**2024《区块链技术与应用》**

**（微众银行）**

**实验1：区块链理解与实践**

1. **实验背景与目标**

本实验旨在通过搭建FISCO BCOS区块链环境，让初学者深入理解区块链技术的工作原理和应用实践，特别是在共识算法和智能合约部署方面的实际操作。

**实验目标**

* 理解区块链基础：通过实验，参与者将能够理解区块链的基本概念，包括分布式账本、加密技术、共识机制等。
* 掌握FISCO BCOS平台：学习如何在Unix环境下搭建和配置FISCO BCOS区块链网络，为进一步的区块链开发和研究打下基础。
* 智能合约开发与部署：通过部署和调用HelloWorld合约，参与者将学习智能合约的编写、部署和调用过程，理解智能合约在区块链中的应用。
* 共识算法实践：通过模拟PBFT（Practical Byzantine Fault Tolerance）共识算法的容错机制，参与者将能够深入理解区块链网络中的共识过程和容错能力。

**实验步骤概述**

* **准备Unix环境**：安装和配置Unix操作系统，为后续的区块链搭建提供基础平台。
* **准备实验环境：**安装必要的软件和依赖，如curl、wget、openssl、openjdk等，确保环境满足FISCO BCOS区块链的运行要求。
* **在命令行搭建FISCO BCOS区块链：**通过命令行工具搭建FISCO BCOS区块链网络，实现节点的配置和管理。
* **使用控制台部署调用HelloWorld合约：**部署一个简单的HelloWorld智能合约，通过控制台进行合约的调用和交互，体验智能合约的功能。
* **PBFT共识算法容错模拟实践：**模拟PBFT共识算法的容错过程，观察并分析在不同网络条件下的区块链网络表现，理解共识算法的重要性和作用。

1. **准备Unix环境**

**注意：**若你已经准备好Unix环境（包括但不限于：Ubuntu、CentOS、macOS、WindowsWSL等）可直接跳过本章节，进行下一章节的操作。

安装Unix系统目前在互联网已经有非常多方法，且与本实验相关性不大，本文提供一些可靠的学习链接，学生可自行学习。

[虚拟机||使用VirtualBox安装Ubuntu详细图文教程（安装+调整分辨率+共享文件）\_virtualbox安装ubuntu12-CSDN博客](https://blog.csdn.net/Inochigohan/article/details/119791518)

[VMware 安装配置 Ubuntu（最新版、超详细）\_vmware-workstation-full-17.5.1-23298084.exe-CSDN博客](https://blog.csdn.net/m0_70885101/article/details/137694608)

[超详细Windows10/Windows11 子系统（WSL2）安装Ubuntu20.04（带桌面环境）\_wsl安装ubuntu20.04-CSDN博客](https://blog.csdn.net/weixin_44301630/article/details/122390018)

1. **准备实验环境**

**注意：**若你已经在上个步骤的系统环境中准备好curl、wget、openssl、openjdk等基本环境依赖，以及已经安装VSCode、IntelliJ IDEA等的现代IDE，可跳过本章节，进行下一章节的操作。

<https://fisco-bcos-doc.readthedocs.io/zh-cn/latest/docs/quick_start/air_installation.html#id2>

安装macOS依赖

*# 最新homebrew默认下载的为openssl@3，需要指定版本openssl@1.1下载*

brew install openssl@1.1 curl wget openjdk@8

安装ubuntu依赖

sudo apt install -y curl openssl wget default-jdk

安装centos依赖

sudo yum install -y curl openssl openssl-devel wget java java-devel

[VSCode 安装使用教程 环境安装配置 保姆级教程\_vscode安装包-CSDN博客](https://blog.csdn.net/weixin_46474921/article/details/132841711)

[Intellij IDEA社区版下载与安装教程\_idea官网社区版-CSDN博客](https://blog.csdn.net/wangmeixi/article/details/103841842)

1. **在命令行搭建FISCO BCOS区块链**

打开系统的终端工具，创建操作目录。本实验示例以fisco文件夹为例。

**创建操作目录并下载建链脚本**

|  |
| --- |
| *# 创建操作目录*  cd ~ && mkdir -p fisco && cd fisco  *# 下载建链脚本*  curl -#LO https://github.com/FISCO-BCOS/FISCO-BCOS/releases/download/v3.11.0/build\_chain.sh && chmod u+x build\_chain.sh  *# Note: 若访问git网速太慢，可尝试如下命令下载建链脚本:*  curl -#LO https://osp-1257653870.cos.ap-guangzhou.myqcloud.com/FISCO-BCOS/FISCO-BCOS/releases/v3.11.0/build\_chain.sh && chmod u+x build\_chain.sh |

**快速搭建4节点的区块链**

**注意：**在搭建之前请确保机器的30300~30303，20200~20203端口没有被占用。

bash build\_chain.sh -l 127.0.0.1:4 -p 30300,20200

此处的127.0.0.1:4的意思是在127.0.0.1的IP上构建4个节点，30300,20200的意思是所构建的四个节点中P2P端口以30300开始，RPC端口以20200开始。搭建脚本build\_chain.sh介绍文档 [参考这里](https://fisco-bcos-doc.readthedocs.io/zh-cn/latest/docs/tutorial/air/build_chain.html)

命令成功后会输出All completed：

|  |
| --- |
| [INFO] Generate ca cert successfully!  Processing IP:127.0.0.1 Total:4  writing RSA key  [INFO] Generate ./nodes/127.0.0.1/sdk cert successful!  writing RSA key  [INFO] Generate ./nodes/127.0.0.1/node0/conf cert successful!  writing RSA key  [INFO] Generate ./nodes/127.0.0.1/node1/conf cert successful!  writing RSA key  [INFO] Generate ./nodes/127.0.0.1/node2/conf cert successful!  writing RSA key  [INFO] Generate ./nodes/127.0.0.1/node3/conf cert successful!  [INFO] Downloading get\_account.sh from https://osp-1257653870.cos.ap-guangzhou.myqcloud.com/FISCO-BCOS/FISCO-BCOS/tools/get\_account.sh...  *############################################################################################################################################################### 100.0%*  [INFO] Admin account: 0x4c7239cfef6d41b7322c1567f082bfc65c69acc5  [INFO] Generate uuid success: 167A2233-5444-4CA4-8792-C8E68130D5FC  [INFO] Generate uuid success: CC117CAF-224C-4940-B548-6DED31D24B18  [INFO] Generate uuid success: 16B5E4BD-51C1-416E-BF44-6D1BB05F7666  [INFO] Generate uuid success: 60DD77F2-F3A5-49F2-8C7F-8151E8823C6D  ==============================================================  [INFO] GroupID : group0  [INFO] ChainID : chain0  [INFO] fisco-bcos path : bin/fisco-bcos  [INFO] Auth mode : false  [INFO] Start port : 30300 20200  [INFO] Server IP : 127.0.0.1:4  [INFO] SM model : false  [INFO] enable HSM : false  [INFO] Output dir : ./nodes  [INFO] All completed. Files **in** ./nodes |

**启动FISCO BCOS链**

使用以下命令即可

bash nodes/127.0.0.1/start\_all.sh

启动成功会输出如下信息。否则请使用netstat -an |grep tcp检查机器30300~30303, 20200~20203端口是否被占用。

|  |
| --- |
| try to start node0  try to start node1  try to start node2  try to start node3  node3 start successfully pid=36430  node2 start successfully pid=36427  node1 start successfully pid=36433  node0 start successfully pid=36428 |

**检查节点进程**

使用以下命令查看进程是否启动：

ps aux |grep -v grep |grep fisco-bcos

正常情况会有类似下面的输出； 如果进程数不为4，则进程没有启动（一般是端口被占用导致的）

|  |
| --- |
| fisco 35249 7.1 0.2 5170924 57584 s003 S 2:25下午 0:31.63 /home/fisco/nodes/127.0.0.1/node1/../fisco-bcos -c config.ini -g config.genesis  fisco 35218 6.8 0.2 5301996 57708 s003 S 2:25下午 0:31.78 /home/fisco/nodes/127.0.0.1/node0/../fisco-bcos -c config.ini -g config.genesis  fisco 35277 6.7 0.2 5301996 57660 s003 S 2:25下午 0:31.85 /home/fisco/nodes/127.0.0.1/node2/../fisco-bcos -c config.ini -g config.genesis  fisco 35307 6.6 0.2 5301996 57568 s003 S 2:25下午 0:31.93 /home/fisco/nodes//127.0.0.1/node3/../fisco-bcos -c config.ini -g config.genesis |

1. **使用控制台部署调用HelloWorld合约**

控制台提供了向FISCO BCOS节点部署合约、发起合约调用、查询链状态等功能

1. **下载控制台**

cd ~/fisco && curl -LO https://github.com/FISCO-BCOS/console/releases/download/v3.7.0/download\_console.sh && bash download\_console.sh

如果因为网络问题导致长时间无法下载，请尝试 cd ~/fisco && curl -#LO [https://gitee.com/FISCO-BCOS/console/raw/master/tools/download\_console.sh && bash download\_console.sh](https://gitee.com/FISCO-BCOS/console/raw/master/tools/download_console.sh%20&&%20bash%20download_console.sh)

1. **配置控制台证书**

控制台与节点之间默认开启SSL连接，控制台需要配置证书才可连接节点。开发建链脚本在生成节点的同时，生成了SDK证书，可直接拷贝生成的证书供控制台使用：

cp -r nodes/127.0.0.1/sdk/\* console/conf

1. **启动控制台**

到控制台所在的目录下，使用脚本启动控制台

cd ~/fisco/console && bash start.sh

输出下述信息表明启动成功 否则请检查conf/config.toml中节点端口配置是否正确

Welcome to FISCO BCOS console(3.7.0)!

Type 'help' or 'h' **for** help. Type 'quit' or 'q' to quit console.

用控制台获取信息

|  |
| --- |
| *# 获取节点列表信息*  [group0]: /> getGroupPeers  peer0: 48fde62f1d2dc59a65bed2e3bb9bd199de26de6b9894e2201199726d6e9e98d090bb1d7e22c931b412728a832ffacdd2727b34fc808f5a755af6bc623c44aba6  peer1: 8e510d66644a8a6caa7e031f097f604501bc42a3851b817f65a18eede0c539f2d011349d470da74cb84a3cf88dbd64a4cc18369fa09b17dac4eec9f65975ecc2  peer2: 9d7ff3f0d3abd54054a99d17a6af27c839f8f7702a4335bdb502323c87a0d7978534a2479cfedb614e1548a869efe038fc49da442b5770aa52c0cc793ca13602  peer3: f0ffa45cee35dcc1bcf1e1ef7b7c3d96590c25ba75198a28ef5ceb89dc6bec310619cb6850231018c8d5a5d698eaf1e5669118e17ea79379211bd332896aa56a  *# 获取共识节点列表信息*  [group0]: /> getSealerList  [  Sealer{  nodeID='9d7ff3f0d3abd54054a99d17a6af27c839f8f7702a4335bdb502323c87a0d7978534a2479cfedb614e1548a869efe038fc49da442b5770aa52c0cc793ca13602',  weight=1  },  Sealer{  nodeID='f0ffa45cee35dcc1bcf1e1ef7b7c3d96590c25ba75198a28ef5ceb89dc6bec310619cb6850231018c8d5a5d698eaf1e5669118e17ea79379211bd332896aa56a',  weight=1  },  Sealer{  nodeID='8e510d66644a8a6caa7e031f097f604501bc42a3851b817f65a18eede0c539f2d011349d470da74cb84a3cf88dbd64a4cc18369fa09b17dac4eec9f65975ecc2',  weight=1  },  Sealer{  nodeID='48fde62f1d2dc59a65bed2e3bb9bd199de26de6b9894e2201199726d6e9e98d090bb1d7e22c931b412728a832ffacdd2727b34fc808f5a755af6bc623c44aba6',  weight=1  }  ] |

1. **部署和调用合约**
   1. **编写HelloWorld合约**

HelloWorld合约提供了两个接口get()和set()，用于获取/设置合约变量name，合约内容如下：

|  |
| --- |
| pragma solidity >=0.6.10 <0.8.20;  contract HelloWorld {  string name;  constructor() public {  name = "Hello, World!";  }  **function** get() public view returns (string memory) {  **return** name;  }  **function** set(string memory n) public {  name = n;  }  } |

* 1. **部署HelloWorld合约**

为了方便用户快速体验，HelloWorld合约已经内置于控制台中，位于控制台目录contracts/solidity/HelloWorld.sol，参考下面命令部署：

*# 在控制台输入以下指令 部署成功则返回合约地址*

[group0]: /> deploy HelloWorld

transaction hash: 0x796b573aece250bba891b9251b8fb464d22f41cb36e7cae407b2bd0a870f5b72

contract address: 0x6849F21D1E455e9f0712b1e99Fa4FCD23758E8F1

currentAccount: 0x7b047472a4516e9697446576f8c7fcc064f967fa

*# 查看当前块高*

[group0]: /> getBlockNumber

1

* 1. **调用HelloWorld合约**

|  |
| --- |
| *# 调用get接口获取name变量，此处的合约地址是deploy指令返回的地址*  [group0]: /> call HelloWorld 0x6849F21D1E455e9f0712b1e99Fa4FCD23758E8F1 get  ---------------------------------------------------------------------------------------------  Return code: 0  description: transaction executed successfully  Return message: Success  ---------------------------------------------------------------------------------------------  Return value size:1  Return types: (string)  Return values:(Hello, World!)  ---------------------------------------------------------------------------------------------  *# 查看当前块高，块高不变，因为get接口不更改账本状态*  [group0]: /> getBlockNumber  1  *# 调用set方法设置name*  [group0]: /> call HelloWorld 0x6849F21D1E455e9f0712b1e99Fa4FCD23758E8F1 set "Hello, FISCO BCOS"  transaction hash: 0x2f7c85c2c59a76ccaad85d95b09497ad05ca7983c5ec79c8f9d102d1c8dddc30  ---------------------------------------------------------------------------------------------  transaction status: 0  description: transaction executed successfully  ---------------------------------------------------------------------------------------------  Receipt message: Success  Return message: Success  Return value size:0  Return types: ()  Return values:()  ---------------------------------------------------------------------------------------------  Event logs  Event: {}  *# 查看当前块高，因为set接口修改了账本状态，块高增加到2*  [group0]: /> getBlockNumber  2  *# 退出控制台*  [group0]: /> exit |

**至此，我们完成了第一条FISCO-BCOS链的搭建、控制台的配置和使用，并部署和调用了第一个合约。**

1. **PBFT共识算法容错模拟实践**

在我们部署的区块链中有4个节点，我们可以选择其中的节点进行停止操作，进行模拟该节点因为故障退出的效果。

我们可选择node2、node3两个节点进行停止，操作命令如下：

cd ~/fisco/nodes/127.0.0.1/node2 && bash stop.sh

cd ~/fisco/nodes/127.0.0.1/node3 && bash stop.sh

通过逐渐停掉其中1个节点、2个节点，观察是否还能成功部署HelloWorld合约，尝试分析其原因。

提供逐渐停掉其中1个节点、2个节点，控制台连接节点截图部署HelloWorld合约的截图；尝试分析操作（成功/失败）的原因.