet length from the header-1.For the compressed packet,the compressed body length-1	The argument of the command,if present.

The Command codes are defined in include/mysql_com.h:

```
/*
  You should add new commands to the end of this list, otherwise old
  servers won't be able to handle them as 'unsupported'.
*/

enum enum_server_command
{
  COM_SLEEP, COM_QUIT, COM_INIT_DB, COM_QUERY, COM_FIELD_LIST,
  COM_CREATE_DB, COM_DROP_DB, COM_REFRESH, COM_SHUTDOWN, COM_STATISTICS,
  COM_PROCESS_INFO, COM_CONNECT, COM_PROCESS_KILL, COM_DEBUG, COM_PING,
  COM_TIME, COM_DELAYED_INSERT, COM_CHANGE_USER, COM_BINLOG_DUMP,
  COM_TABLE_DUMP, COM_CONNECT_OUT, COM_REGISTER_SLAVE,
  COM_STMT_PREPARE, COM_STMT_EXECUTE, COM_STMT_SEND_LONG_DATA, COM_STMT_CLOSE,
  COM_STMT_RESET, COM_SET_OPTION, COM_STMT_FETCH, COM_DAEMON,
  COM_BINLOG_DUMP_GTID,
  /* don't forget to update const char *command_name[] in sql_parse.cc */

  /* Must be last */
  COM_END
};
```

The command-handling logic can be found in the switch statement of

dispatch_commond() in sql/sql_parse.cc,to long to show..
Command Phase

Hex	Constant Name	Argument description	Command Description	Return
00	COM_SLEEP	No argument	Internal server command,Nerver sent by a client	ERR_Packet
01	COM_QUIT	No argument	Tells the server that the client wants to close the connection(mysql_close())	Either a connection close or a OK_Packet
02	COM_INIT_DB	A string containing the name of the database	Change the default schema of the connection(mysql_select_db())	OK_Packet or ERR_Packet
03	COM_QUERY	A string containing the query	Used to send the server a text-based query that is executed immediately(mysql_query).	COM_QUERY_Response(https://dev.mysql.com/doc/internals/en/com-query-response.html)
04	COM_FIELD_LIST	A string containing the name of the table	Get the column definitions of a table(mysql_list_fields)	COM_FIELD_LIST Response(https://dev.mysql.com/doc/internals/en/com-field-list-response.html)
05	COM_CREATE_DB	A string containing the name of the database	Create a new database with the specified name	OK_Packet or ERR_Packet
06	COM_DROP_DB	Schema name	Drop a schema	OK_Packet or ERR_Packet
07	COM_REFRESH	A byte containing the bit mask of reloading	Tells the server to refresh the table	OK_Packet or ERR_Packet

		operations	cache,retate the logs,reread the access control tables,clear the re host name lookup cache,reset the status variables to 0,clear the replication master logs, or reset the replication slave depending on the options in the bit mask, Issued by the Client API call mysql_refresh( )	
08	COM_SHUTDOWN	No argument	Shut down the server(mysql_ shutdown)	EOF_Packet or ERR_Packet
09	COM_STATISTICS	No argument	Tells the server to send back a string containing a brief status report (mysql_stat)	
0a	COM_PROCESS_INFO	No argument	Get a list of active threads	A protocol Text::Resultset or ERR_Packet
0b	COM_CONNECT		Internal server command,Nerver sent by a client	
0c	COM_PROCESS_KILL	A 4-byte integer with the low byte first containing the MySql ID of the thread to	Tells the server to terminate the connection by the argument(mysql_kill).	OK_Packet Or ERR_Packet

		be terminated		
0d	COM_DEBUG	No argument	Tells the server to dump some debugging information into its error log(mysql_dump_debug_info)	
0e	COM_PING	No argument	Tells the server to respond with an OK packet(mysql_ping)	
0f	COM_TIME	No argument	Internal server command,Nerver sent by a client	
10	COM_DELAYED_INSERT		An internal command in server	ERR_Packet
11	COM_CHANGE_USER	•Usename •auth_plugin_data_len •auth_plugin_data •schema •character_set •auth_plugin_name •connect_attrs_len	Change the user of the current connection and reset the connection state.	Authentication Method Switch Request Packet or ERR_Packet
12	COM_BINLOG_DUMP	A byte sequence in the following format:4-byte integer for the offset,2-byte integer for the flags,4-byte	Tells the server to send a continuous feed of the replication master log events starting at the specified offset in the specified log. Used by the	Binlog network stream A ERR_Packet EOF_Packet

		integer for the slave server ID,and a string for the log name.	replication slave,and in the mysqlbinlog command-line utility.	
13	COM_TABLE_DUMP	database_len database name table_len tablename	Dump a table	A table dump or ERR_Packet
14	COM_CONNECT_OUT		Internal command in the server	
15	COM_REGISTER_SLAVE	A byte sequence in the following format:4-byte integer for the server ID,then a sequence of 1 byte-length prefixed strings in the following order:slave hostname,slave user to connect as,slave user password.Then a 2-byte slave user port,4-byte replication recovery rank,and another 4-	Register a slave at the master.Should be sent before requesting a binlog events	

		byte field that is currently unused.		
16	COM_STMT_PREPARE	A string containing the statement	Tells the server to prepare the statement specified by the argument(mysql_stmt_prepare).	COM_STMT_PREPARE_OK on success,ERR_Packet otherwise
17	COM_STMT_EXECUTE	A byte sequence in the following format:4-byte statement ID, 1 byte for flags,and 4-byte iteration count.	Tells the server to execute the statement referenced by the statement ID(mysql_stmt_excute)	COM_STMT_EXECUTE Response
18	COM_STMT_SEND_LONG_DATA		Send the data for a column.Repeating to send it,appends the data to the parameter.	
19	COM_STMT_CLOSE		Deallocates a prepared statement	
1a	COM_STMT_RESET		Reset the data of a prepared statement which was accumulated with COM_STMT_SEND_LONG_DATA	OK_Packet or ERR_Packet
1b	COM_SET_OPTION		Allows to enable and disable Client_multi_statements for the current	

			connection.	
1c	COM_STMT_FETCH		Fetch rows from a existing resultset after a COM_STMT_EXECUTE	
1d	COM_DAEMON		Internal command	
1e	COM_BINLOG_DUMP_GTID		Requet the binlog network stream	A Binlog Network Stream or an ERR_Packet
1f	COM_RESET_CONNECTION		Resets the session state.	

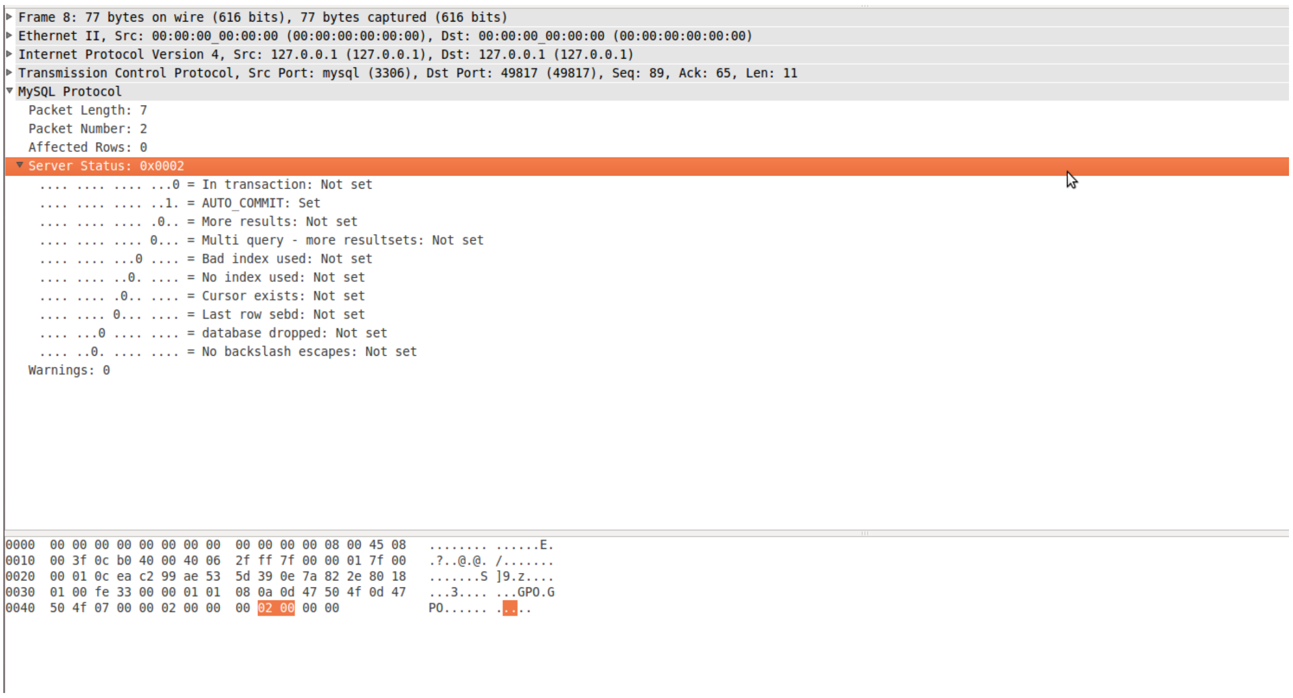
COM_QUERY_Response:https://dev.mysql.com/doc/internals/en/com-query-response.html

Server Responses

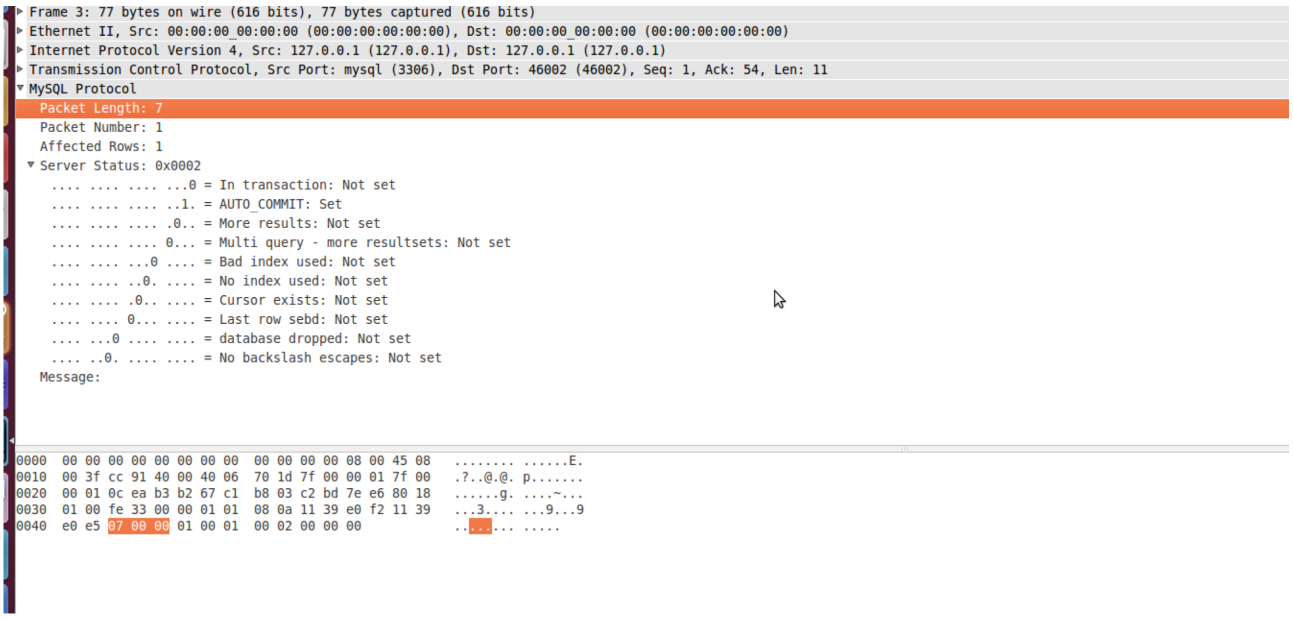
Data Field

TODO

OK Packet



```
▶ Frame 8: 77 bytes on wire (616 bits), 77 bytes captured (616 bits)
▶ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 49817 (49817), Seq: 89, Ack: 65, Len: 11
▼ MySQL Protocol
    Packet Length: 7
    Packet Number: 2
    Affected Rows: 0
  ▼ Server Status: 0x0002
      .... .... .... ...0 = In transaction: Not set
      .... .... .... ..1. = AUTO_COMMIT: Set
      .... .... .... .0.. = More results: Not set
      .... .... .... 0... = Multi query - more resultsets: Not set
      .... .... ...0 .... = Bad index used: Not set
      .... .... ..0. .... = No index used: Not set
      .... .... .0.. .... = Cursor exists: Not set
      .... .... 0... .... = Last row sebd: Not set
      .... ...0 .... .... = database dropped: Not set
      .... ..0. .... .... = No backslash escapes: Not set
    Warnings: 0

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 3f 0c b0 40 00 40 06  2f ff 7f 00 00 01 7f 00   .?..@.@. /.......
0020  00 01 0c ea c2 99 ae 53  5d 39 0e 7a 82 2e 80 18   .......S ]9.z....
0030  01 00 fe 33 00 00 01 01  08 0a 0d 47 50 4f 0d 47   ...3.... ...GPO.G
0040  50 4f 07 00 00 02 00 00  00 02 00 00 00            PO...... .....
```



```
▶ Frame 3: 77 bytes on wire (616 bits), 77 bytes captured (616 bits)
▶ 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 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 46002 (46002), Seq: 1, Ack: 54, Len: 11
▼ MySQL Protocol
    Packet Length: 7
    Packet Number: 1
    Affected Rows: 1
  ▼ Server Status: 0x0002
      .... .... .... ...0 = In transaction: Not set
      .... .... .... ..1. = AUTO_COMMIT: Set
      .... .... .... .0.. = More results: Not set
      .... .... .... 0... = Multi query - more resultsets: Not set
      .... .... ...0 .... = Bad index used: Not set
      .... .... ..0. .... = No index used: Not set
      .... .... .0.. .... = Cursor exists: Not set
      .... .... 0... .... = Last row sebd: Not set
      .... ...0 .... .... = database dropped: Not set
      .... ..0. .... .... = No backslash escapes: Not set
    Message:

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 3f cc 91 40 00 40 06  70 1d 7f 00 00 01 7f 00   .?..@.@. p.......
0020  00 01 0c ea b3 b2 67 c1  b8 03 c2 bd 7e e6 80 18   ......g. ....~...
0030  01 00 fe 33 00 00 01 01  08 0a 11 39 e0 f2 11 39   ...3.... ...9...9
0040  e0 e5 07 00 00 01 00 01  00 02 00 00 00            ........ .....
```

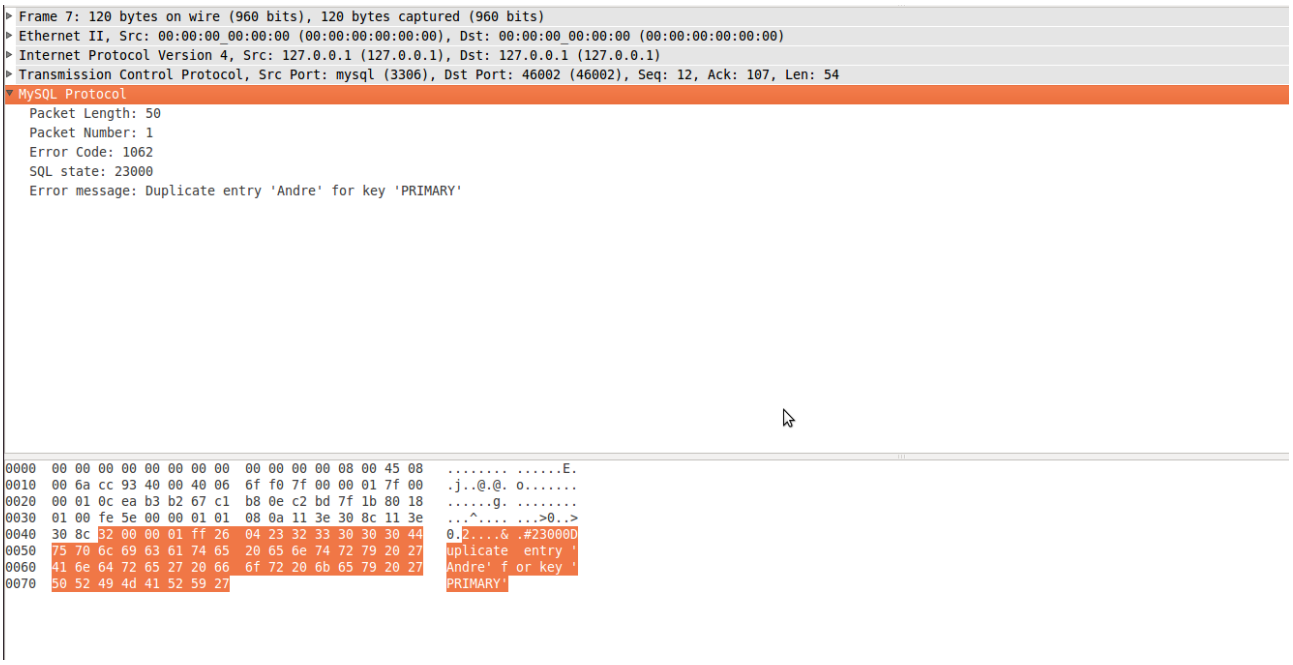
Format of server's OK pakcet:

Offset in the body	Length	Description
0	1	A byte with the value (0) indicating that the packet has no fields
1	rows_len(1byte)	The number of records that the query has changed.
1+rows_len	id_len(1byte)	The value of the generated auto-increment ID for the primary key. Set to 0 if not applicable in the context.
1+rows_len+id_len	2	Server status bit mask
3+rows_len+id_len	2	The number of warnings the last command has generated
5+rows_len+id_len	Msglen	An optional field for the status message if one is present in the standard data field format with the field length followed by field value,which in this case is a character string

Sql/protocol.cc

```
/**
  A default implementation of "OK" packet response to the client.

  Currently this implementation is re-used by both network-oriented
  protocols -- the binary and text one. They do not differ
  in their OK packet format, which allows for a significant simplification
  on client side.
*/

bool Protocol::send_ok(uint server_status, uint statement_warn_count,
                       ulonglong affected_rows, ulonglong last_insert_id,
                       const char *message)
{
  DBUG_ENTER("Protocol::send_ok");
  const bool retval=
    net_send_ok(thd, server_status, statement_warn_count,
                affected_rows, last_insert_id, message);
  DBUG_RETURN(retval);
}
```

Error Packet



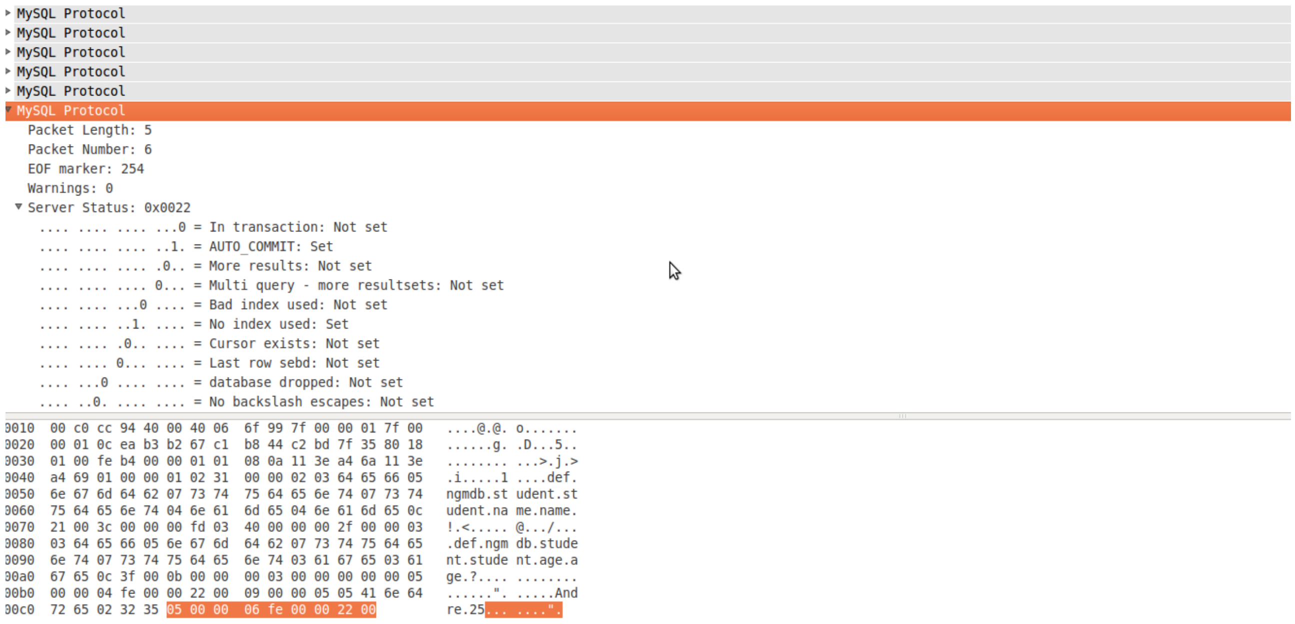
```
▸ Frame 7: 120 bytes on wire (960 bits), 120 bytes captured (960 bits)
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▸ Transmission Control Protocol, Src Port: mysql (3306), Dst Port: 46002 (46002), Seq: 12, Ack: 107, Len: 54
▾ MySQL Protocol
    Packet Length: 50
    Packet Number: 1
    Error Code: 1062
    SQL state: 23000
    Error message: Duplicate entry 'Andre' for key 'PRIMARY'

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 08   ........ ......E.
0010  00 6a cc 93 40 00 40 06  6f f0 7f 00 00 01 7f 00   .j..@.@. o.......
0020  00 01 0c ea b3 b2 67 c1  b8 0e c2 bd 7f 1b 80 18   ......g. ........
0030  01 00 fe 5e 00 00 01 01  08 0a 11 3e 30 8c 11 3e   ...^.... ...>0..>
0040  30 8c 32 00 00 01 ff 26  04 23 32 33 30 30 30 44   0.2....& .#23000D
0050  75 70 6c 69 63 61 74 65  20 65 6e 74 72 79 20 27   uplicate  entry '
0060  41 6e 64 72 65 27 20 66  6f 72 20 6b 65 79 20 27   Andre' f or key '
0070  50 52 49 4d 41 52 59 27                            PRIMARY'
```

When something goes wrong with the processing of a command,the server responds with a error packet

Offset in the body	Length	Description
0	1	A byte containing 255 as an error message
1	2	The error code
3	1	'#' the sql-state marker
4	5	Sql-state

9		Human readable error message

EOF Packet



```
▸ MySQL Protocol
▸ MySQL Protocol
▸ MySQL Protocol
▸ MySQL Protocol
▸ MySQL Protocol
▾ MySQL Protocol
    Packet Length: 5
    Packet Number: 6
    EOF marker: 254
    Warnings: 0
  ▾ Server Status: 0x0022
      .... .... .... ...0 = In transaction: Not set
      .... .... .... ..1. = AUTO_COMMIT: Set
      .... .... .... .0.. = More results: Not set
      .... .... .... 0... = Multi query - more resultsets: Not set
      .... .... ...0 .... = Bad index used: Not set
      .... .... ..1. .... = No index used: Set
      .... .... .0.. .... = Cursor exists: Not set
      .... .... 0... .... = Last row sebd: Not set
      .... ...0 .... .... = database dropped: Not set
      .... ..0. .... .... = No backslash escapes: Not set
```

```
0010  00 c0 cc 94 40 00 40 06  6f 99 7f 00 00 01 7f 00   ....@.@. o.......
0020  00 01 0c ea b3 b2 67 c1  b8 44 c2 bd 7f 35 80 18   ......g. .D...5..
0030  01 00 fe b4 00 00 01 01  08 0a 11 3e a4 6a 11 3e   ........ ...>.j.>
0040  a4 69 01 00 00 01 02 31  00 00 02 03 64 65 66 05   .i.....1 ....def.
0050  6e 67 6d 64 62 07 73 74  75 64 65 6e 74 07 73 74   ngmdb.st udent.st
0060  75 64 65 6e 74 04 6e 61  6d 65 04 6e 61 6d 65 0c   udent.na me.name.
0070  21 00 3c 00 00 00 fd 03  40 00 00 00 2f 00 00 03   !.<..... @.../...
0080  03 64 65 66 05 6e 67 6d  64 62 07 73 74 75 64 65   .def.ngm db.stude
0090  6e 74 07 73 74 75 64 65  6e 74 03 61 67 65 03 61   nt.stude nt.age.a
00a0  67 65 0c 3f 00 0b 00 00  00 03 00 00 00 00 00 05   ge.?.... ........
00b0  00 00 04 fe 00 00 22 00  09 00 00 05 05 41 6e 64   ......". .....And
00c0  72 65 02 32 35 05 00 00  06 fe 00 00 22 00         re.25... ....".
```

The EOF packet is used to communicate a number of messages:

- ➢ End of field information data in a result set
- ➢ End of row data in a result set
- ➢ Server acknowledgement of COM_SHUTDOWN
- ➢ Server reporting success in response to COM_SET_OPTION and COM_DEBUG

Packet Format:

Offset in the body	Length	Description
0	1	Byte with 254 indicate is an EOF packet
1	2	Number of warnings
3	2	Server status bit mask

Result Set Packet

TODO

Refer:https://dev.mysql.com/doc/internals/en/overview.html