Tedu.cn
达内教育

欢迎大家来到第五阶段课程

《分布式流媒体》实训项目

预习课

TNV DAY11

预习课

预习内容

客户机（11）

预习课

客户机（11）

主函数(main)

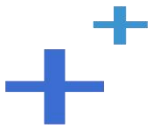
- 打印命令行用法：usage
  - 从跟踪服务器获取组列表
    - client <跟踪服务器地址表> groups
  - 向存储服务器上传文件
    - client <跟踪服务器地址表> upload <应用ID> <用户ID> <文件路径>
  - 向存储服务器询问文件大小
    - client <跟踪服务器地址表> filesize <应用ID> <用户ID> <文件ID>
  - 从存储服务器下载文件
    - client <跟踪服务器地址表> download <应用ID> <用户ID> <文件ID> <偏移> <大小>
  - 删除存储服务器上的文件
    - client <跟踪服务器地址表> delete <应用ID> <用户ID> <文件ID>

知识讲解

主函数(main)

- 根据用户ID生成文件ID：genfileid
  - 格式化身份字符串
    - <用户ID>@<进程ID>@<线程ID>@<随机数>
  - 计算身份数字摘要
    - 计算身份字符串的MD5散列值(包含128个二进制位，即16个字节)
    - 对该MD5散列值做Base16编码(包含32个字符)
    - 取该Base16编码的中间16个字符作为身份数字摘要
  - 格式化文件ID字符串
    - <当前系统时间<秒><微秒>><身份数字摘要><计数值><随机数>

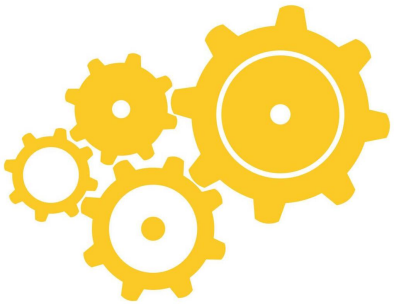
知识讲解

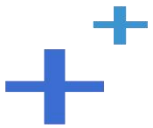


主函数(main)

- 主函数：main
  - 初始化ACL库
  - 日志打印到标准输出
  - 初始化客户机
  - 实例化客户机对象
  - 根据命令行参数执行不同的业务
    - 从跟踪服务器获取组列表
    - 向存储服务器上传文件
    - 向存储服务器询问文件大小
    - 从存储服务器下载文件
    - 删除存储服务器上的文件
  - 终结化客户机

知识讲解





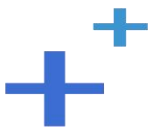
附录：程序清单

TNV/src/05_client/09_main.cpp

知识讲解

```
//客户机
//定义主函数
//
#include <unistd.h>
#include <lib_acl.h>
#include <lib_acl.hpp>
#include "01_types.h"
#include "07_client.h"

//打印命令行用法
void usage(char const* cmd) {
        fprintf(stderr, "Groups   : %s <taddrs> groups\n", cmd);
        fprintf(stderr, "Upload   : %s <taddrs> upload    "
                "<appid> <userid> <filepath>\n", cmd);
        fprintf(stderr, "Filesize: %s <taddrs> filesize "
```



TNV/src/05_client/09_main.cpp

```
                "<appid> <userid> <fileid>\n", cmd);
        fprintf(stderr, "Download: %s <taddrs> download "
                "<appid> <userid> <fileid> <offset> <size>\n", cmd);
        fprintf(stderr, "Delete  : %s <taddrs> delete   "
                "<appid> <userid> <fileid>\n", cmd);
}

// 根据用户ID生成文件ID
std::string genfileid(char const* userid) {
        srand(time(NULL));

        struct timeval now;
        gettimeofday(&now, NULL);

        acl::string str;
```

知识讲解

TNV/src/05_client/09_main.cpp


```
str.format("%s@%d@%lX@%d",
        userid, getpid(), acl_pthread_self(), rand());

acl::md5 md5;
md5.update(str.c_str(), str.size());
md5.finish();

char buf[33] = {};
strncpy(buf, md5.get_string(), 32);
memmove(buf, buf + 8, 16);
memset(buf + 16, 0, 16);

static int count = 0;
if (count >= 8000)
        count = 0;
```

TNV/src/05_client/09_main.cpp

```
        acl::string fileid;
        fileid.format("%08lx%06lx%s%04d%02d", now.tv_sec,
                now.tv_usec, buf, ++count, rand() % 100);

        return fileid.c_str();
}

int main(int argc, char* argv[]) {
        char const* cmd = argv[0];

        if (argc < 3) {
                usage(cmd);
                return -1;
        }
```

TNV/src/05_client/09_main.cpp

知识讲解

```
char const* taddrs = argv[1];
char const* subcmd = argv[2];

//初始化ACL库
acl::acl_cpp_init();

//日志打印到标准输出
acl::log::stdout_open(true);

//初始化客户机
if (client_c::init(taddrs) != OK)
        return -1;

client_c client; //客户机对象
```

TNV/src/05_client/09_main.cpp

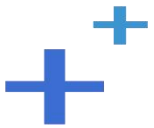

```
//从跟踪服务器获取组列表
if (!strcmp(subcmd, "groups")) {
        std::string groups;
        if (client.groups(groups) != OK) {
                client_c::deinit();
                return -1;
        }
        printf("%s\n", groups.c_str());
}
else
//向存储服务器上传文件
if (!strcmp(subcmd, "upload")) {
        if (argc < 6) {
                client_c::deinit();
                usage(cmd);
```

TNV/src/05_client/09_main.cpp


```
            return -1;
        }
        char const* appid    = argv[3];
        char const* userid   = argv[4];
        std::string fileid   = genfileid(userid);
        char const* filepath = argv[5];
        if (client.upload(appid, userid,
                fileid.c_str(), filepath) != OK) {
            client_c::deinit();
            return -1;
        }
        printf("Upload success: %s\n", fileid.c_str());
    }
    else
    //向存储服务器询问文件大小
```

TNV/src/05_client/09_main.cpp


```
if (!strcmp(subcmd, "filesize")) {
        if (argc < 6) {
                client_c::deinit();
                usage(cmd);
                return -1;
        }
        char const* appid    = argv[3];
        char const* userid   = argv[4];
        char const* fileid   = argv[5];
        long long   filesize = 0;
        if (client.filesize(appid, userid, fileid, &filesize) != OK) {
                client_c::deinit();
                return -1;
        }
        printf("Get filesize success: %lld\n", filesize);
```



TNV/src/05_client/09_main.cpp


```
}
else
//从存储服务器下载文件
if (!strcmp(subcmd, "download")) {
        if (argc < 8) {
                client_c::deinit();
                usage(cmd);
                return -1;
        }
        char const* appid    = argv[3];
        char const* userid   = argv[4];
        char const* fileid   = argv[5];
        long long   offset   = atoll(argv[6]);
        long long   size     = atoll(argv[7]);
        char*       filedata = NULL;
```

TNV/src/05_client/09_main.cpp


```
            long long   filesize = 0;
            if (client.download(appid, userid,
                    fileid, offset, size, &filedata, &filesize) != OK) {
                    client_c::deinit();
                    return -1;
            }
            printf("Download success: %lld\n", filesize);
            free(filedata);
    }
    else
    //删除存储服务器上的文件
    if (!strcmp(subcmd, "delete")) {
            if (argc < 6) {
                    client_c::deinit();
                    usage(cmd);
```

TNV/src/05_client/09_main.cpp


```
                return -1;
        }
        char const* appid  = argv[3];
        char const* userid = argv[4];
        char const* fileid = argv[5];
        if (client.del(appid, userid, fileid) != OK) {
                client_c::deinit();
                return -1;
        }
        printf("Delete success: %s\n", fileid);
}
else {
        client_c::deinit();
        usage(cmd);
        return -1;
```

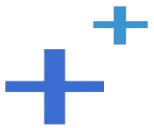
TNV/src/05_client/09_main.cpp

```
        }

        //终结化客户机
        client_c::deinit();

        return 0;
}
```

知识讲解



直播课见