作业二 董天智 2017100937

1. 分布式系统中使用多线程特别具有吸引力，因为分布式系统通常会部署在多台机器上，一台机器需要同时处理来自多台其他系统中的机器的交互请求。如果不使用多线程，则每台机器只能在同一时间相应来自一台外部机器的交互请求，这将大大降低整体系统的交互性能。

多线程服务器是指服务器能够利用多线程同时响应来自多个客户端的交互请求。好处显而易见，客户端不必等待上一个连接结束才能进行下一个连接，而是随时可以和服务器顺畅的通信而不用在乎已经有多少客户端和服务器建立了连接。

服务器端程序代码：

服务器端主要功能是监听本地的8888端口，然后等待客户端和自己建立连接。每当有一个客户端和自己建立连接，服务端就会启动一个新的线程来处理客户端的连接，然后自己等待下一个客户端的连接。

def server():

HOST = '' # Symbolic name meaning all available interfaces

PORT = 8888 # Arbitrary non-privileged port

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

s.bind((HOST, PORT))

s.listen(100)

print('Socket now listening on port: ' + str(PORT))

# now keep talking with the client

while True:

# wait for client connect

conn, addr = s.accept()

print('Connected with ' + addr[0] + ':' + str(addr[1]))

#start a new thread to handle connection

start\_new\_thread(clientthread, (conn, addr))

# Function for handling connections. This will be used to create threads

def clientthread(conn, addr):

FBASE = 'filebase\_server'

# infinite loop so that function do not terminate and thread do not end.

# now keep talking with the client

while True:

# wait to accept command

cmd = conn.recv(4096)

if not cmd:

print('Disconnected with ' + addr[0] + ':' + str(addr[1]))

break

cmd\_recv = str(cmd, encoding='utf-8').strip()

if cmd\_recv == 'upload':

filename\_b = read\_till\_End(conn)

conn.sendall(End)

data\_b = read\_till\_End(conn)

try:

with open('%s/%s' % (FBASE, str(filename\_b, encoding='utf-8')), 'wb') as f:

f.write(data\_b)

message = 'upload success'

except Exception as err:

message = 'upload failed'

print(message)

conn.sendall(bytes(message, encoding='utf-8'))

elif cmd\_recv == 'download':

filename\_b = read\_till\_End(conn)

filename = '%s/%s' % (FBASE, str(filename\_b, encoding='utf-8'))

if os.path.exists(filename):

with open(filename, 'rb') as f:

conn.sendfile(f)

conn.sendall(End)

print('download success')

else:

conn.sendall(End)

print('download failed')

elif cmd\_recv == 'list':

files = os.listdir(FBASE)

file\_info = '\n'.join(files)

conn.sendall(bytes(file\_info, encoding='utf-8'))

conn.sendall(End)

print(file\_info)

elif cmd\_recv == 'quit':

print('Disconnected with ' + addr[0] + ':' + str(addr[1]))

break

else:

reply = 'command not defined: ' + cmd\_recv

conn.sendall(bytes(reply, encoding='utf-8'))

print(reply)

# came out of loop

conn.close()

def send\_append\_End(s, data\_b):

s.sendall(data\_b)

s.sendall(End)

def read\_till\_End(s):

total\_data = []

while True:

data = s.recv(8192)

if End in data:

total\_data.append(data[:data.find(End)])

break

total\_data.append(data)

if len(total\_data) > 1:

# check if end\_of\_data was split

last\_pair = total\_data[-2] + total\_data[-1]

if End in last\_pair:

total\_data[-2] = last\_pair[:last\_pair.find(End)]

total\_data.pop()

break

return b''.join(total\_data)

客户端程序代码：

客户端的主要功能是和服务器端建立连接，然后接受用户的命令再传递给服务器端，执行相应的交互。

def client():

host = 'localhost'

port = 8888

# create an AF\_INET, STREAM socket (TCP)

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# establish connect

remote\_ip = socket.gethostbyname(host)

s.connect((remote\_ip, port))

print('Socket Connected to ' + host + ' on ip ' + remote\_ip + ' and port ' + str(port))

while True:

try:

print('\n请选择要进行的操作：\n'

'upload upload a file to server file base\n'

'download download a file from server file base\n'

'list list all files available on server\n'

'quit quit program')

cmd = sys.stdin.readline().strip('\n')

s.sendall(bytes(cmd, encoding='utf-8'))

if cmd == 'exit':

break

elif cmd == 'upload':

print('please specify file path:')

filepath = sys.stdin.readline().strip('\n')

if not os.path.exists(filepath):

print('file "%s" not exists' % filepath)

continue

send\_append\_End(s, bytes(os.path.basename(filepath), encoding='utf-8'))

s.recv(1024) # 用于阻塞进程

with open(filepath, 'rb') as f:

s.sendfile(f)

s.sendall(End)

res = s.recv(4096)

print(str(res, encoding='utf-8'))

elif cmd == 'download':

print('please specify file name:')

filename = sys.stdin.readline().strip('\n')

send\_append\_End(s, bytes(filename, encoding='utf-8'))

data\_b = read\_till\_End(s)

if len(data\_b) == 0:

print('file "%s" not exists' % filename)

else:

with open('filebase\_client/%s' % filename, 'wb') as f:

f.write(data\_b)

print('download success')

elif cmd == 'list':

data\_b = read\_till\_End(s)

file\_info = str(data\_b, encoding='utf-8')

print(file\_info)

elif cmd == 'quit':

break

else:

da = s.recv(4096)

print(str(da, encoding='utf-8'))

except socket.error:

# Send failed

print('Send failed')

sys.exit()

s.close()

def send\_append\_End(s, data\_b):

s.sendall(data\_b)

s.sendall(End)

def read\_till\_End(s):

total\_data = []

while True:

data = s.recv(8192)

if End in data:

total\_data.append(data[:data.find(End)])

break

total\_data.append(data)

if len(total\_data) > 1:

# check if end\_of\_data was split

last\_pair = total\_data[-2] + total\_data[-1]

if End in last\_pair:

total\_data[-2] = last\_pair[:last\_pair.find(End)]

total\_data.pop()

break

return b''.join(total\_data)

2、

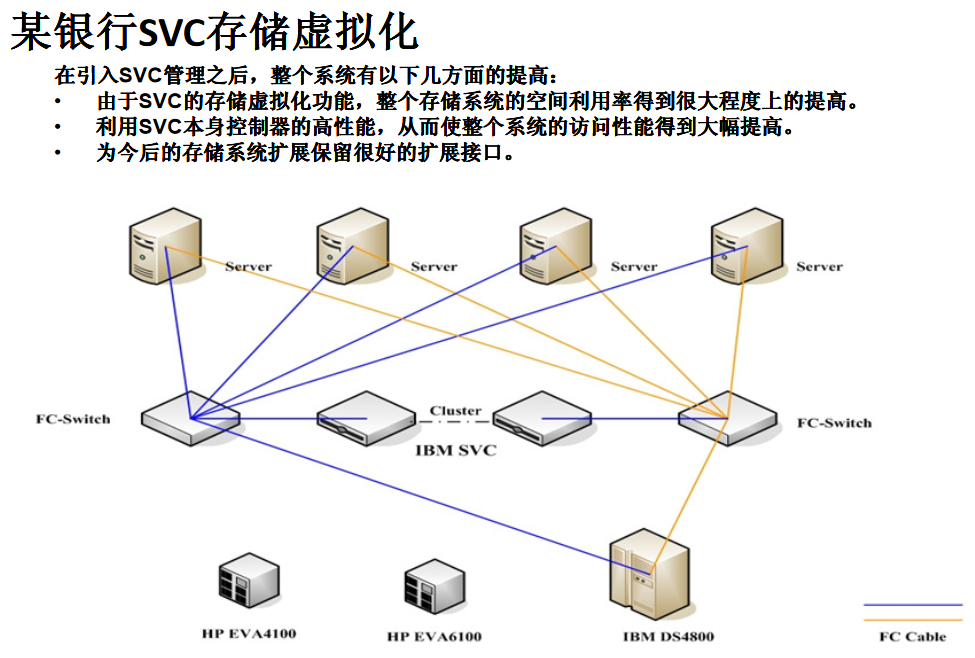
2.1什么是虚拟机

虚拟机（Virtual Machine）指通过软件模拟的具有完整硬件系统功能的、运行在一个完全隔离环境中的完整计算机系统。（百度百科）

2.2什么是虚拟化，虚拟化的关键特征是什么，虚拟化在分布式系统中的作用有哪些

虚拟化，是指通过虚拟化技术将一台计算机虚拟为多台逻辑计算机。在一台计算机上同时运行多个逻辑计算机，每个逻辑计算机可运行不同的操作系统，并且应用程序都可以在相互独立的空间内运行而互不影响，从而显著提高计算机的工作效率。（百度百科）虚拟化的关键特征是独立，包括硬件独立和软件独立，即每个逻辑计算机之间的硬件和软件是独立的。在分布式系统中，虚拟化能够将一台服务器虚拟化成多台虚拟服务器，提高服务器资源的利用率。

2.3举例说明



从图中可以看出，采用存储虚拟化技术后，系统的空间利用率得到很大的提高，而且虚拟化系统的高性能也是的整个系统的访问性能得到大幅度提高。（图片来自IBM）

3、

3.1 代码迁移是什么

代码迁移是指将一台机器上运行的代码迁移到其他机器上。

3.2 代码迁移的理由有哪些

代码迁移可以提高系统的性能，比如服务器和客户端程序，把服务器端的部分代码迁移到客户端运行，可以使得客户端响应速度更快等。另外，如果原先运行代码的机器比较陈旧，把代码迁移到更新的机器上，则可以实现运行速度的提高。

3.3 代码迁移的模型有哪些

第一层，分为弱移动和强移动。弱移动只移动代码和初始化的数据，最后被目标进程执行或者另外一个独立的进程。强移动移动执行部分，要么复制进程(正在执行的进程停下来，移动后再恢复)，要么克隆(所有数据完全复制到另外一台机器上，和原来的进程并行)。

第二层，发送者初始化(sender-initiated mobility)或者接收者初始化(receiver-initiated mobility)。（百度知道）

3.4 同构系统和异构系统中代码迁移的差别和挑战，举例说明

同构系统中，代码迁移相对容易，通常在保证系统的各项依赖满足的情况下，可以直接进行代码迁移。而在异构系统中，代码迁移通常就比较困难，因为很多系统的依赖得不到满足。比如我在linux系统中运行的代码，调用了某个只有linux系统才有的外部程序包，这时候想把代码迁移到windows系统下就非常困难，除非可以自己把那个linux的包在windows下重新实现一遍，否则很难完成代码迁移。

当然，有一些语言在发明之初就是为了解决异构平台之间代码迁移的困难的。比如java。我们知道java程序是必须运行在java虚拟机上的。即只要机器安装了java虚拟机，不管它是同构还是异构系统，java代码均可以在上面运行。这在很大程度上解决了代码迁移的难点。也因此，java是目前非常流行的开发语言，收到了很多大公司的青睐。