C/C++Linux服务器开发

高级架构师课程

三年课程沉淀

五次精益升级

十年行业积累

百个实战项目

十万内容受众

讲师:darren/326873713

扫一扫 升职加薪

班主任:柚子/2690491738

讲师介绍--专业来自专注和实力

Darren老师

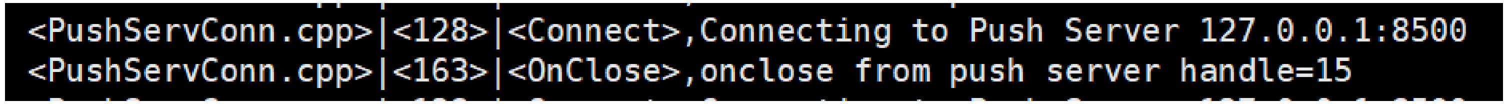
曾供职于国内知名半导体公司（珠海扬智/深圳联发科），曾在某互联网公司担任音视频通话项目经理。主要从事音视频驱动、多媒体中间件、流媒体服务器的开发，开发过即时通讯+音视频通话的大型项目，在音视频、C/C++/GO Linux服务器领域有丰富的实战经验。

3 IM消息服务器和路由服务器设计

1. 登录msg_server拉取数据过程
2. route_server

0. 修改msg_server

修改msg_server



```
<PushServConn.cpp>|<128>|<Connect>,Connecting to Push Server 127.0.0.1:8500
<PushServConn.cpp>|<163>|<OnClose>,onclose from push server handle=15
```

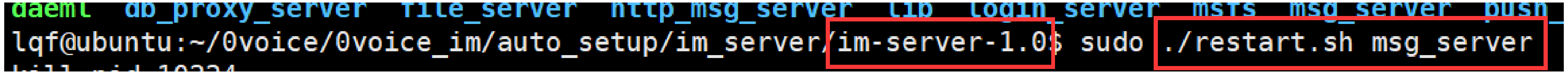
1. 进到0voice_im/server/src/msg_server

（1）把PushServerConn.cpp 128行 163行注释， 避免影响查看其它log

（2）在MsgConn.cpp 272行加上if(pPdu->GetCommandId()!=CID_OTHER_HEARTBEAT) log(“HandlePdu cmd:0x%04x\n”, pPdu->GetCommandId());， 以方便查看服务器收到的客户端信令

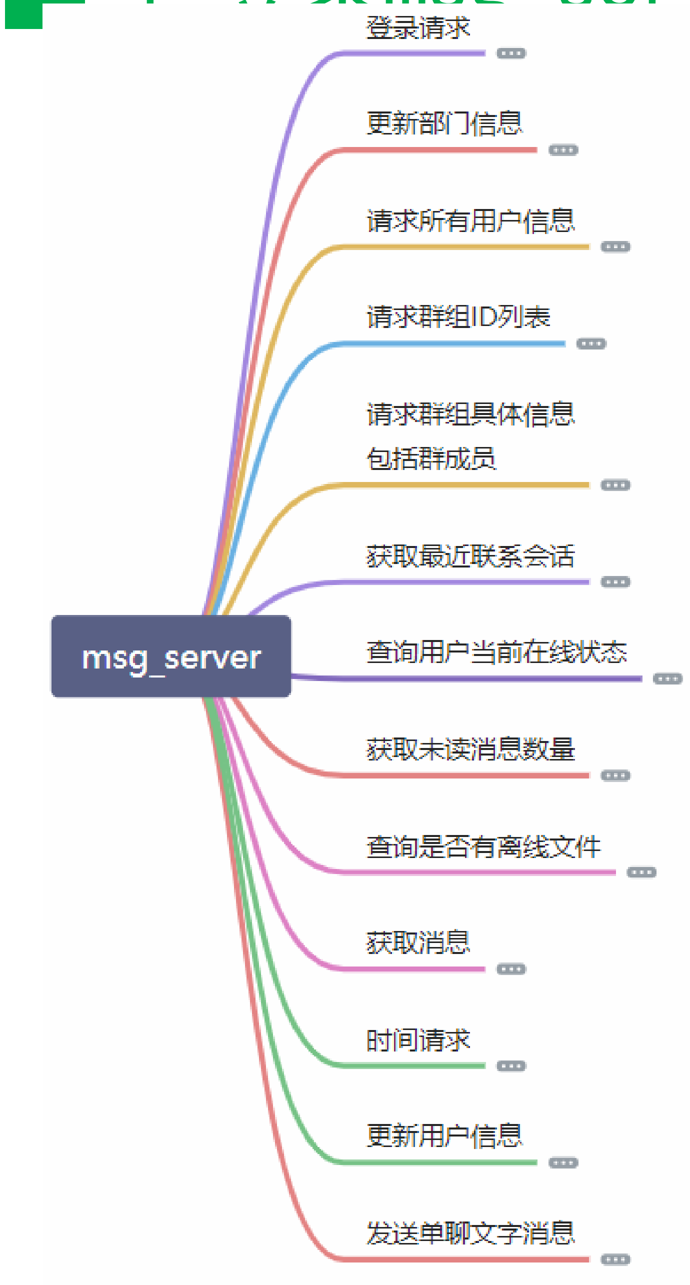
2. sudo make 进行编译得到msg_server， 并将msg_server执行文件拷贝到0voice_im/auto_setup/im_server/im-server-1.0/msg_server

3. 使用sudo ./restart.sh msg_server命令重启msg_server



```
daeml   db_proxy_server   file_server   http_msg_server   lib   login_server   msfs   msg_server   push
lqf@ubuntu:~/0voice/0voice_im/auto_setup/im_server/im-server-1.0$ sudo ./restart.sh msg_server
```

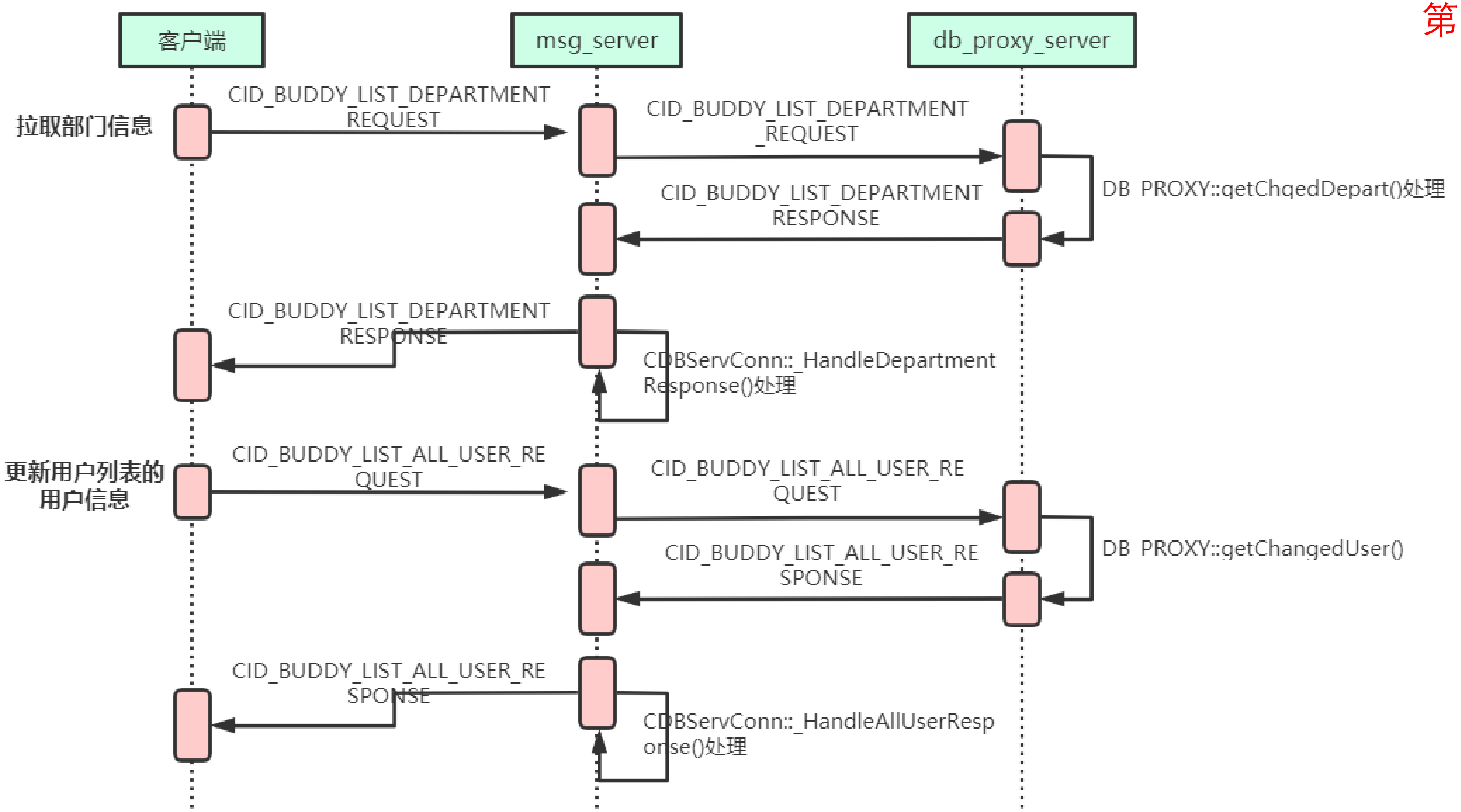
1 登录msg_server拉取数据过程



打开获取完整的思维导图 https://www.yuque.com/docs/share/a086da1e-b8f3-4750-9b00-f2149386fac9?# （密码：yqng）《3-1-思维导图-消息服务器设计》

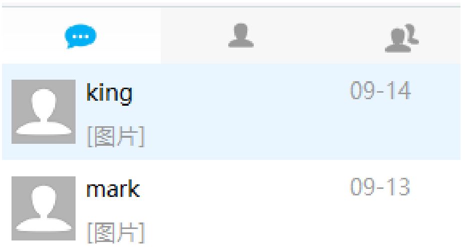
1.0 msg_server登录请求响应过程-更新信息1

第一次全量拉取 1970年
第二次增量拉取



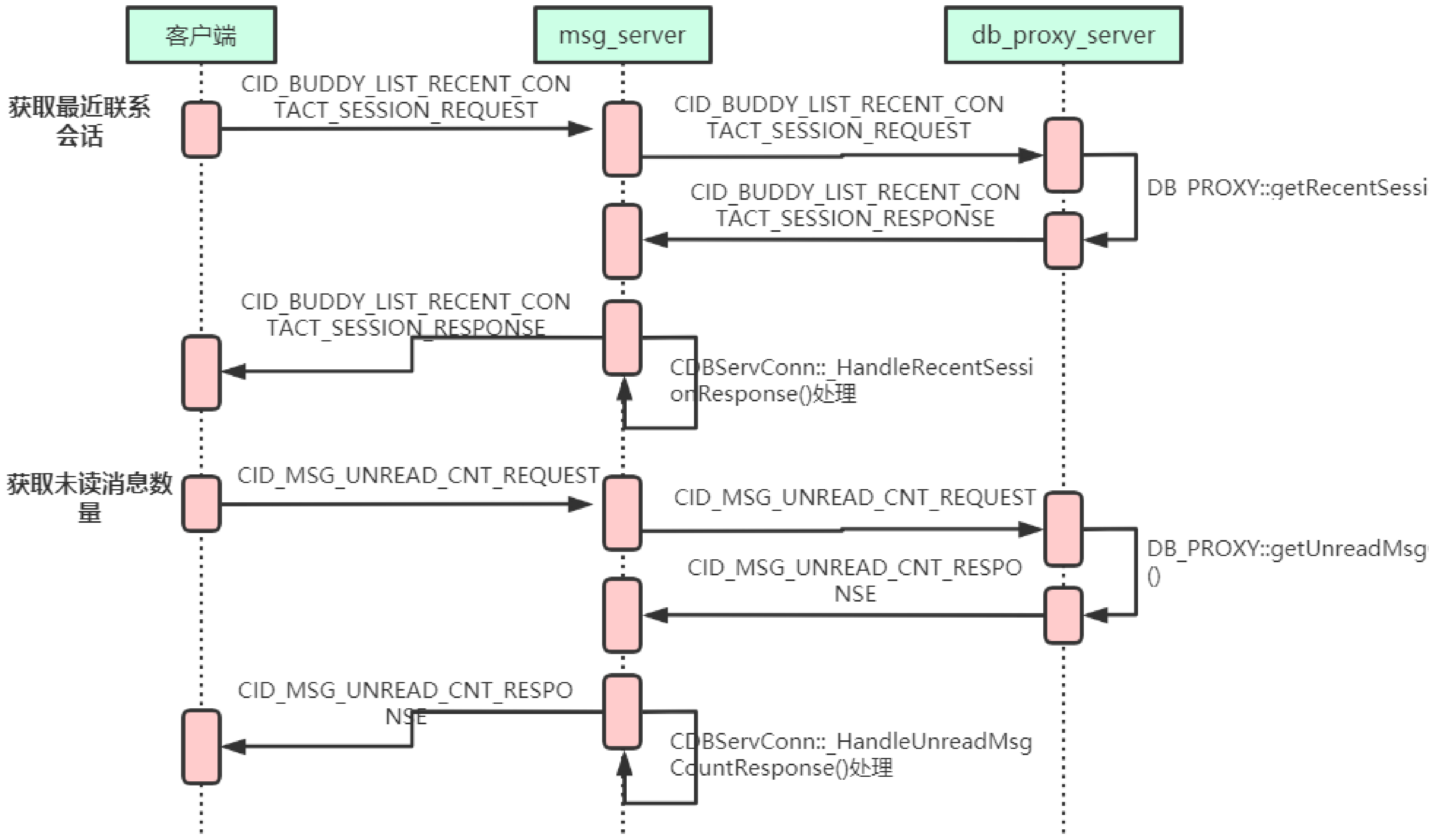
客户端
msg_server
db_proxy_server
拉取部门信息
CID_BUDDY_LIST_DEPARTMENT REQUEST
CID_BUDDY_LIST_DEPARTMENT _REQUEST
DB PROXY::getChgedDepart()处理
CID_BUDDY_LIST_DEPARTMENT RESPONSE
CID_BUDDY_LIST_DEPARTMENT RESPONSE
CDBServConn::_HandleDepartment Response()处理
更新用户列表的用户信息
CID_BUDDY_LIST_ALL_USER_RE QUEST
CID_BUDDY_LIST_ALL_USER_RE QUEST
DB PROXY::getChangedUser()
CID_BUDDY_LIST_ALL_USER_RE SPONSE
CID_BUDDY_LIST_ALL_USER_RE SPONSE
CDBServConn::_HandleAllUserResp onse()处理

1.0 msg_server登录请求响应过程2



king 09-14
[图片]

mark 09-13
[图片]



客户端

msg_server

db_proxy_server

获取最近联系会话

CID_BUDDY_LIST_RECENT_CONTACT_SESSION_REQUEST

CID_BUDDY_LIST_RECENT_CONTACT_SESSION_REQUEST

DB_PROXY::getRecentSession()处理

CID_BUDDY_LIST_RECENT_CONTACT_SESSION_RESPONSE

CID_BUDDY_LIST_RECENT_CONTACT_SESSION_RESPONSE

CDBServConn::_HandleRecentSessionResponse()处理

获取未读消息数量

CID_MSG_UNREAD_CNT_REQUEST

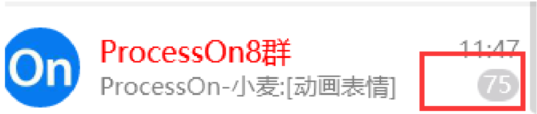
CID_MSG_UNREAD_CNT_REQUEST

DB_PROXY::getUnreadMsgCounter()

CID_MSG_UNREAD_CNT_RESPONSE

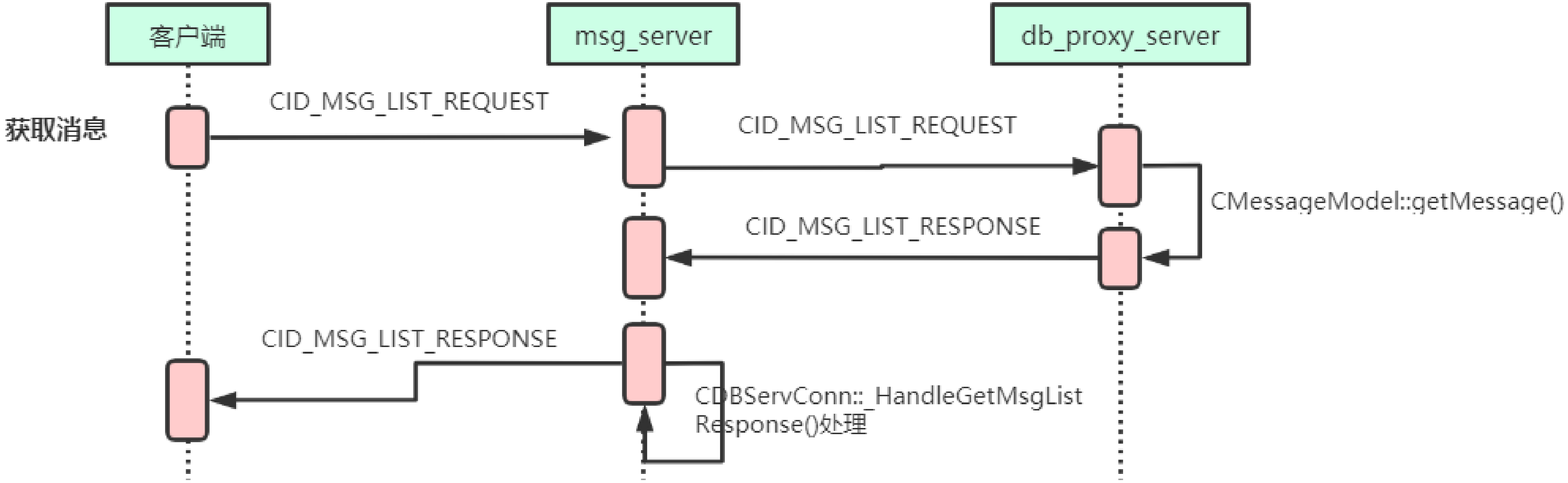
CID_MSG_UNREAD_CNT_RESPONSE

CDBServConn::_HandleUnreadMsgCountResponse()处理

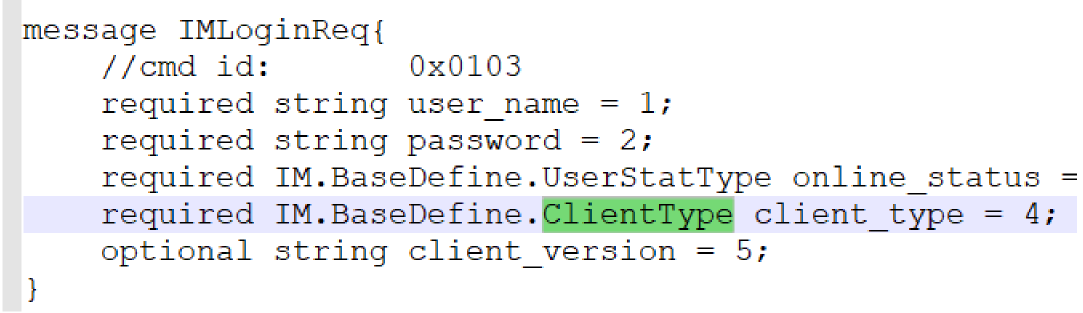


ProcessOn8群 11:47
ProcessOn-小麦:[动画表情] 75

1.0 msg_server登录请求响应过程3



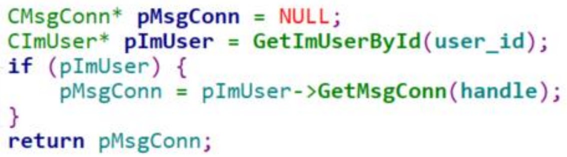
1. 1 msg_server登录流程1



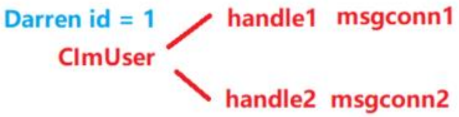
```
message IMLoginReq{
    //cmd id:        0x0103
    required string user_name = 1;
    required string password = 2;
    required IM.BaseDefine.UserStatType online_status =
    required IM.BaseDefine.ClientType client_type = 4;
    optional string client_version = 5;
}
```

用户登录

CMsgConn 对应1个客户端



```
CMsgConn* pMsgConn = NULL;
CImUser* pImUser = GetImUserById(user_id);
if (pImUser) {
    pMsgConn = pImUser->GetMsgConn(handle);
}
return pMsgConn;
```

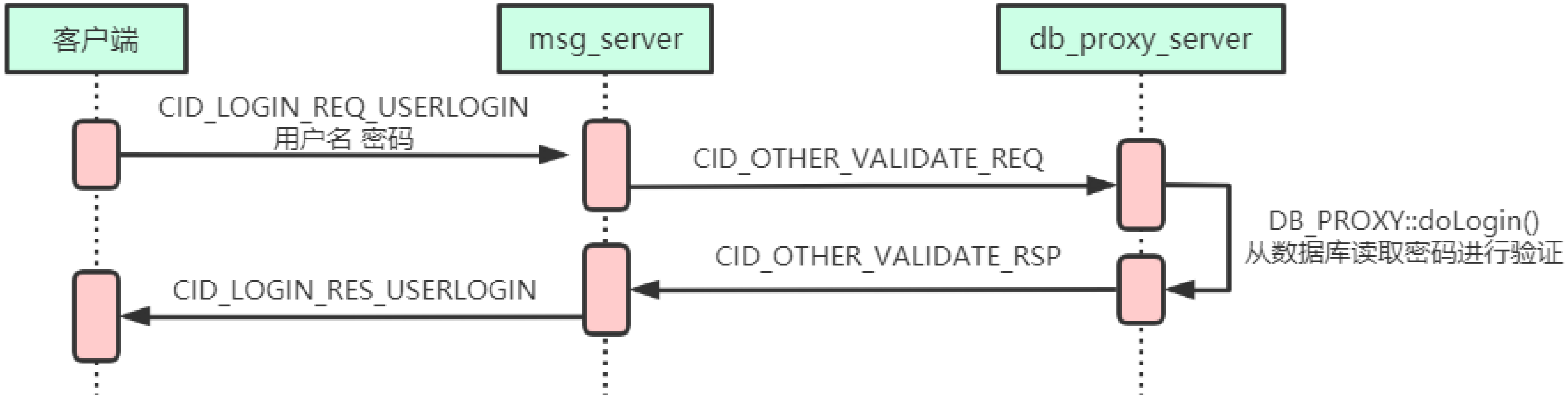


Darren id = 1

CImUser

handle1 msgconn1

handle2 msgconn2



客户端

msg_server

db_proxy_server

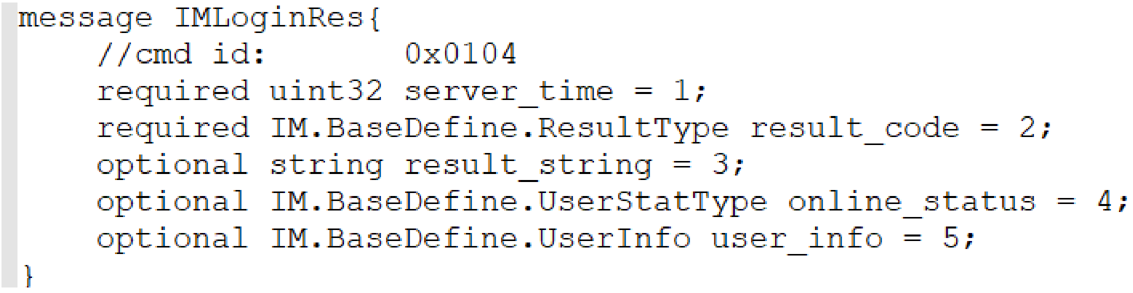
CID_LOGIN_REQ_USERLOGIN
用户名 密码

CID_OTHER_VALIDATE_REQ

DB_PROXY::doLogin()
从数据库读取密码进行验证

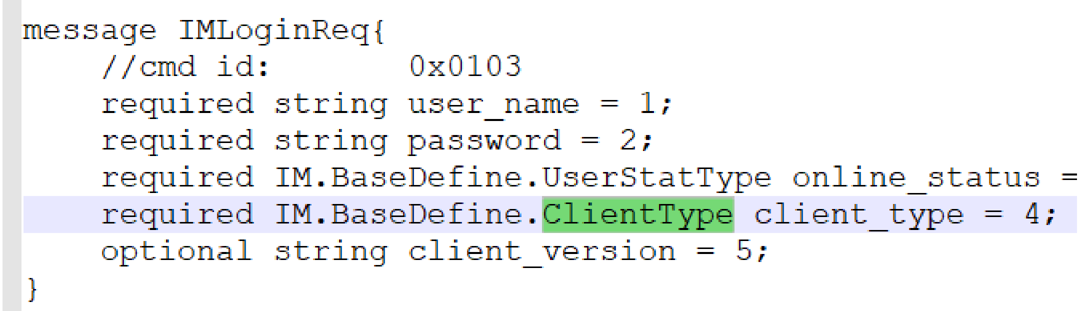
CID_OTHER_VALIDATE_RSP

CID_LOGIN_RES_USERLOGIN



```
message IMLoginRes{
    //cmd id:        0x0104
    required uint32 server_time = 1;
    required IM.BaseDefine.ResultType result_code = 2;
    optional string result_string = 3;
    optional IM.BaseDefine.UserStatType online_status = 4;
    optional IM.BaseDefine.UserInfo user_info = 5;
}
```

1.1 msg_server登录流程2

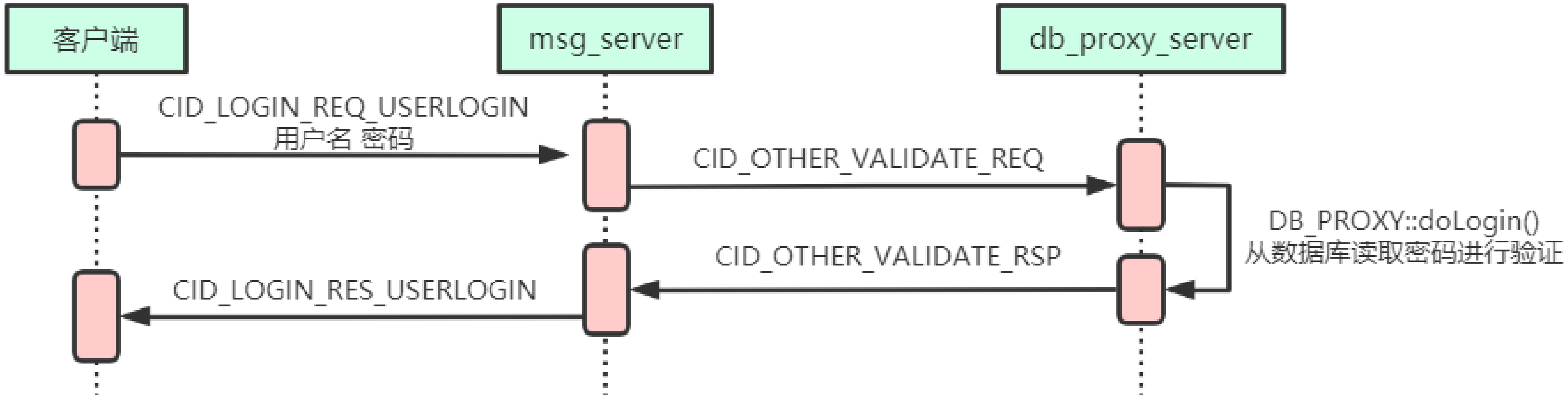


```
message IMLoginReq{
    //cmd id:       0x0103
    required string user_name = 1;
    required string password = 2;
    required IM.BaseDefine.UserStatType online_status =
    required IM.BaseDefine.ClientType client_type = 4;
    optional string client_version = 5;
}
```

响应函数

_HandleValidateResponse

1. pdu.SetCommandId(CID_OTHER_SERVER_KICK_USER); // 踢掉同一类型的登录
2. SendUserStatusUpdate(IM::BaseDefine::USER_STATUS_ONLINE); 更新状态信息
3. SetCommandId(CID_LOGIN_RES_USERLOGIN);回应登录结果



客户端

msg_server

db_proxy_server

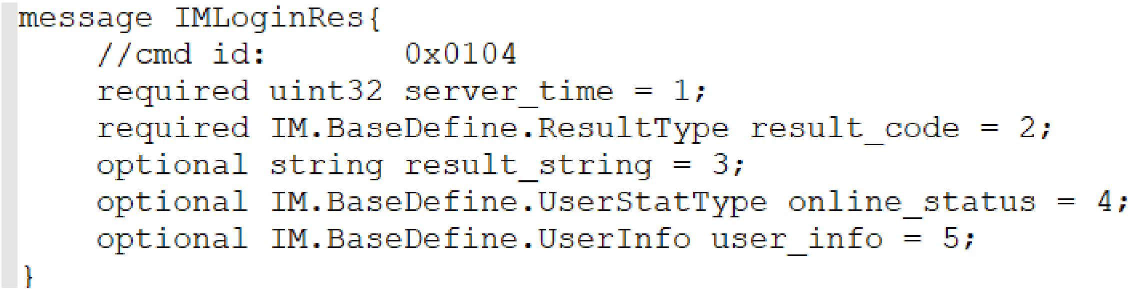
CID_LOGIN_REQ_USERLOGIN
用户名 密码

CID_OTHER_VALIDATE_REQ

DB_PROXY::doLogin()
从数据库读取密码进行验证

CID_OTHER_VALIDATE_RSP

CID_LOGIN_RES_USERLOGIN

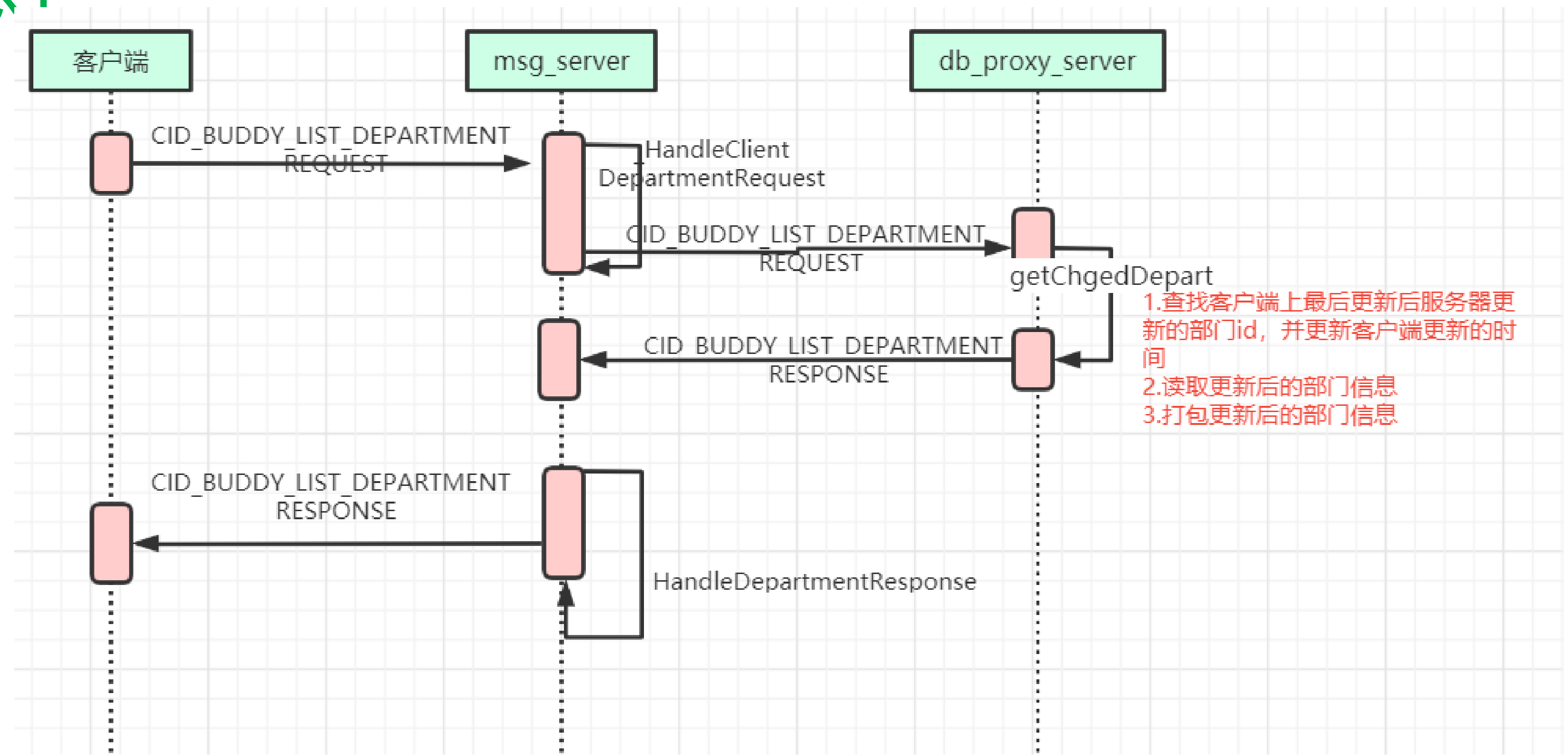


```
message IMLoginRes{
    //cmd id:       0x0104
    required uint32 server_time = 1;
    required IM.BaseDefine.ResultType result_code = 2;
    optional string result_string = 3;
    optional IM.BaseDefine.UserStatType online_status = 4;
    optional IM.BaseDefine.UserInfo user_info = 5;
}
```

1.2 更新部门信息1

请求：

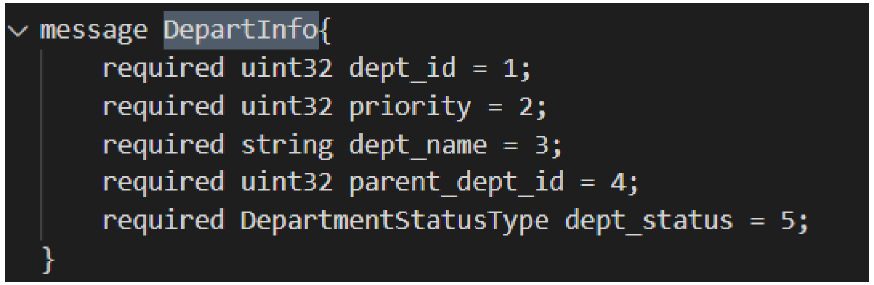
```
message IMDepartmentReq{
        //cmd id:          0x0210
        required uint32 user_id = 1;
        required uint32 latest_update_time = 2;
        optional bytes attach_data = 20;
}
```



1.查找客户端上最后更新后服务器更新的部门id，并更新客户端更新的时间
2.读取更新后的部门信息
3.打包更新后的部门信息

响应：

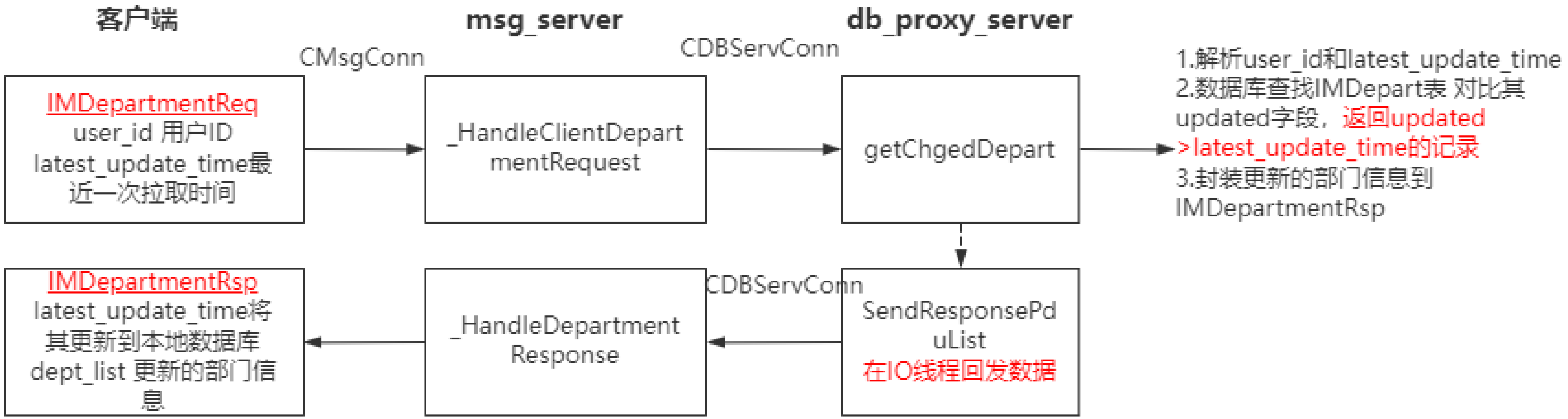
```
message IMDepartmentRsp{
        //cmd id:          0x0211
        required uint32 user_id = 1;
        required uint32 latest_update_time = 2;
        repeated IM.BaseDefine.DepartInfo dept_list = 3;
        optional bytes attach_data = 20;
}
```



```
message DepartInfo{
        required uint32 dept_id = 1;
        required uint32 priority = 2;
        required string dept_name = 3;
        required uint32 parent_dept_id = 4;
        required DepartmentStatusType dept_status = 5;
}
```

1. 2 更新部门信息2

拉取部门信息



客户端

msg_server

db_proxy_server

CMsgConn

CDBServConn

IMDepartmentReq
user_id 用户ID
latest_update_time最近一次拉取时间

_HandleClientDepartmentRequest

getChgedDepart

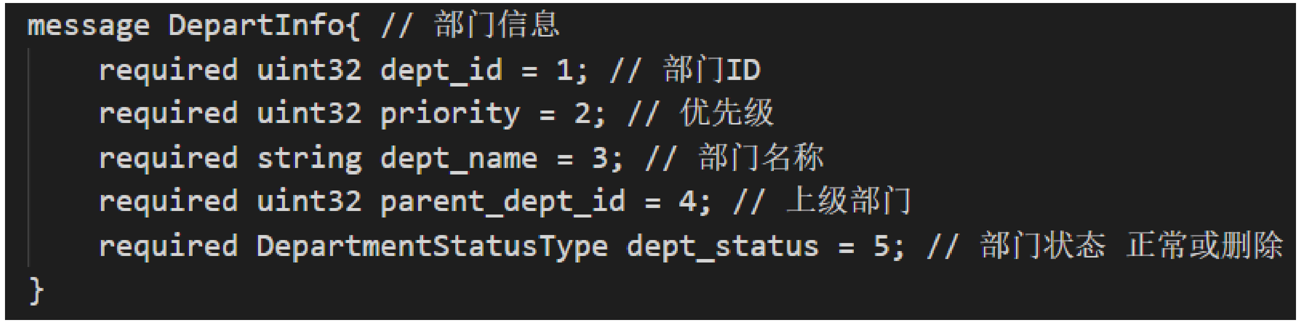
1.解析user_id和latest_update_time
2.数据库查找IMDepart表 对比其updated字段，返回updated>latest_update_time的记录
3.封装更新的部门信息到IMDepartmentRsp

IMDepartmentRsp
latest_update_time将其更新到本地数据库
dept_list 更新的部门信息

_HandleDepartmentResponse

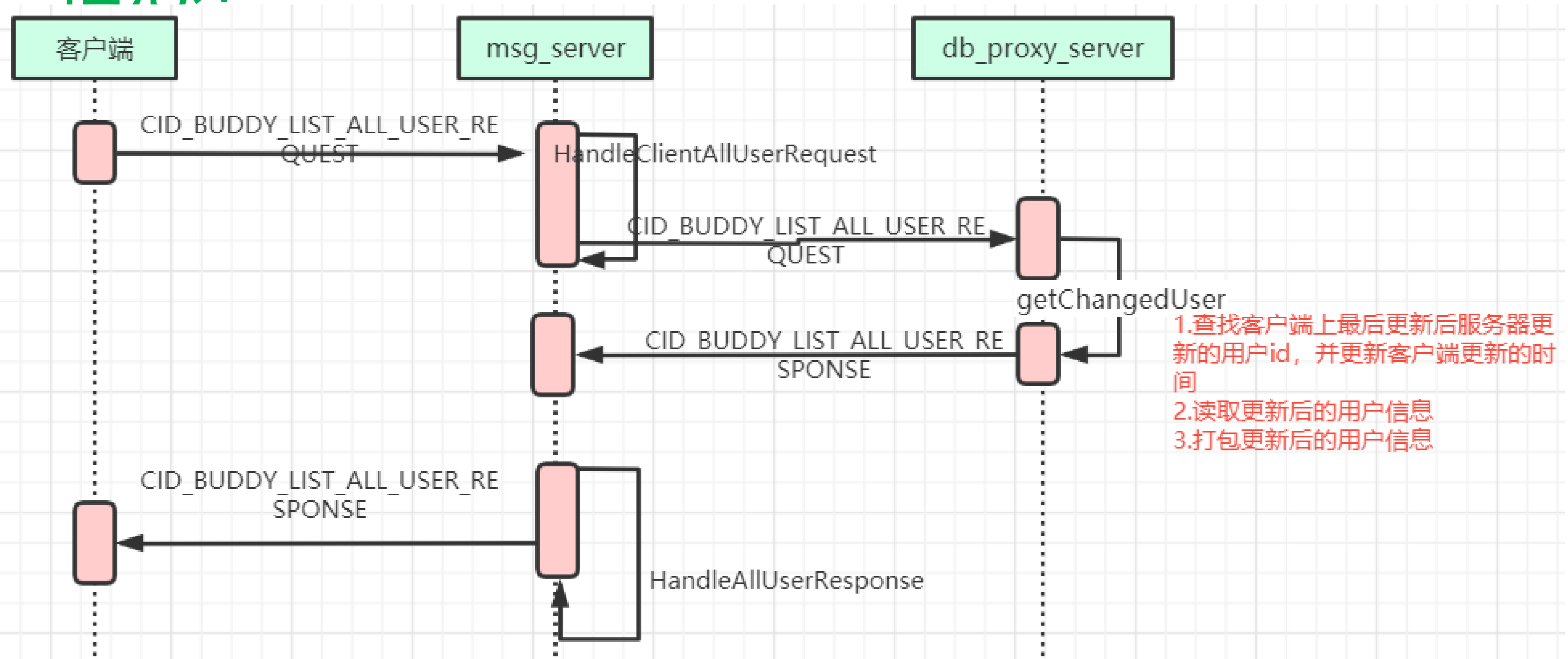
CDBServConn

SendResponsePduList
在IO线程回发数据



```
message DepartInfo{ // 部门信息
    required uint32 dept_id = 1; // 部门ID
    required uint32 priority = 2; // 优先级
    required string dept_name = 3; // 部门名称
    required uint32 parent_dept_id = 4; // 上级部门
    required DepartmentStatusType dept_status = 5; // 部门状态 正常或删除
}
```

1. 3请求所有用户信息1



客户端
msg_server
db_proxy_server
CID_BUDDY_LIST_ALL_USER_REQUEST
HandleClientAllUserRequest
CID_BUDDY_LIST_ALL_USER_REQUEST
getChangedUser
CID_BUDDY_LIST_ALL_USER_RESPONSE
CID_BUDDY_LIST_ALL_USER_RESPONSE
HandleAllUserResponse

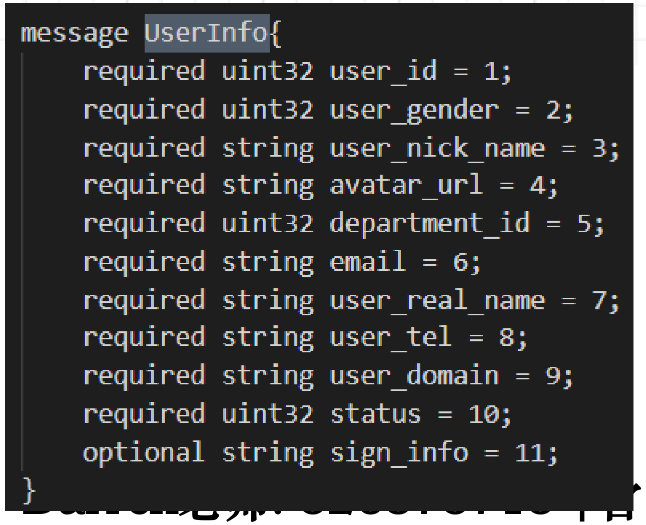
1.查找客户端上最后更新后服务器要更新的用户id，并更新客户端更新的时间
2.读取更新后的用户信息
3.打包更新后的用户信息

请求:

```
message IMAllUserReq{
        //cmd id:            0x0208
        required uint32 user_id = 1;
        required uint32 latest_update_time = 2
        optional bytes attach_data = 20;
}
```

响应:

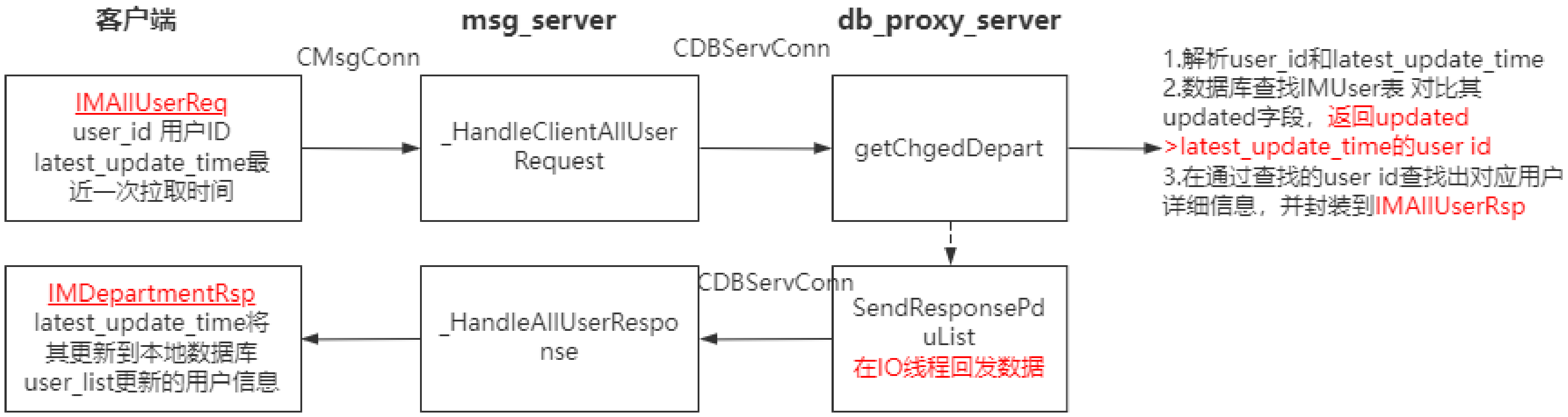
```
message IMAllUserRsp{
        //cmd id:            0x0209
        required uint32 user_id = 1;
        required uint32 latest_update_time = 2;
        repeated IM.BaseDefine.UserInfo user_list = 3;
        optional bytes attach_data = 20;
}
```



```
message UserInfo{
    required uint32 user_id = 1;
    required uint32 user_gender = 2;
    required string user_nick_name = 3;
    required string avatar_url = 4;
    required uint32 department_id = 5;
    required string email = 6;
    required string user_real_name = 7;
    required string user_tel = 8;
    required string user_domain = 9;
    required uint32 status = 10;
    optional string sign_info = 11;
}
```

1. 3请求所有用户信息2

更新用户列表的用户信息



客户端

msg_server

db_proxy_server

CMsgConn

CDBServConn

IMAllUserReq
user_id 用户ID
latest_update_time最近一次拉取时间

_HandleClientAllUserRequest

getChgedDepart

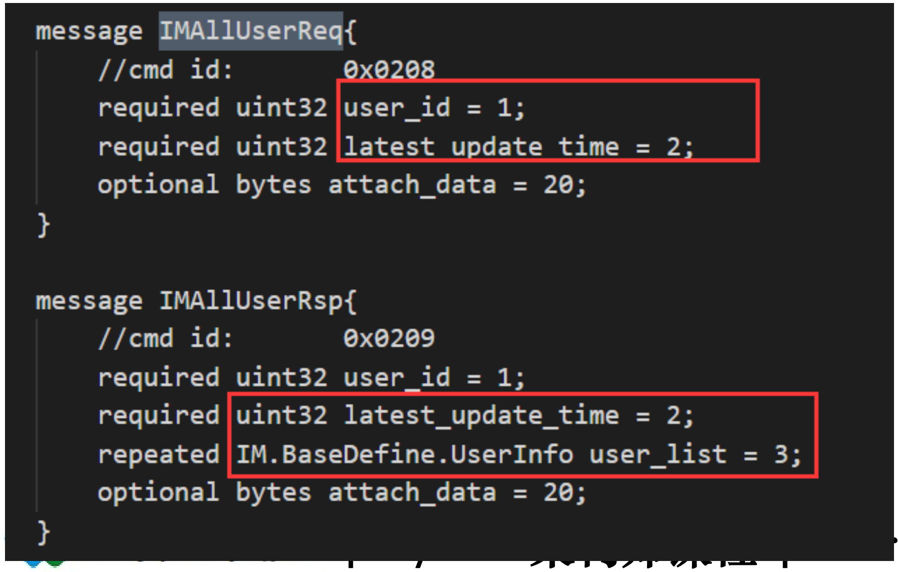
1.解析user_id和latest_update_time
2.数据库查找IMUser表 对比其updated字段，返回updated>latest_update_time的user id
3.在通过查找的user id查找出对应用户详细信息，并封装到IMAllUserRsp

SendResponsePduList
在IO线程回发数据

CDBServConn

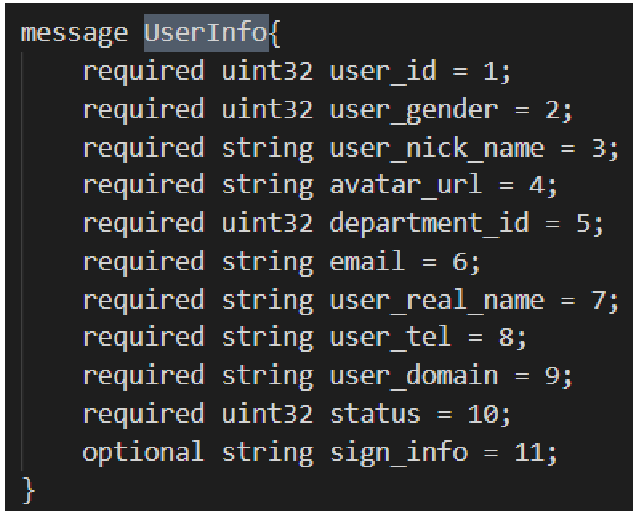
_HandleAllUserResponse

IMDepartmentRsp
latest_update_time将其更新到本地数据库
user_list更新的用户信息



```
message IMAllUserReq{
    //cmd id:        0x0208
    required uint32 user_id = 1;
    required uint32 latest_update_time = 2;
    optional bytes attach_data = 20;
}

message IMAllUserRsp{
    //cmd id:        0x0209
    required uint32 user_id = 1;
    required uint32 latest_update_time = 2;
    repeated IM.BaseDefine.UserInfo user_list = 3;
    optional bytes attach_data = 20;
}
```



```
message UserInfo{
    required uint32 user_id = 1;
    required uint32 user_gender = 2;
    required string user_nick_name = 3;
    required string avatar_url = 4;
    required uint32 department_id = 5;
    required string email = 6;
    required string user_real_name = 7;
    required string user_tel = 8;
    required string user_domain = 9;
    required uint32 status = 10;
    optional string sign_info = 11;
}
```

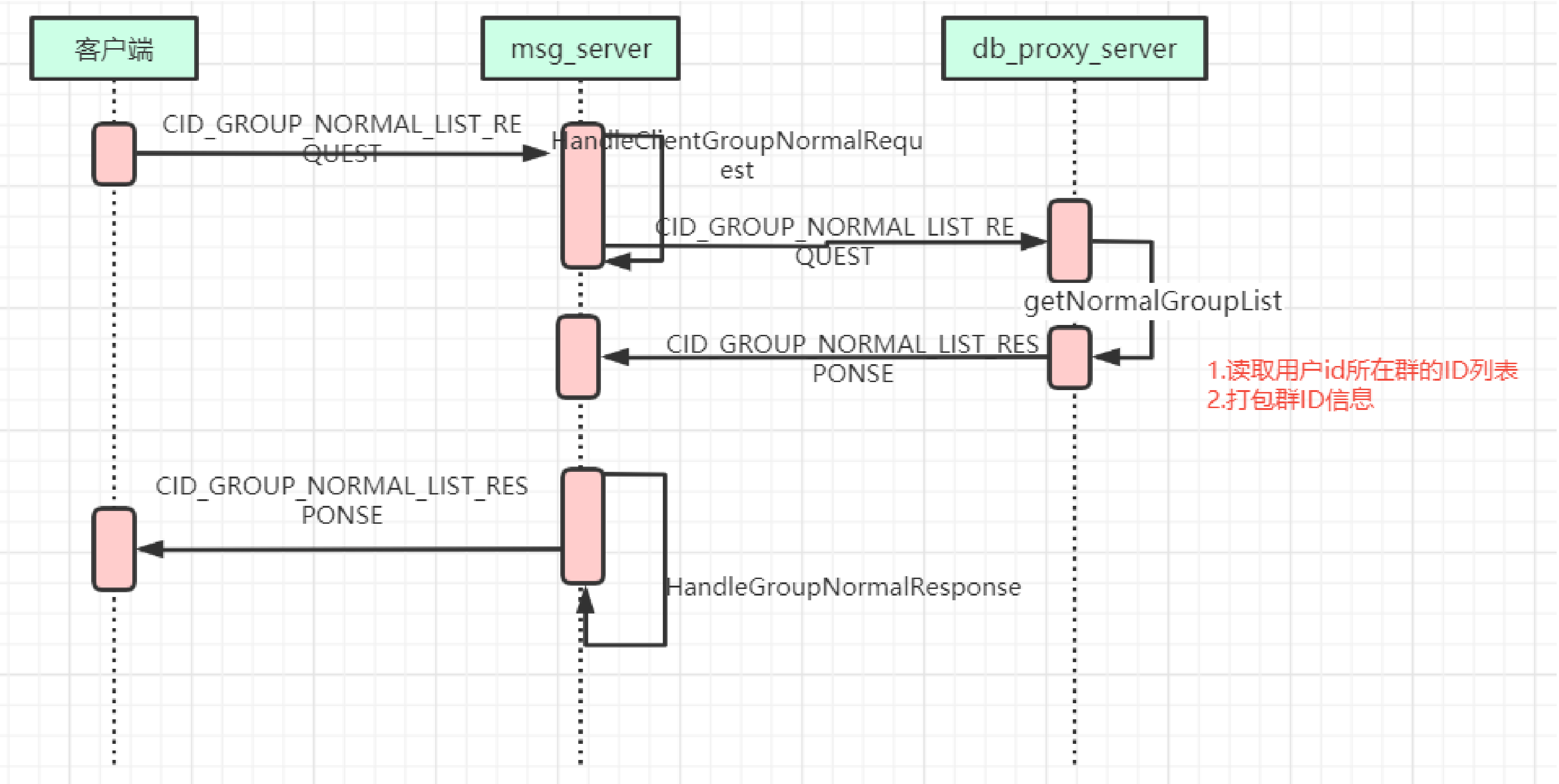
1. 4请求群组ID列表1

请求：

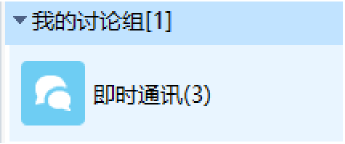
```
message IMNormalGroupListReq{
        //cmd id:                 0x0401
        required uint32 user_id = 1;
        optional bytes attach_data = 20;
}
```

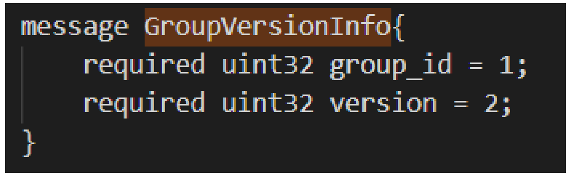
响应：

```
message IMNormalGroupListRsp{
        //cmd id:                 0x0402
        required uint32 user_id = 1;
        repeated IM.BaseDefine.GroupVersionInf
                group_version_list = 2;
        optional bytes attach_data = 20;
}
```


客户端
msg_server
db_proxy_server
CID_GROUP_NORMAL_LIST_REQUEST
HandleClientGroupNormalRequest
CID_GROUP_NORMAL_LIST_REQUEST
getNormalGroupList
CID_GROUP_NORMAL_LIST_RESPONSE
1.读取用户id所在群的ID列表
2.打包群ID信息
CID_GROUP_NORMAL_LIST_RESPONSE
HandleGroupNormalResponse

请求群组ID列表

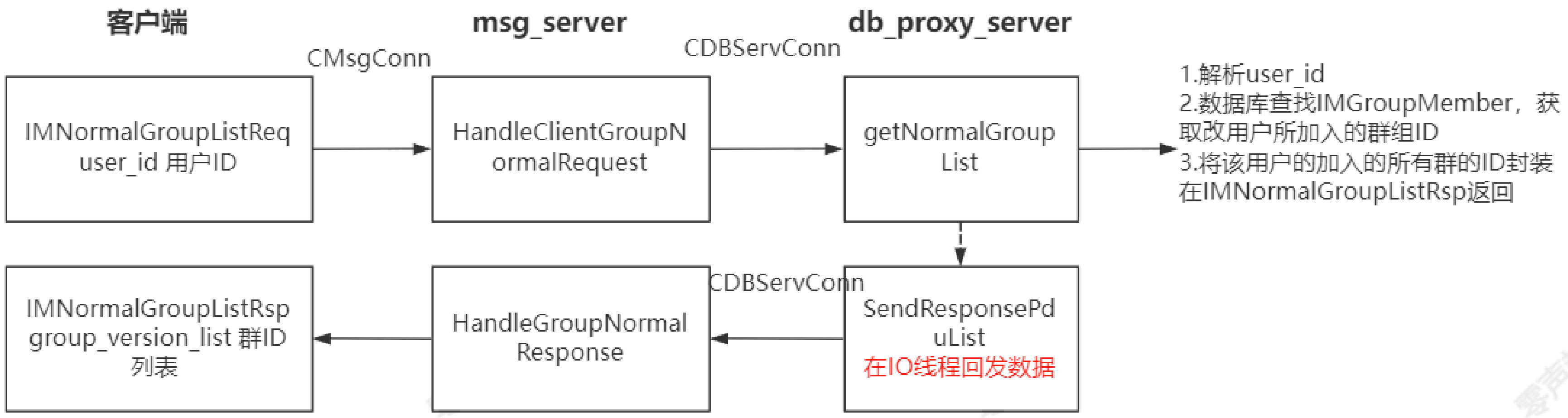




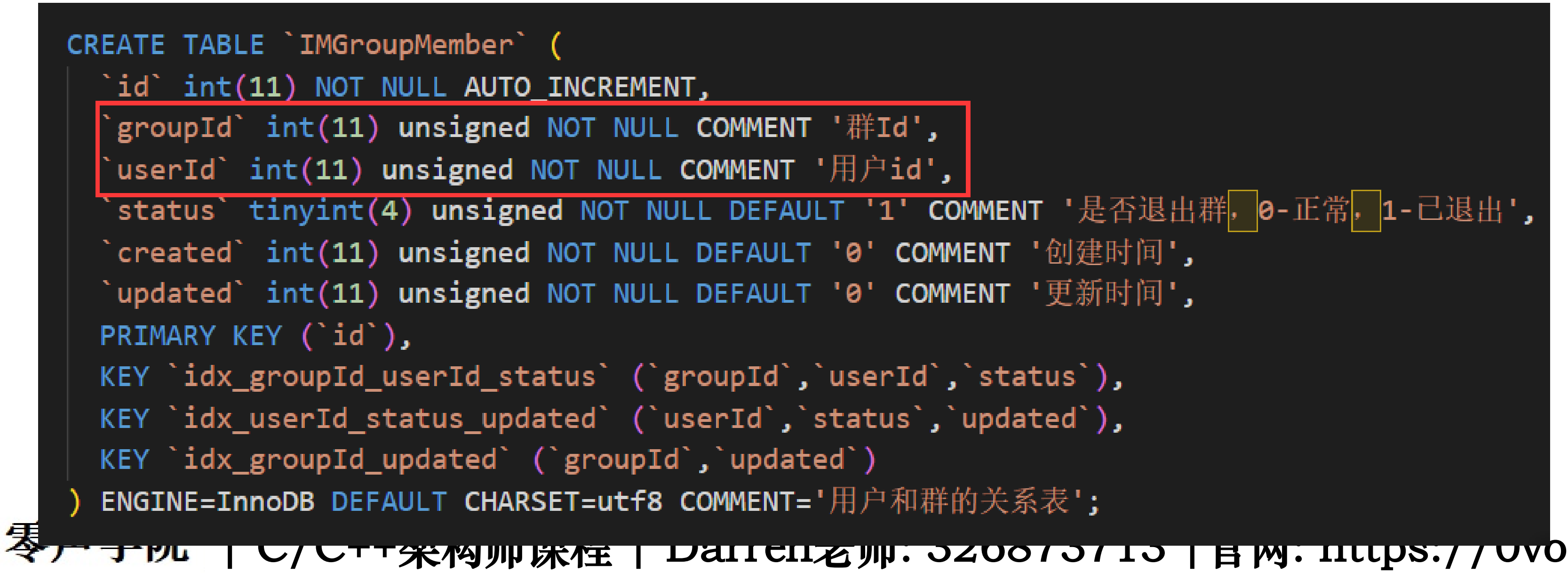
```
message GroupVersionInfo{
    required uint32 group_id = 1;
    required uint32 version = 2;
}
```

1. 4请求群组ID列表2

群组ID列表



客户端
msg_server
db_proxy_server
CMsgConn
CDBServConn
IMNormalGroupListReq
user_id 用户ID
HandleClientGroupNormalRequest
getNormalGroupList
1.解析user_id
2.数据库查找IMGroupMember，获取改用户所加入的群组ID
3.将该用户的加入的所有群的ID封装在IMNormalGroupListRsp返回
SendResponsePdu List
在IO线程回发数据
CDBServConn
HandleGroupNormalResponse
IMNormalGroupListRsp
group_version_list 群ID列表



```
CREATE TABLE `IMGroupMember` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `groupId` int(11) unsigned NOT NULL COMMENT '群Id',
  `userId` int(11) unsigned NOT NULL COMMENT '用户id',
  `status` tinyint(4) unsigned NOT NULL DEFAULT '1' COMMENT '是否退出群，0-正常，1-已退出',
  `created` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间',
  `updated` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间',
  PRIMARY KEY (`id`),
  KEY `idx_groupId_userId_status` (`groupId`,`userId`,`status`),
  KEY `idx_userId_status_updated` (`userId`,`status`,`updated`),
  KEY `idx_groupId_updated` (`groupId`,`updated`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='用户和群的关系表';
```

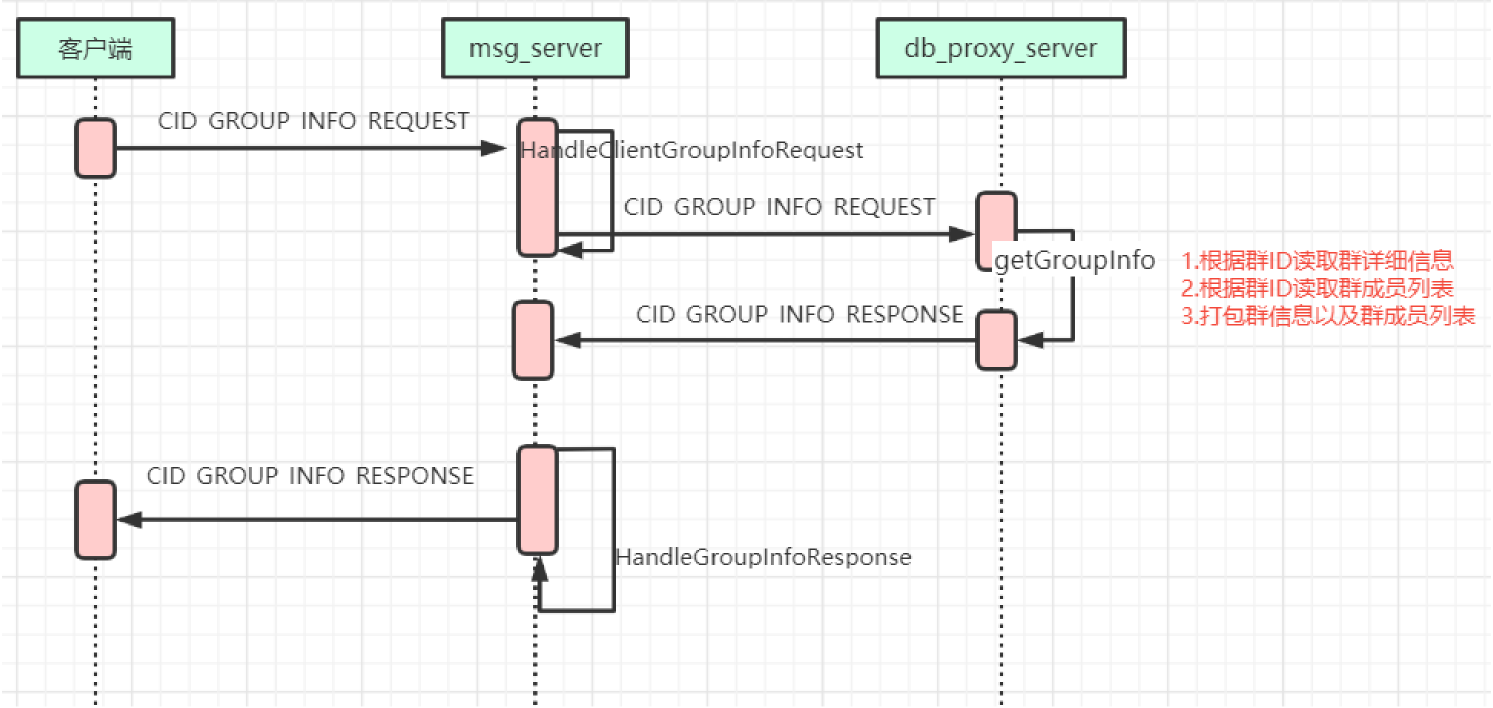
1. 5请求群组具体信息1

请求：

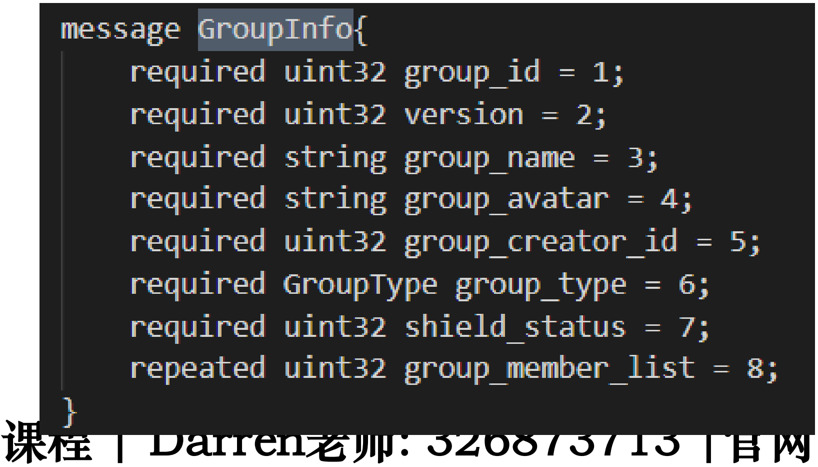
```
message IMGroupInfoListReq{
        //cmd id:                    0x0403
        required uint32 user_id = 1;
        repeated IM.BaseDefine.GroupVersionInfo
            group_version_list = 2;
        optional bytes attach_data = 20;
}
```

响应：

```
message IMGroupInfoListRsp{
        //cmd id:                    0x0404
        required uint32 user_id = 1;
        repeated IM.BaseDefine.GroupInfo
           group_info_list = 2;
        optional bytes attach_data = 20;
}
```



1.根据群ID读取群详细信息
2.根据群ID读取群成员列表
3.打包群信息以及群成员列表

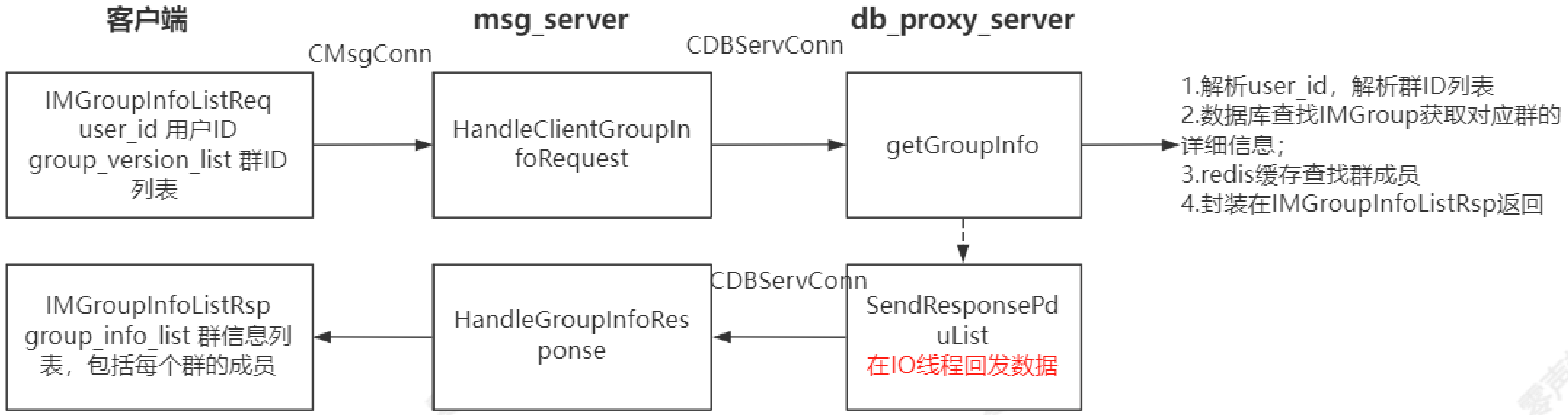


```
message GroupInfo{
    required uint32 group_id = 1;
    required uint32 version = 2;
    required string group_name = 3;
    required string group_avatar = 4;
    required uint32 group_creator_id = 5;
    required GroupType group_type = 6;
    required uint32 shield_status = 7;
    repeated uint32 group_member_list = 8;
}
```

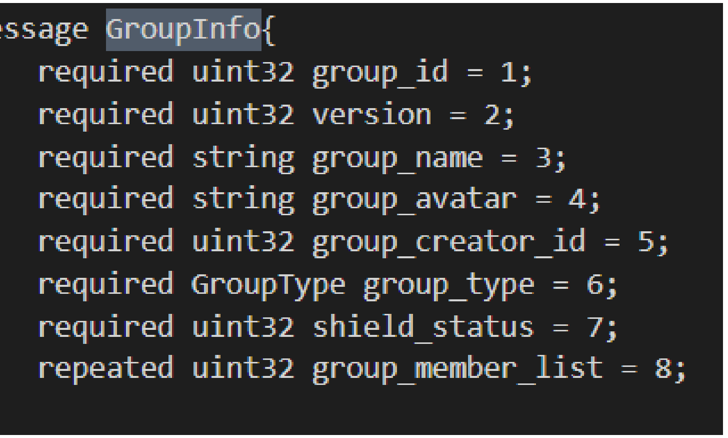
群成员列表

1. 5请求群组具体信息2

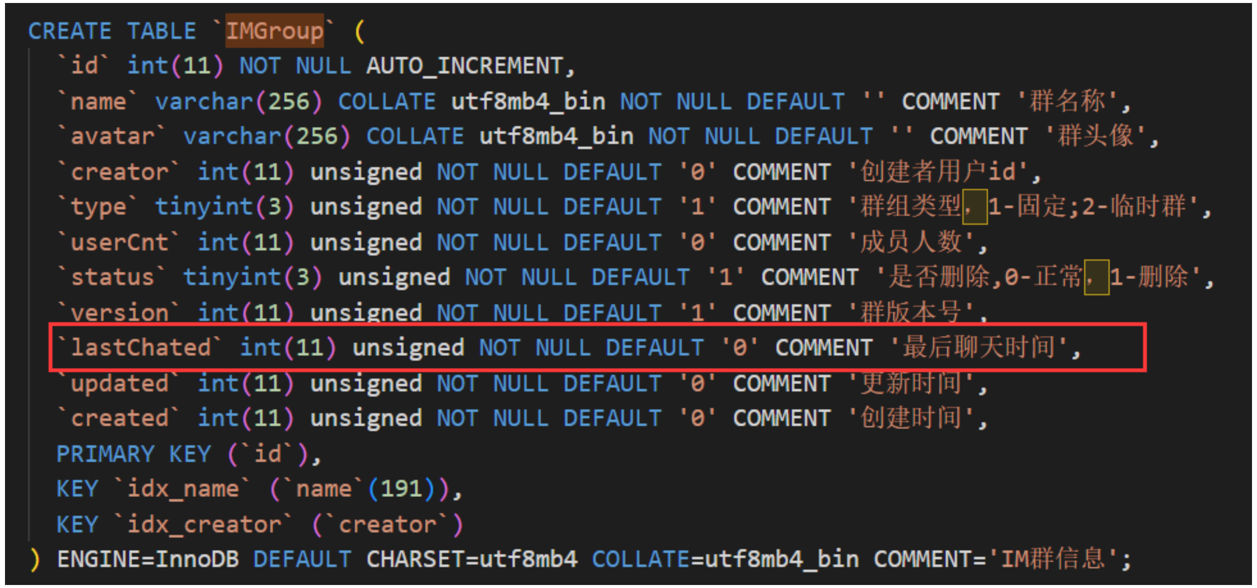
请求群组具体信息



1.解析user_id，解析群ID列表
2.数据库查找IMGroup获取对应群的详细信息；
3.redis缓存查找群成员
4.封装在IMGroupInfoListRsp返回



```
ssage GroupInfo{
  required uint32 group_id = 1;
  required uint32 version = 2;
  required string group_name = 3;
  required string group_avatar = 4;
  required uint32 group_creator_id = 5;
  required GroupType group_type = 6;
  required uint32 shield_status = 7;
  repeated uint32 group_member_list = 8;
```



```
CREATE TABLE `IMGroup` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(256) COLLATE utf8mb4_bin NOT NULL DEFAULT '' COMMENT '群名称',
  `avatar` varchar(256) COLLATE utf8mb4_bin NOT NULL DEFAULT '' COMMENT '群头像',
  `creator` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '创建者用户id',
  `type` tinyint(3) unsigned NOT NULL DEFAULT '1' COMMENT '群组类型，1-固定;2-临时群',
  `userCnt` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '成员人数',
  `status` tinyint(3) unsigned NOT NULL DEFAULT '1' COMMENT '是否删除,0-正常，1-删除',
  `version` int(11) unsigned NOT NULL DEFAULT '1' COMMENT '群版本号',
  `lastChated` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '最后聊天时间',
  `updated` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间',
  `created` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间',
  PRIMARY KEY (`id`),
  KEY `idx_name` (`name`(191)),
  KEY `idx_creator` (`creator`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin COMMENT='IM群信息';
```

1. 6获取最近联系会话1

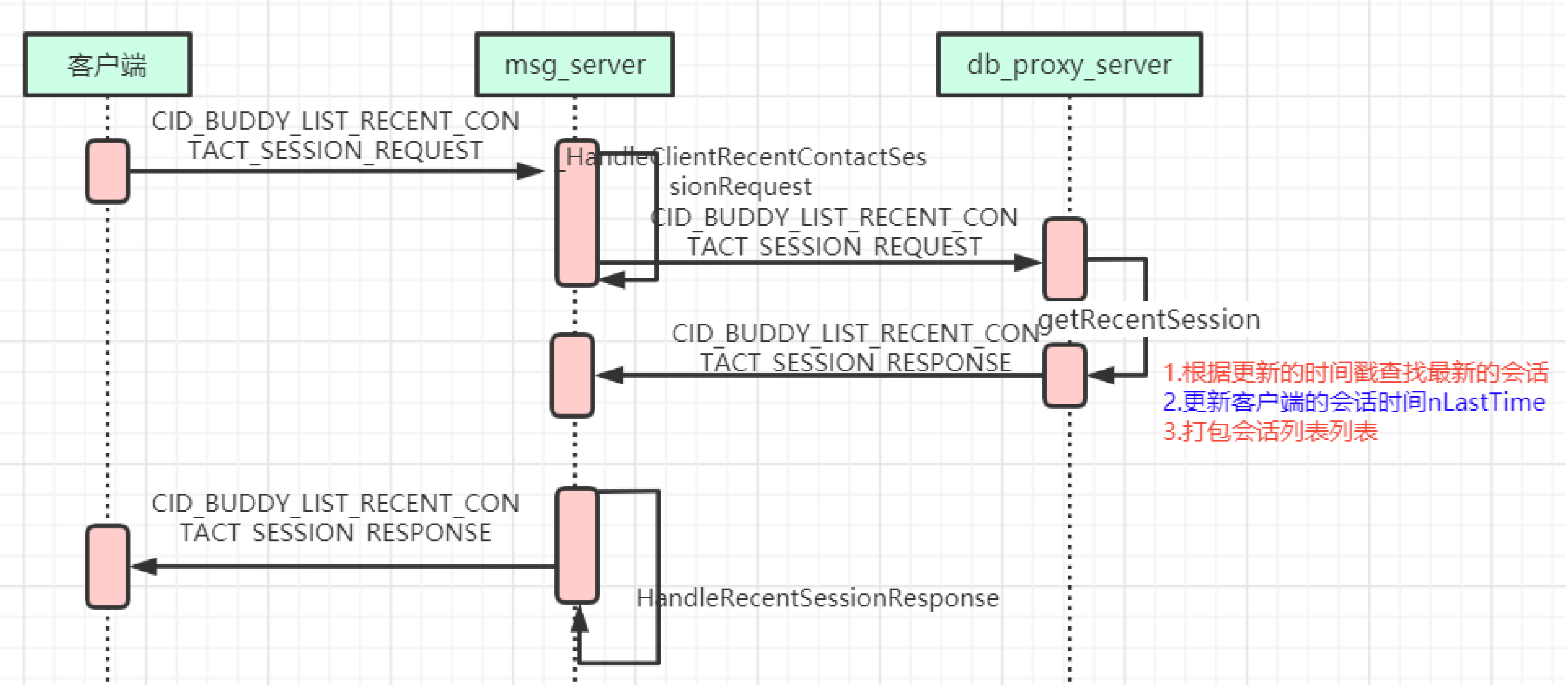
IMRecentSession

请求：

```
message IMRecentContactSessionReq{
        //cmd id:             0x0201
        required uint32 user_id = 1;
        required uint32 latest_update_time = 2;
        optional bytes attach_data = 20;
}
```

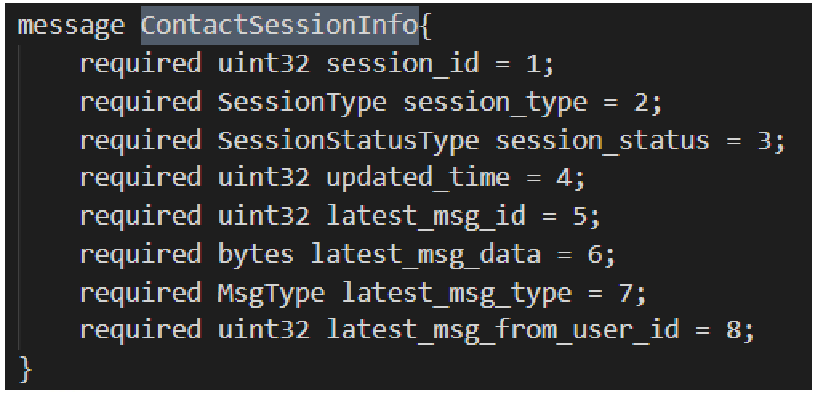
响应：

```
message IMRecentContactSessionRsp{
        //cmd id:             0x0202
        required uint32 user_id = 1;
        repeated IM.BaseDefine.ContactSessionInfo
          contact_session_list = 2;
        optional bytes attach_data = 20;
}
```

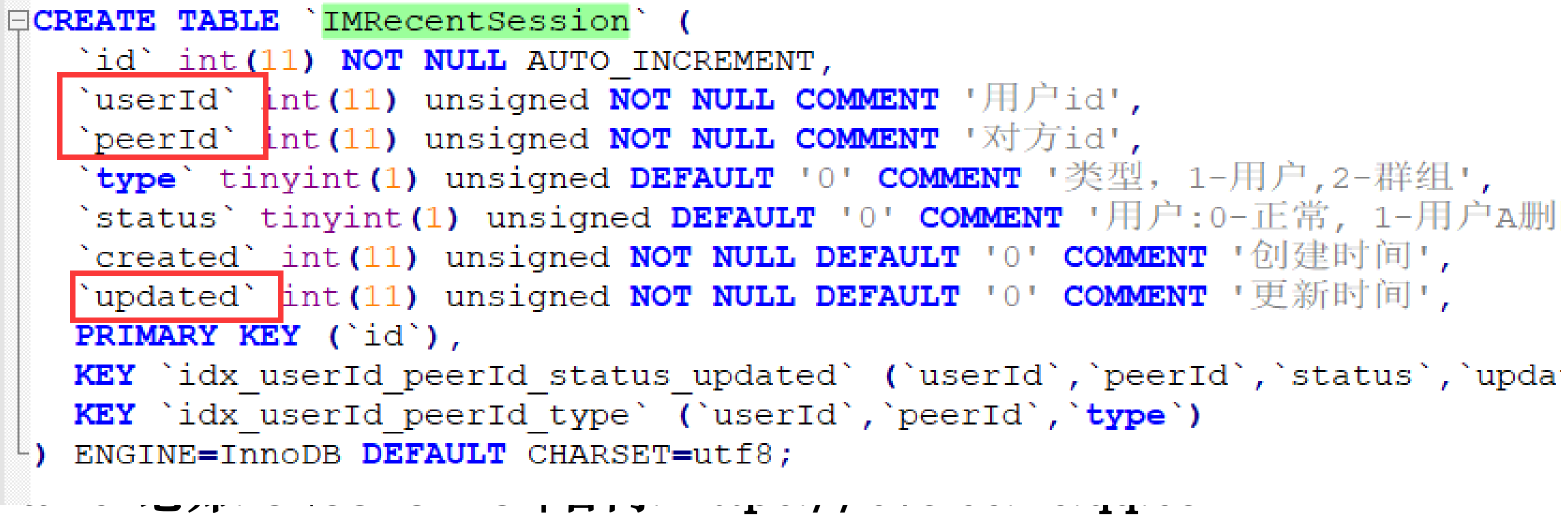


客户端
msg_server
db_proxy_server
CID_BUDDY_LIST_RECENT_CONTACT_SESSION_REQUEST
HandleClientRecentContactSessionRequest
CID_BUDDY_LIST_RECENT_CONTACT_SESSION_REQUEST
getRecentSession
CID_BUDDY_LIST_RECENT_CONTACT_SESSION_RESPONSE
CID_BUDDY_LIST_RECENT_CONTACT_SESSION_RESPONSE
HandleRecentSessionResponse

1.根据更新的时间戳查找最新的会话
2.更新客户端的会话时间nLastTime
3.打包会话列表列表



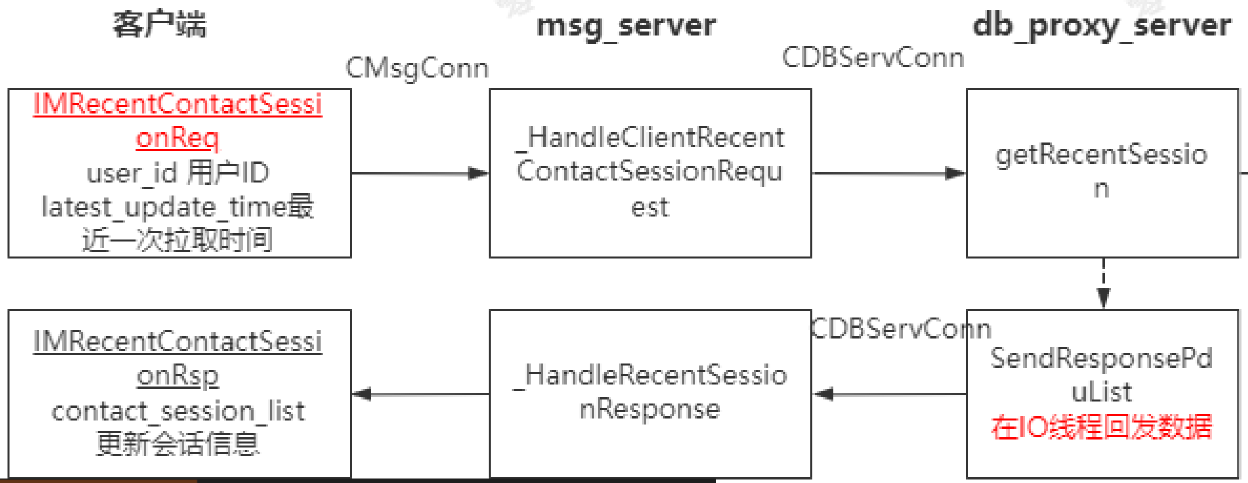
```
message ContactSessionInfo{
    required uint32 session_id = 1;
    required SessionType session_type = 2;
    required SessionStatusType session_status = 3;
    required uint32 updated_time = 4;
    required uint32 latest_msg_id = 5;
    required bytes latest_msg_data = 6;
    required MsgType latest_msg_type = 7;
    required uint32 latest_msg_from_user_id = 8;
}
```



```
CREATE TABLE `IMRecentSession` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `userId` int(11) unsigned NOT NULL COMMENT '用户id',
  `peerId` int(11) unsigned NOT NULL COMMENT '对方id',
  `type` tinyint(1) unsigned DEFAULT '0' COMMENT '类型，1-用户,2-群组',
  `status` tinyint(1) unsigned DEFAULT '0' COMMENT '用户:0-正常, 1-用户A删
  `created` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间',
  `updated` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间',
  PRIMARY KEY (`id`),
  KEY `idx_userId_peerId_status_updated` (`userId`,`peerId`,`status`,`upda
  KEY `idx_userId_peerId_type` (`userId`,`peerId`,`type`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

1. 6获取最近联系会话2

获取最近联系会话



客户端

msg_server

db_proxy_server

CMsgConn

CDBServConn

IMRecentContactSessionReq
user_id 用户ID
latest_update_time最近一次拉取时间

_HandleClientRecentContactSessionRequest

getRecentSession

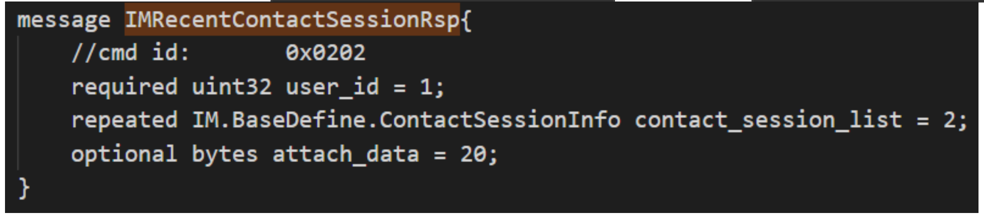
SendResponsePduList
在IO线程回发数据

CDBServConn

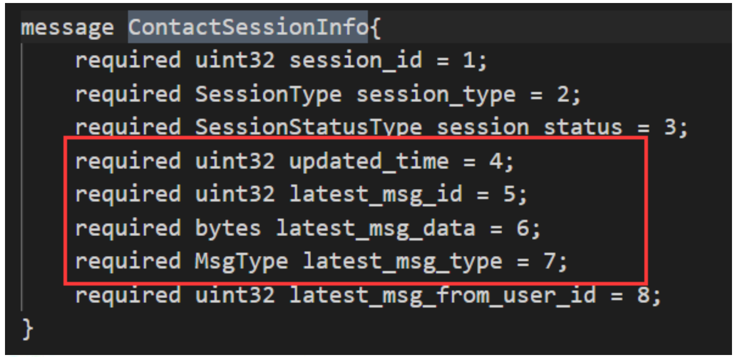
_HandleRecentSessionResponse

IMRecentContactSessionRsp
contact_session_list
更新会话信息

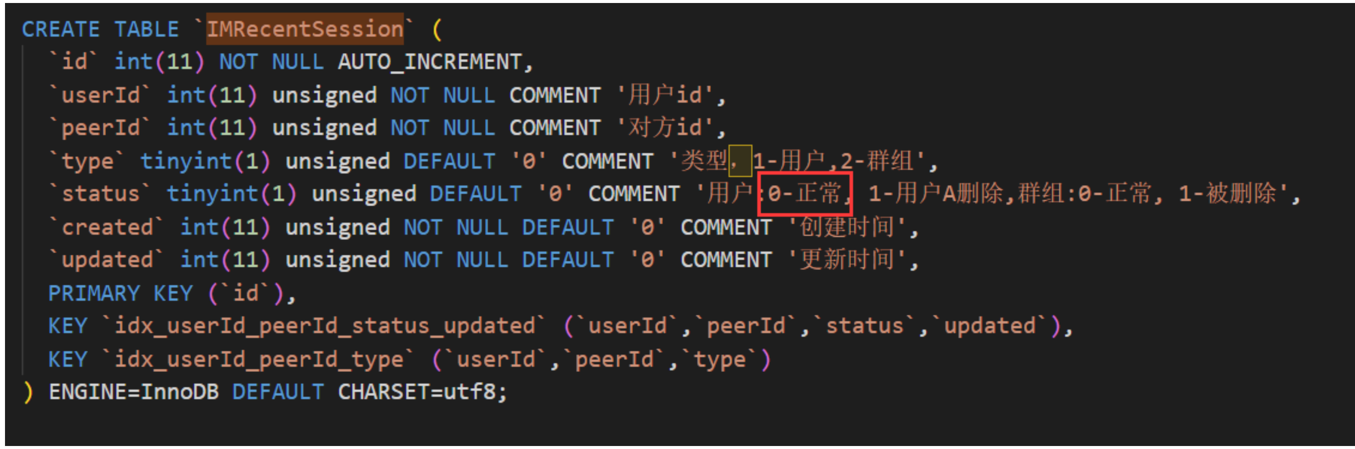
1.解析user_id和latest_update_time
2.数据库查找IMRecentSession表 对比其updated字段，返回updated >latest_update_time的user id
3.在通过查找的user id查找出对应会话信息，并封装到IMRecentContactSessionRsp
需要注意的是，返回的会话还包括了最后的聊天信息



```
message IMRecentContactSessionRsp{
    //cmd id:        0x0202
    required uint32 user_id = 1;
    repeated IM.BaseDefine.ContactSessionInfo contact_session_list = 2;
    optional bytes attach_data = 20;
}
```



```
message ContactSessionInfo{
    required uint32 session_id = 1;
    required SessionType session_type = 2;
    required SessionStatusType session_status = 3;
    required uint32 updated_time = 4;
    required uint32 latest_msg_id = 5;
    required bytes latest_msg_data = 6;
    required MsgType latest_msg_type = 7;
    required uint32 latest_msg_from_user_id = 8;
}
```



```
CREATE TABLE `IMRecentSession` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `userId` int(11) unsigned NOT NULL COMMENT '用户id',
  `peerId` int(11) unsigned NOT NULL COMMENT '对方id',
  `type` tinyint(1) unsigned DEFAULT '0' COMMENT '类型，1-用户,2-群组',
  `status` tinyint(1) unsigned DEFAULT '0' COMMENT '用户:0-正常, 1-用户A删除,群组:0-正常, 1-被删除',
  `created` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间',
  `updated` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间',
  PRIMARY KEY (`id`),
  KEY `idx_userId_peerId_status_updated` (`userId`,`peerId`,`status`,`updated`),
  KEY `idx_userId_peerId_type` (`userId`,`peerId`,`type`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

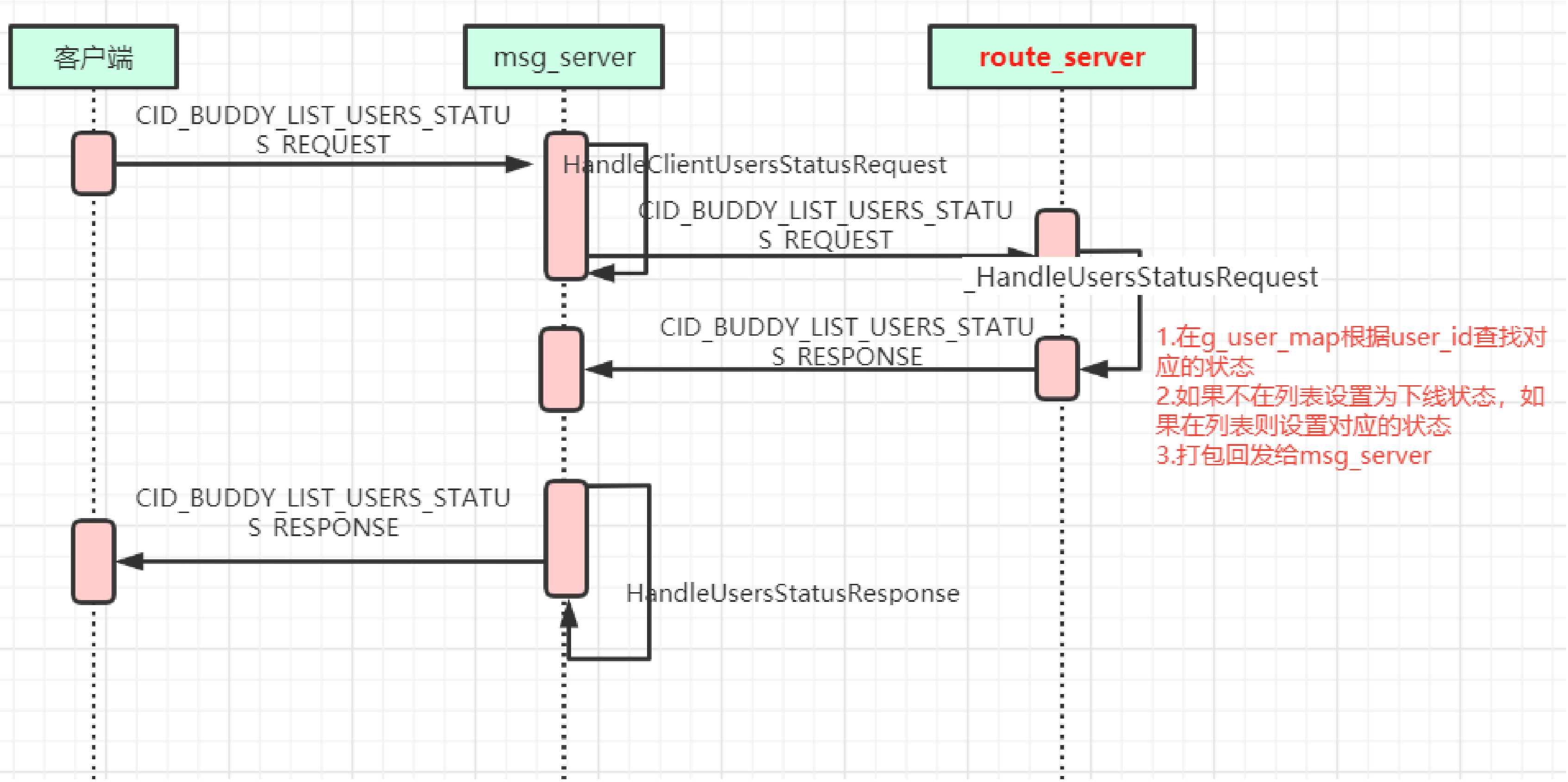
1. 7查询用户当前在线状态1

请求：

```
message IMUsersStatReq{
        //cmd id:            0x020a
        required uint32 user_id = 1;
        repeated uint32 user_id_list = 2;
        optional bytes attach_data = 20;
}
```

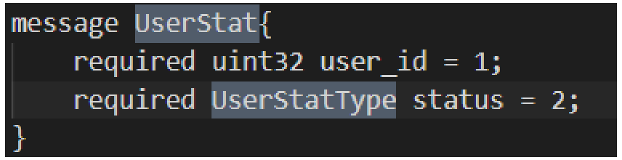
响应：

```
message IMUsersStatRsp{
        //cmd id:            0x020b
        required uint32 user_id = 1;
        repeated IM.BaseDefine.UserStat
            user_stat_list = 2;
        optional bytes attach_data = 20;
}
```

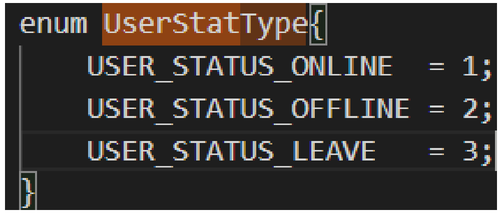


客户端
msg_server
route_server
CID_BUDDY_LIST_USERS_STATUS_REQUEST
HandleClientUsersStatusRequest
CID_BUDDY_LIST_USERS_STATUS_REQUEST
_HandleUsersStatusRequest
CID_BUDDY_LIST_USERS_STATUS_RESPONSE
CID_BUDDY_LIST_USERS_STATUS_RESPONSE
HandleUsersStatusResponse

1.在g_user_map根据user_id查找对应的状态
2.如果不在列表设置为下线状态，如果在列表则设置对应的状态
3.打包回发给msg_server

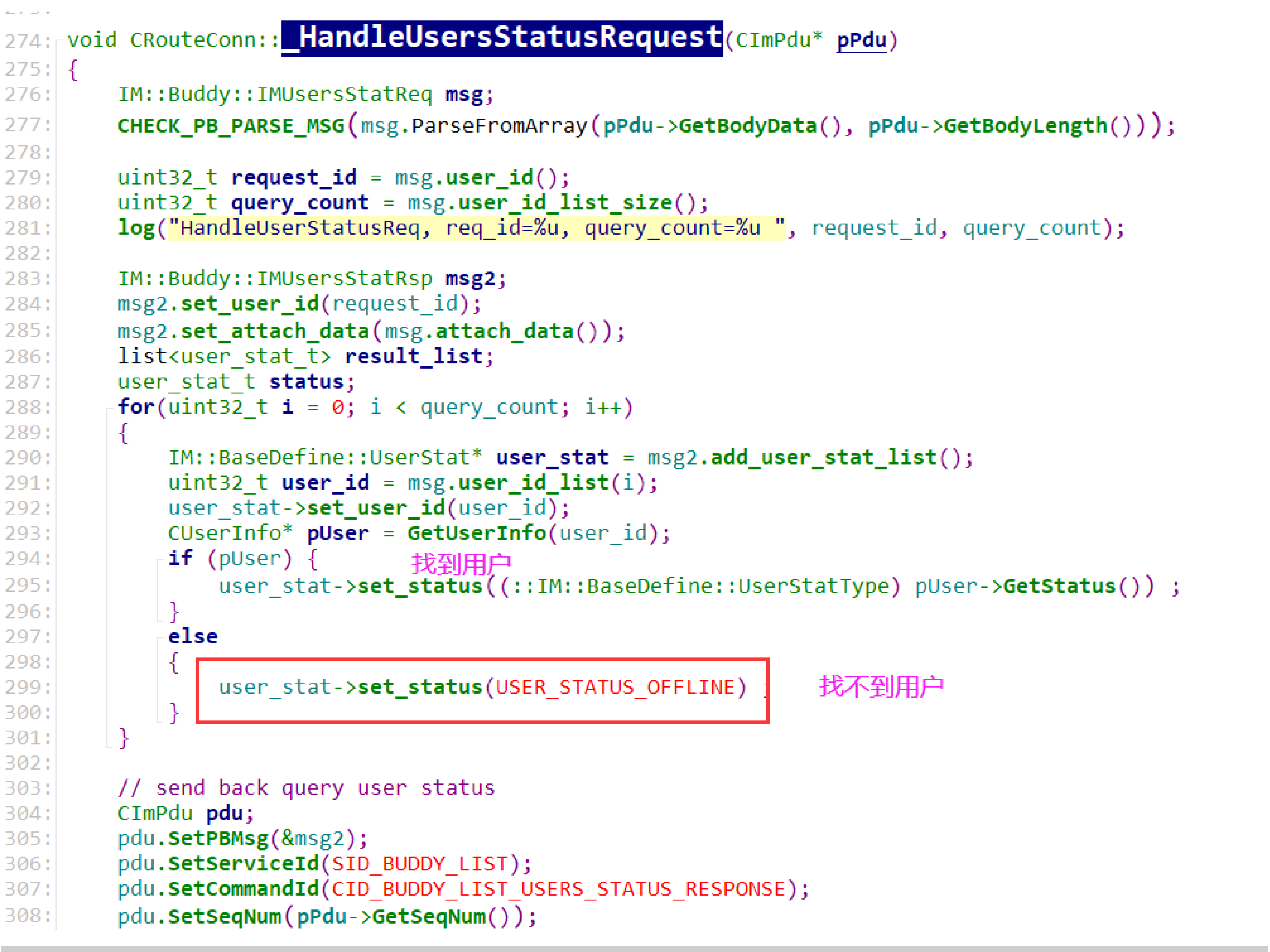


```
message UserStat{
    required uint32 user_id = 1;
    required UserStatType status = 2;
}
```



```
enum UserStatType{
    USER_STATUS_ONLINE   = 1;
    USER_STATUS_OFFLINE  = 2;
    USER_STATUS_LEAVE    = 3;
}
```

1. 7查询用户当前在线状态2



```
void CRouteConn::_HandleUsersStatusRequest(CImPdu* pPdu)
{
    IM::Buddy::IMUsersStatReq msg;
    CHECK_PB_PARSE_MSG(msg.ParseFromArray(pPdu->GetBodyData(), pPdu->GetBodyLength()));

    uint32_t request_id = msg.user_id();
    uint32_t query_count = msg.user_id_list_size();
    log("HandleUserStatusReq, req_id=%u, query_count=%u ", request_id, query_count);

    IM::Buddy::IMUsersStatRsp msg2;
    msg2.set_user_id(request_id);
    msg2.set_attach_data(msg.attach_data());
    list<user_stat_t> result_list;
    user_stat_t status;
    for(uint32_t i = 0; i < query_count; i++)
    {
        IM::BaseDefine::UserStat* user_stat = msg2.add_user_stat_list();
        uint32_t user_id = msg.user_id_list(i);
        user_stat->set_user_id(user_id);
        CUserInfo* pUser = GetUserInfo(user_id);
        if (pUser) {
            user_stat->set_status((::IM::BaseDefine::UserStatType) pUser->GetStatus()) ;
        }
        else
        {
            user_stat->set_status(USER_STATUS_OFFLINE)
        }
    }

    // send back query user status
    CImPdu pdu;
    pdu.SetPBMsg(&msg2);
    pdu.SetServiceId(SID_BUDDY_LIST);
    pdu.SetCommandId(CID_BUDDY_LIST_USERS_STATUS_RESPONSE);
    pdu.SetSeqNum(pPdu->GetSeqNum());
```

找到用户

找不到用户

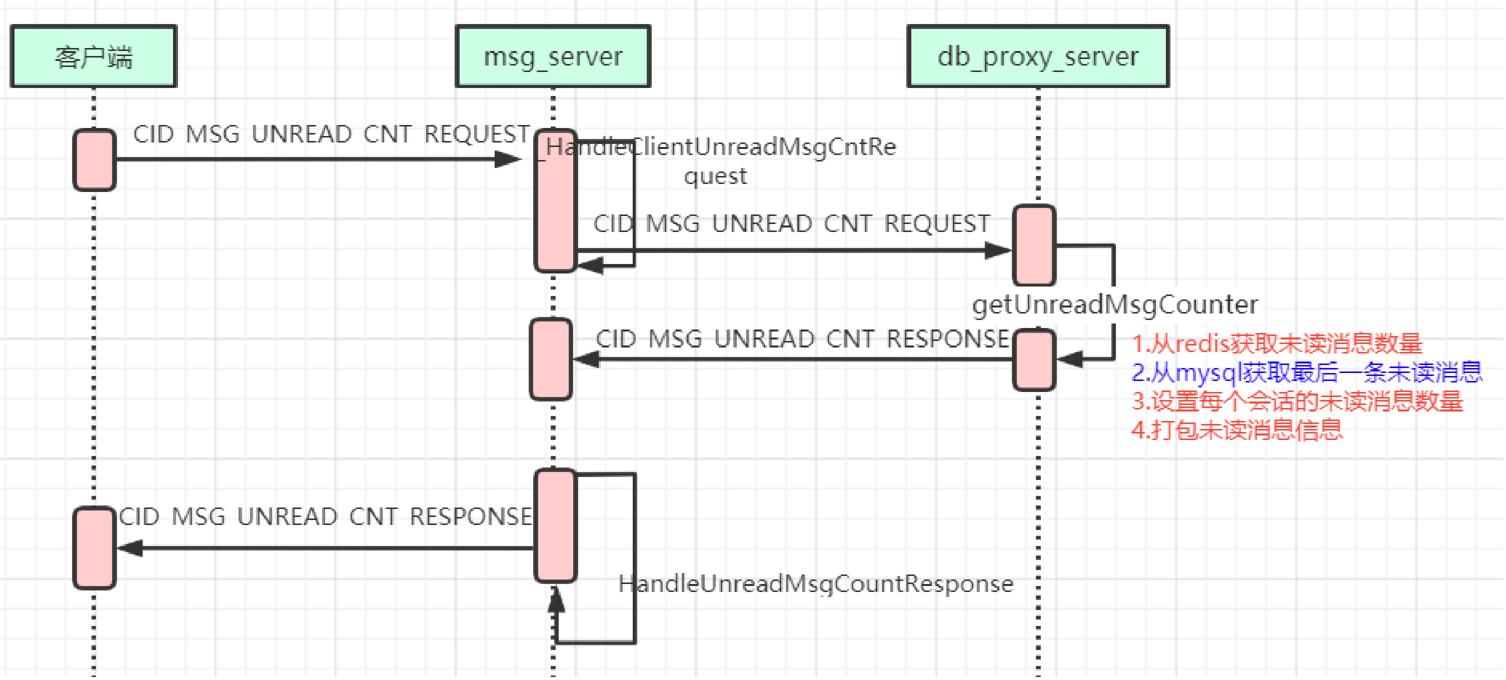
1. 8获取未读消息数量1

请求：

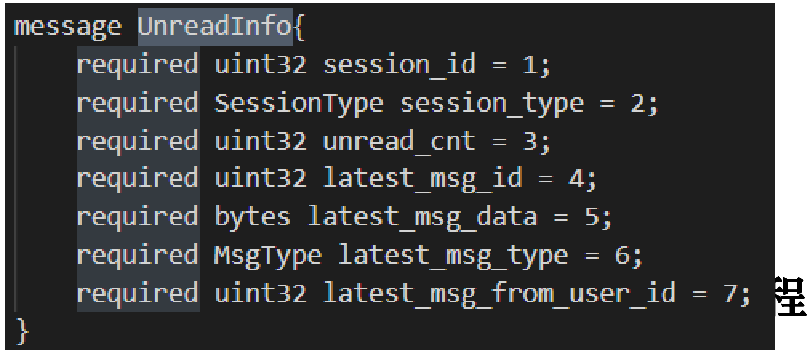
```
message IMUnreadMsgCntReq{
        //cmd id:             0x0307
        required uint32 user_id = 1;
        optional bytes attach_data = 20;
}
```

响应：

```
message IMUnreadMsgCntRsp{
        //cmd id:             0x0308
        required uint32 user_id = 1;
        required uint32 total_cnt = 2;
        repeated IM.BaseDefine.UnreadInfo
            unreadinfo_list = 3;
        optional bytes attach_data = 20;
}
```


客户端
msg_server
db_proxy_server
CID_MSG_UNREAD_CNT_REQUEST
HandleClientUnreadMsgCntRequest
CID_MSG_UNREAD_CNT_REQUEST
getUnreadMsgCounter
CID_MSG_UNREAD_CNT_RESPONSE
CID_MSG_UNREAD_CNT_RESPONSE
HandleUnreadMsgCountResponse

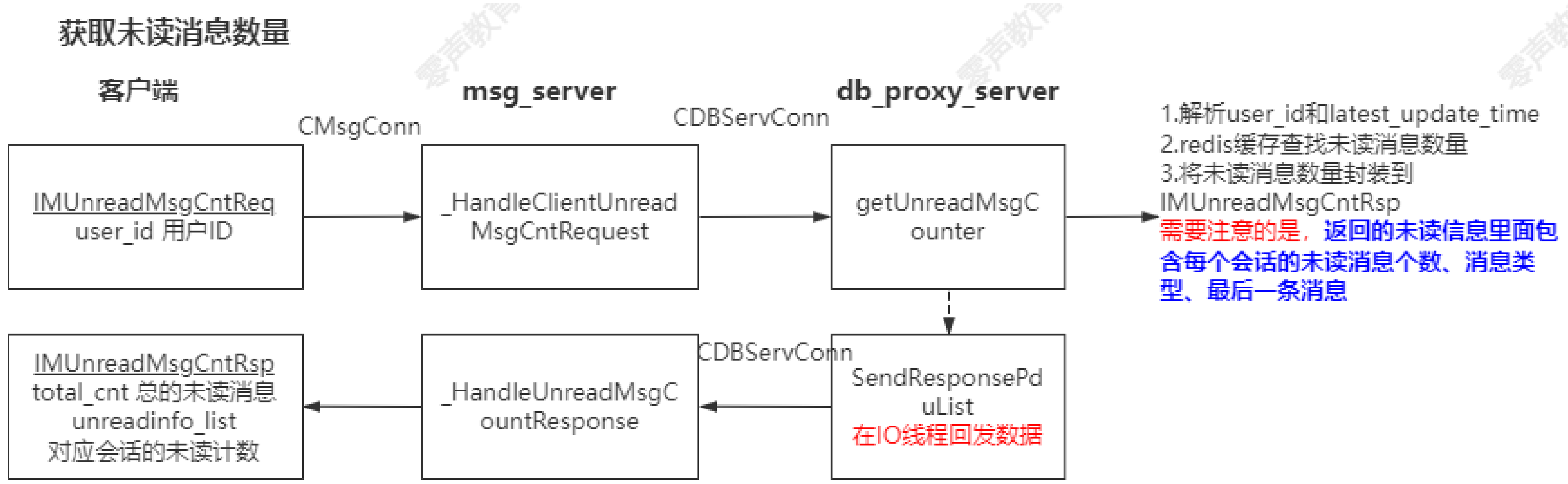
1.从redis获取未读消息数量
2.从mysql获取最后一条未读消息
3.设置每个会话的未读消息数量
4.打包未读消息信息



```
message UnreadInfo{
    required uint32 session_id = 1;
    required SessionType session_type = 2;
    required uint32 unread_cnt = 3;
    required uint32 latest_msg_id = 4;
    required bytes latest_msg_data = 5;
    required MsgType latest_msg_type = 6;
    required uint32 latest_msg_from_user_id = 7;
}
```

涉及redis和mysql

1. 8获取未读消息数量2



获取未读消息数量

客户端

msg_server

db_proxy_server

CMsgConn

CDBServConn

IMUnreadMsgCntReq
user_id 用户ID

_HandleClientUnread
MsgCntRequest

getUnreadMsgC
ounter

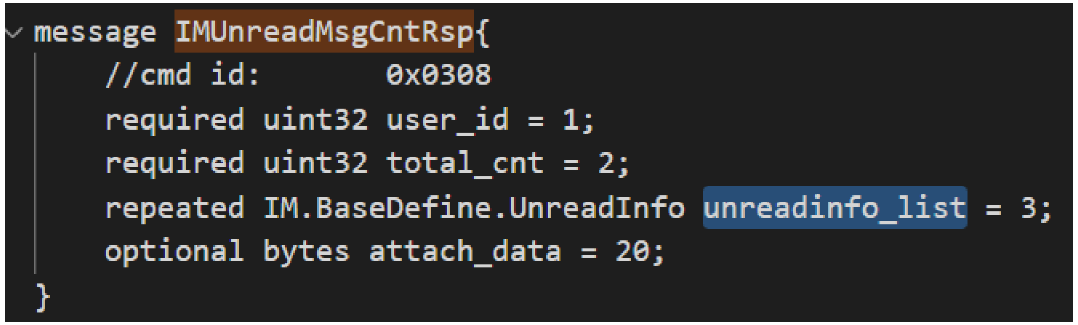
1.解析user_id和latest_update_time
2.redis缓存查找未读消息数量
3.将未读消息数量封装到
IMUnreadMsgCntRsp
需要注意的是，返回的未读信息里面包
含每个会话的未读消息个数、消息类
型、最后一条消息

CDBServConn

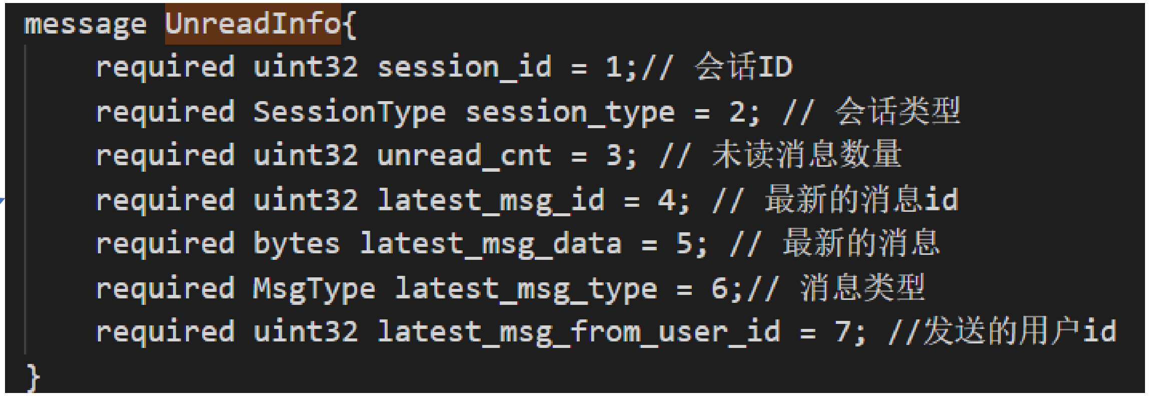
SendResponsePd
uList
在IO线程回发数据

_HandleUnreadMsgC
ountResponse

IMUnreadMsgCntRsp
total_cnt 总的未读消息
unreadinfo_list
对应会话的未读计数

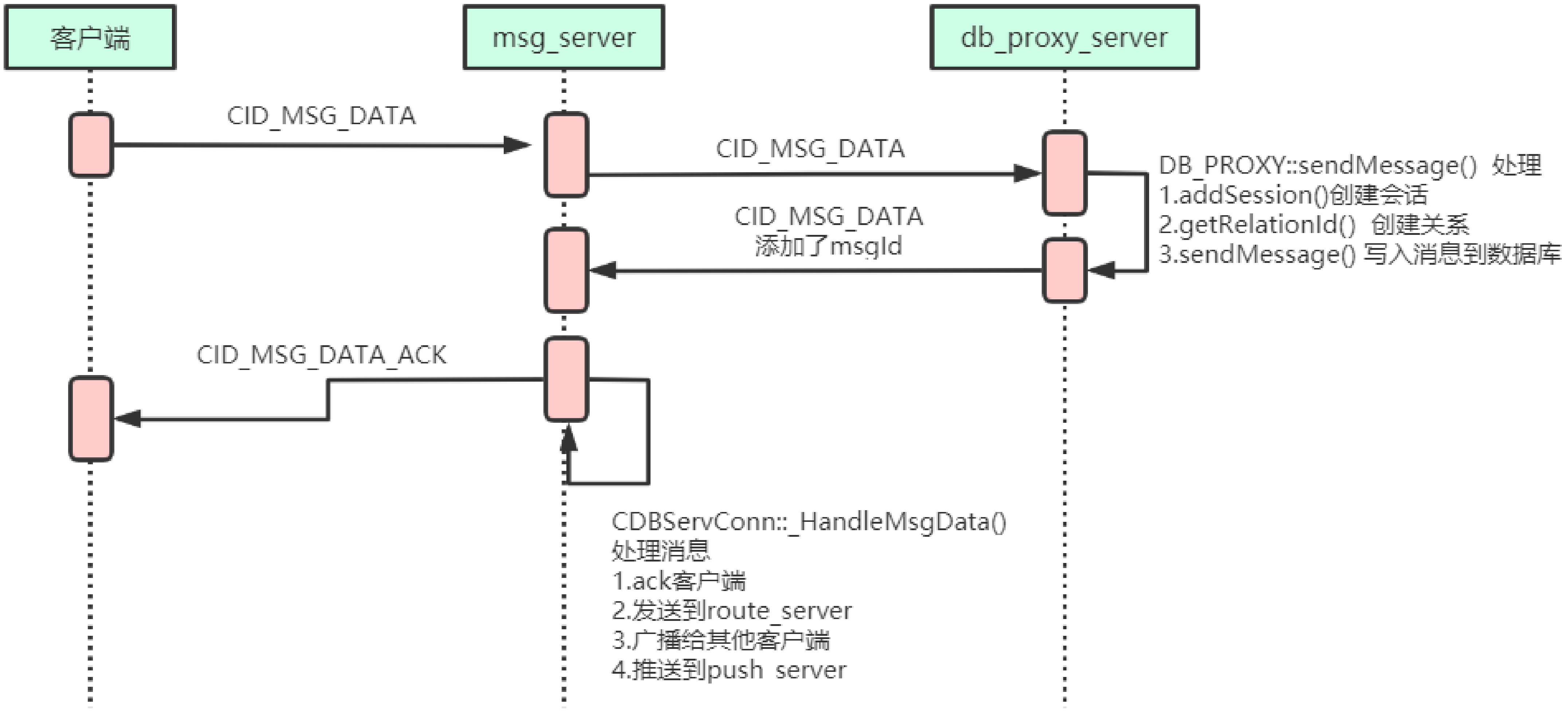


```
message IMUnreadMsgCntRsp{
    //cmd id:       0x0308
    required uint32 user_id = 1;
    required uint32 total_cnt = 2;
    repeated IM.BaseDefine.UnreadInfo unreadinfo_list = 3;
    optional bytes attach_data = 20;
}
```



```
message UnreadInfo{
    required uint32 session_id = 1;// 会话ID
    required SessionType session_type = 2; // 会话类型
    required uint32 unread_cnt = 3; // 未读消息数量
    required uint32 latest_msg_id = 4; // 最新的消息id
    required bytes latest_msg_data = 5; // 最新的消息
    required MsgType latest_msg_type = 6;// 消息类型
    required uint32 latest_msg_from_user_id = 7; //发送的用户id
}
```

1.9 msg_server发送消息



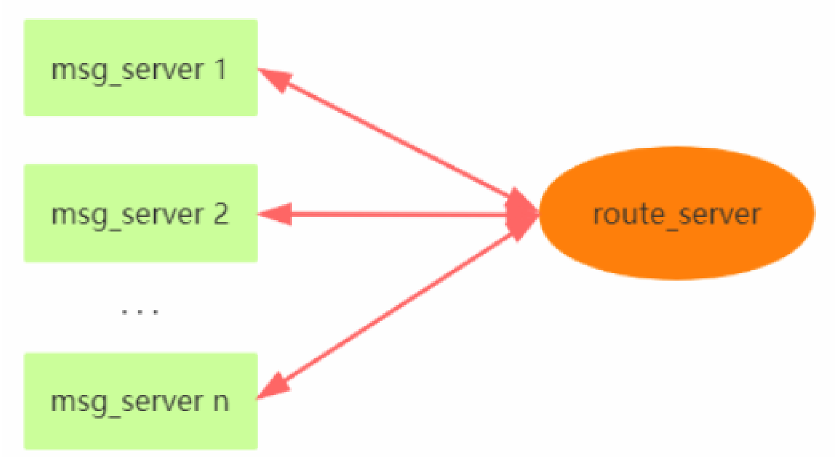
DB_PROXY::sendMessage() 处理
1.addSession()创建会话
2.getRelationId() 创建关系
3.sendMessage() 写入消息到数据库

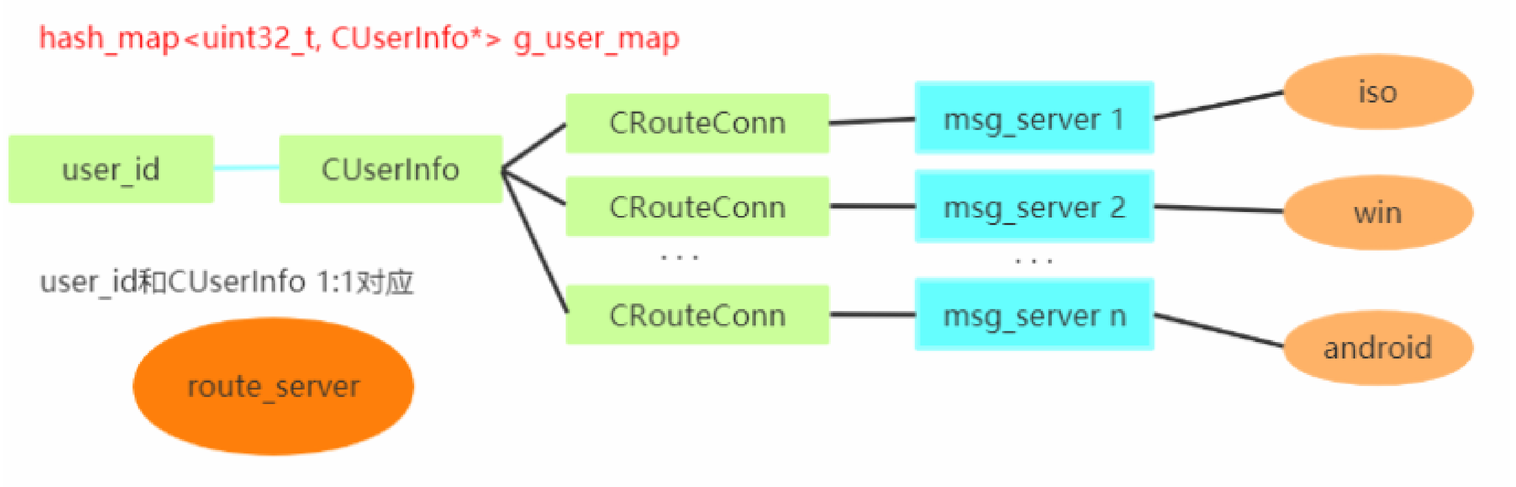
CDBServConn::_HandleMsgData()
处理消息
1.ack客户端
2.发送到route_server
3.广播给其他客户端
4.推送到push server

2 route_server

```
class CUserInfo
{
...
private:
   set<CRouteConn* > m_RouteConnSet; // 对应msg_server的连接
    // 1个user_id可能有同时登陆不同的msg_server
   map<uint32_t /* client_type */, uint32_t /* count */> m_ClientTypeList;
};
```

user_id – user - connection – 多终端 之间的关联





3 msg_server CImUser

```
class CImUser
{
map<uint32_t /* handle */, CMsgConn*>     m_conn_map;
}
```

一个用户可能有多个连接，比如PC/Android同时在线则有2个连接。

3 更多内容

见:

- 2-2-IM登录服务器和消息服务器设计-文档（msg_server部分内容）
- 3-1-思维导图-消息服务器设计.png 网页版本： https://www.yuque.com/docs/share/a086da1e-b8f3-4750-9b00-f2149386fac9?# 《3-1-思维导图-消息服务器设计》
- 3-2-思维导图-路由服务器设计.png 网页版本： https://www.yuque.com/docs/share/a4e526a2-e376-4ecf-ba7f-946446435473?# 《3-2-思维导图-路由服务器设计》
- 3-3-IM路由服务器route_server.pdf