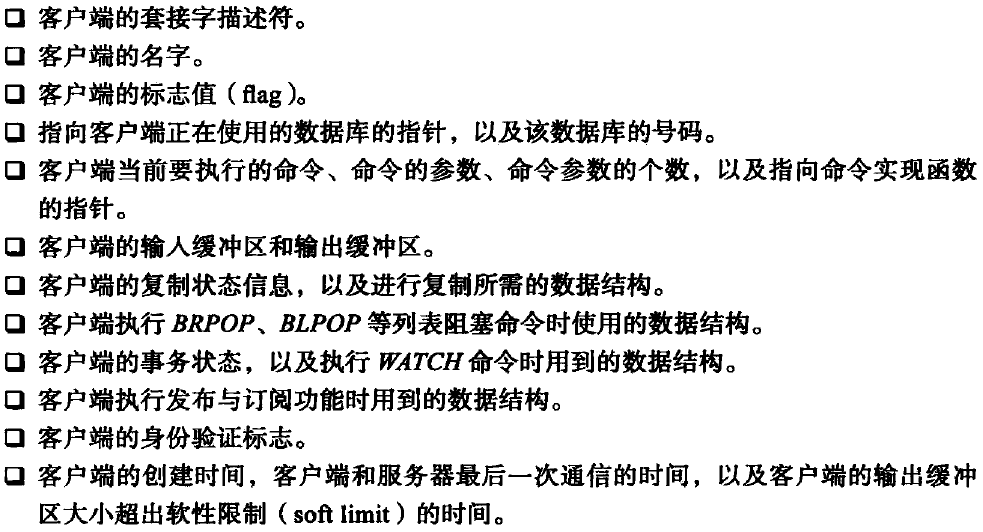
# Redis客户端

Redis使用redisClient结构保存客户端状态，这个结构保存了客户端当前的状态信息，以及执行相关功能时需要用到的数据结构，其中包括：



还有，Redis服务器状态结构中的clients属性是链表，该链表保存了所有与服务器连接的客户端状态。

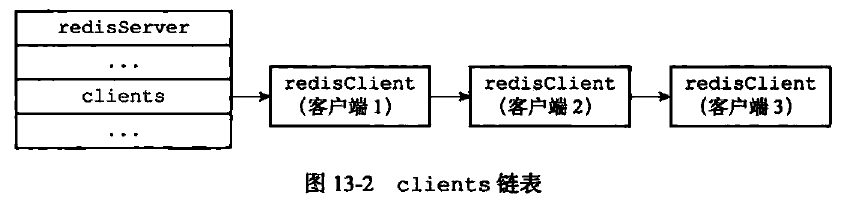
Struct redisServer{

//…

List \*client; //一个链表

//…

}



## 客户端属性

### 输入缓冲区

客户端状态的输入缓冲区用于保存客户端发送的命令请求：

Typedef struct redisClient{

//…

Sds querybuf;

//…

}

### 命令与命令参数

在服务器将客户端发送的命令请求保存到客户端状态的querybuf属性之后，服务器将对命令请求的内容进行分析，并将得出的命令参数以及命令参数的个数分别保存到客户端状态的argv属性和argc属性：

Typedef struct redisClient{

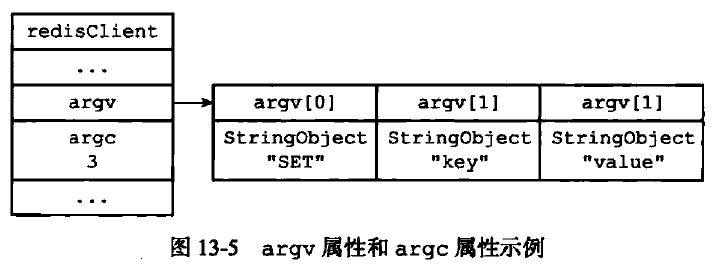
//…

Robj \*\*argv; //argv[0]为要执行的命令，之后是参数

Int argc; //数组长度

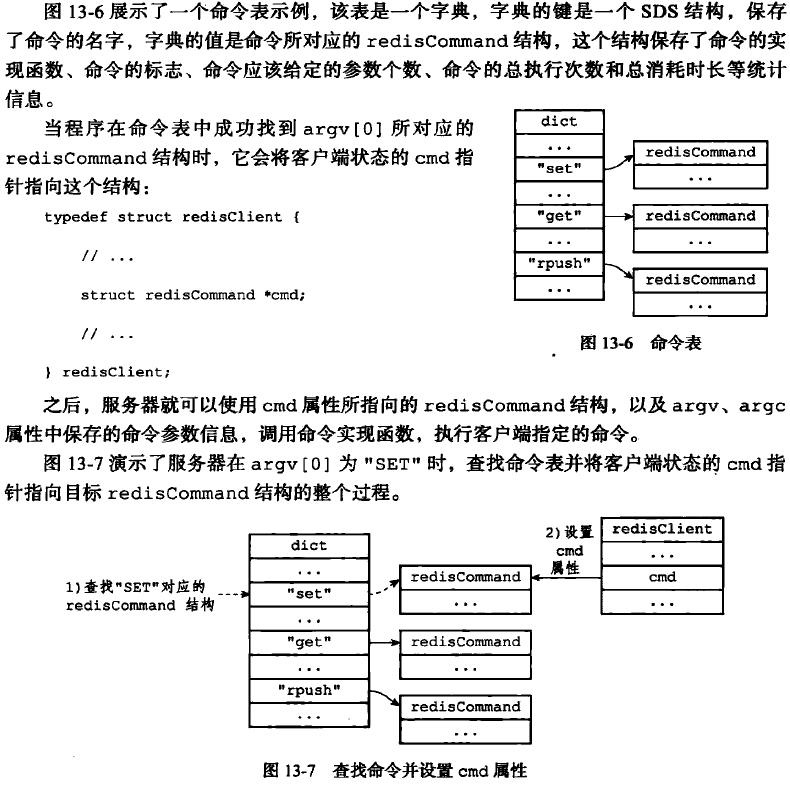
//…

}redisClient;



### 命令的实现函数

当服务器从协议内容中分析得出argv属性和argc属性的值后，服务器将根据argv[0]的只，在命令标中查找命令对应的命令实现函数。



### 输出缓冲区

命令回复会保存在输出缓冲区里面，每个客户端有两个输出缓冲区可用：

1. 固定大小的缓冲区，用于保存长度较小的回复
2. 可变大小的缓冲区，用于保存长度较大的回复

固定大小的缓冲区：

Typedef struct redisClient{

//…

Char buf[REDIS\_REPLY\_CHUNK\_BYTES];

Int bufpos; //默认值为16\*1024

//…

}

可变大小的缓冲区：

Typedef struct redisClient{

//…

List \*reply; //里面放的是字符串对象

//…

}redisClient;

### 身份验证

Authenticated属性记录客户端是否通过身份验证

Typedef struct redisClient{

//…

Int authenticated; //0未通过，1通过

//…

}redisClient;

该属性仅在身份验证功能启用时使用

### 时间

Typedef struct redisClient{

//…

time\_t ctime; //创建客户端的时间

time\_t lastinteraction; //客户端与服务器最后一次互动的时间

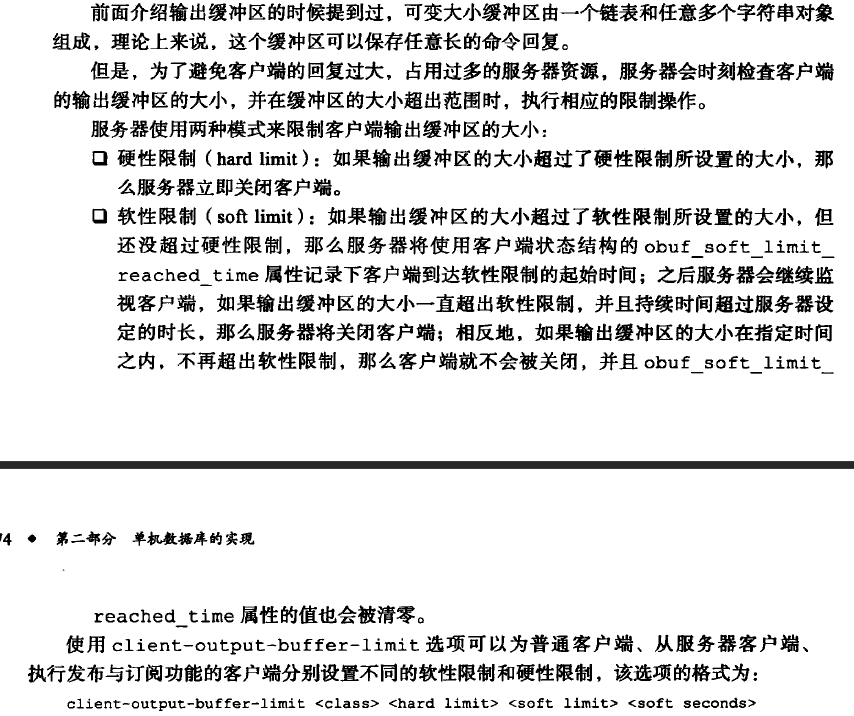
time\_t obuf\_soft\_limit\_reached\_time; //输出缓冲区第一次到达软性限制的时间

//…

}redisClient;

## 客户端的创建与关闭

### 客户端的软硬限制



### 伪客户端

