区块链服务(BCS)

开发指南

文档版本 01

发布日期 2023-03-27

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

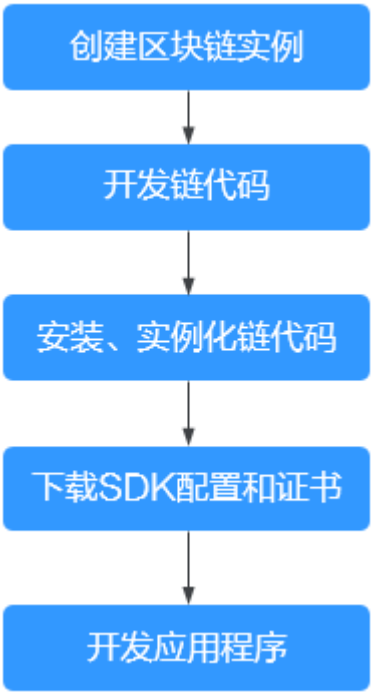
目 录

1 简介

在使用区块链服务时，您需要开发自己的链代码和应用。本文档主要介绍链代码的开发及其应用配置，支持具备Go/Java开发经验的开发人员使用。

区块链服务使用流程如下：

图 1-1 使用流程



1. 创建区块链实例

   区块链实例支持在CCE集群上部署，具体可参见**基于CCE集群**。
2. 开发链代码

   链代码是用Go语言编写的程序，主要用于操作账本上的数据，具体可参见**链代码开发**。
3. 安装、实例化链代码

区块链服务为您提供界面化链代码管理功能，包括链代码安装、实例化等，具体可参见**链代码管理**。

4. 下载SDK配置和证书

   应用程序开发前，您需要获取对应实例的SDK配置文件和证书，具体可参见**访问区块链**。

2 链代码开发

2.1 开发前准备

链代码（Chaincode）又称智能合约，是用Go、Node.js语言编写的程序，主要用于操作账本上的数据。链代码是运行在区块链上的、特定条件下自动执行的代码逻辑，是用户利用区块链实现业务逻辑的重要途径。基于区块链特点，智能合约的运行结果是可信的，其结果是无法被伪造和篡改的。

在使用区块链服务BCS时，用户需要开发自己的链代码和应用程序。用户的应用程序通过区块链网络中的Peer节点调用链代码，用户链代码通过区块链网络的Peer节点来操作账本数据。

开发环境准备

1. 安装Go开发环境。安装包下载地址为：**https://golang.org/dl/**。（请选择1.9.2之后的版本）

   各个系统对应的包名（以1.11.12版本为例）：

操作系统	包名
Windows	go1.11.12.windows-amd64.msi
Linux	go1.11.12.linux-amd64.tar.gz

   - Windows下您可以使用.msi后缀的安装包来安装。默认情况下.msi文件会安装在“C:\Go”目录下。您可以将“C:\Go\bin”目录添加到Path环境变量中。添加后您需要重启命令窗口才能生效。
   - Linux下，您需要将下载的二进制包解压至/usr/local目录。将/usr/local/go/bin目录添加至Path环境变量：

     ```
     export PATH=$PATH:/usr/local/go/bin
     ```

   安装完go语言后可以通过命令**go version**查看版本信息，以及通过**go env**命令来查看相关路径配置。

2. 安装Go编辑器。编辑器可自行选择，推荐使用Goland：**https://www.jetbrains.com/go/download**。

下载源码包

下载Fabric源码包作为三方库。

请根据实际需求，选择下载对应版本的Fabric源码包：

https://github.com/hyperledger/fabric/tree/release-2.2

说明

Fabric源码包选择和创建的区块链实例版本对应，即如果创建区块链实例时，Hyperledger Fabric增强版内核是v2.2（4.X.X版本），则Fabric源码包对应选择2.2版本。

2.2 开发规范

防止出现 panic 后链代码容器异常

说明

该内容仅适用于Go语言链代码开发。

为避免出现panic异常时链代码容器异常重启，找不到日志，导致问题无法及时定位，可在Invoke函数入口处添加defer语句时，出现panic异常时返回错误给客户端。

```
// 定义命名返回值，发生panic在defer里面赋值，确保客户端可以收到返回值
// 使用debug.PrintStack()将错误的堆栈信息打印到标准输出，方便问题定位
func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) (pr pb.Response) {
  defer func() {
    if err:=recover(); err != nil {
      fmt.Println("recover from invoke:", err)
      debug.PrintStack()
      pr = shim.Error(fmt.Sprintf("invoke painc. err: %v",err))
    }
  }()

  fmt.Println("ex02 Invoke")
  function, args := stub.GetFunctionAndParameters()
  if function == "invoke" {
    // Make payment of X units from A to B
    return t.invoke(stub, args)
  } else if function == "delete" {
    // Deletes an entity from its state
    return t.delete(stub, args)
  } else if function == "query" {
    // the old "Query" is now implemented in invoke
    return t.query(stub, args)
  }

  pr = shim.Error("Invalid invoke function name. Expecting \"invoke\" \"delete\" \"query\"")
  return pr
}
```

分批次查询数据

查询账本数据时，如果在一次查询中返回过多的数据，会导致资源占用过多，接口延时较长（超过30s时peer会中断任务），应预先估计数据量，分批次进行查询。

修改或删除账本数据调用链代码时，同样也需要根据数据量大小确定是否采取分批次操作的方式处理。

合理使用索引（账本数据存储方式为 CouchDB）

对于CouchDB来说，使用索引功能，写入数据时会消耗时间，但可明显提高数据查询速度。因此可以根据业务需要，合理的在某些字段上建立索引。

添加权限验证

对智能合约执行者的权限进行验证，防止无权限的用户执行链代码。

说明

如果业务上不要求确定的某个组织进行背书，为确保链代码上的数据不被任意组织恶意修改（自己安装非法链代码，操作数据等），建议至少两个或两个以上组织共同参与背书。

参数校验

参数（包括入参和代码中定义的各种参数）在使用前需对其个数、类型、长度、取值范围等做校验，验证其合法性，防止出现数组越界等问题。

日志处理

开发时需要对业务逻辑复杂、容易出错的地方，使用fmt打印日志，便于调测。但是在开发调测结束后，需要将其删除，因为fmt会消耗时间和资源。

依赖配置

说明

该内容仅适用于Java语言链代码开发。

请使用Gradle或Maven构建管理工具组织链代码项目。若链代码项目中包含非本地依赖，请确保对应区块链实例的节点均绑定了弹性ip。若链代码容器将运行在受限网络环境，请确保项目中的所有依赖已配置为本地依赖。示例链代码获取方法：登录区块链服务BCS控制台，进入“应用案例”，单击“Java示例Demo-Java SDK Demo”中“Chaincode_Java_Local_Demo”的“下载”按钮。

2.3 Go 语言链代码开发

2.3.1 链代码结构

链代码即一个Go文件，创建好文件后进行函数开发等操作。

链代码接口

- 链代码启动必须通过1.4风格（调用shim包）中的Start函数，入参为shim包中定义的Chaincode接口类型。实际开发中， 您需要自行定义一个结构体，实现Chaincode接口。

```
type Chaincode interface {
  Init(stub ChaincodeStubInterface) pb.Response
  Invoke(stub ChaincodeStubInterface) pb.Response
}
```

- 2.2风格（使用fabric-contract-api-go包）的链代码实际开发中， 您需要自行定义一个结构体，实现Chaincode接口。

```
type Chaincode interface {
    Init(ctx contractapi.TransactionContextInterface, args…) error
    Invoke(ctx contractapi.TransactionContextInterface, args…) error
}
```

链代码结构

- 1.4风格Go语言的链代码结构如下：

```
package main

// 引入必要的包
import (
  "github.com/hyperledger/fabric/core/chaincode/shim"
  pb "github.com/hyperledger/fabric/protos/peer"
)

// 声明一个结构体
type SimpleChaincode struct {}

// 为结构体添加Init方法
func (t *SimpleChaincode) Init(stub shim.ChaincodeStubInterface) pb.Response {
        // 在该方法中实现链代码初始化或升级时的处理逻辑
    // 编写时可灵活使用stub中的API
}

// 为结构体添加Invoke方法
func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response {
        // 在该方法中实现链代码运行中被调用或查询时的处理逻辑
    // 编写时可灵活使用stub中的API
}

//主函数，需要调用shim.Start()方法
func main() {
  err := shim.Start(new(SimpleChaincode))
  if err != nil {
    fmt.Printf("Error starting Simple chaincode: %s", err)
  }
}
```

- 2.2风格Go语言的链代码结构如下：

```
package main

// 引入必要的包
import (
 "github.com/hyperledger/fabric/plugins/fabric-contract-api-go/contractapi")



// 声明一个结构体
type Chaincode struct {
    contractapi.Contract
}

// 为结构体添加Init方法
func (ch * Chaincode) Init(ctx contractapi.TransactionContextInterface, args…) error {
    // 在该方法中实现链代码初始化或升级时的处理逻辑
}

// 为结构体添加Invoke方法
func (ch * Chaincode) Invoke(ctx contractapi.TransactionContextInterface, args…) error {
    // 在该方法中实现链代码运行中被调用或查询时的处理逻辑
}

//主函数
func main() {
    cc, err := contractapi.NewChaincode(new(ABstore))
    if err != nil {
        panic(err.Error())
    }
```

```
    if err := cc.Start(); err != nil {
        fmt.Printf("Error starting ABstore chaincode: %s", err)
    }
}
```

2.3.2 链代码相关的 API

Fabric源码包中的shim包提供了如下几种类型的接口，您可以参考使用：

- 参数解析API：调用链代码时需要给被调用的目标函数/方法传递参数，该API提供解析这些参数的方法。
- 账本状态数据操作API：该API提供了对账本数据状态进行操作的方法，包括对状态数据的查询及事务处理等。
- 交易信息获取API：获取提交的交易信息的相关API。
- 对PrivateData操作的API： Hyperledger Fabric在1.2.0版本中新增的对私有数据操作的相关API。
- 其他API：其他的API，包括事件设置、调用其他链代码操作。

2.3.3 链代码示例（1.4 风格）

如下是一个账户转账的链代码示例（1.4风格）仅供安装实例化，若您需要调测请参考**Fabric官方示例**中的链代码。

```
package main

import (
    "fmt"
    "strconv"
    "github.com/hyperledger/fabric/core/chaincode/shim"
    pb "github.com/hyperledger/fabric/protos/peer"
)

type SimpleChaincode struct {
}

// 初始化数据状态，实例化/升级链代码时被自动调用
func (t *SimpleChaincode) Init(stub shim.ChaincodeStubInterface) pb.Response {
    // println函数的输出信息会出现在链代码容器的日志中
    fmt.Println("ex02 Init")

    // 获取用户传递给调用链代码的所需参数
    _, args := stub.GetFunctionAndParameters()

    var A, B string    // 两个账户
    var Aval, Bval int // 两个账户的余额
    var err error

    // 检查合法性, 检查参数数量是否为4个, 如果不是, 则返回错误信息
    if len(args) != 4 {
        return shim.Error("Incorrect number of arguments. Expecting 4")
    }

    A = args[0]  // 账户A用户名
    Aval, err = strconv.Atoi(args[1])  // 账户A余额
    if err != nil {
        return shim.Error("Expecting integer value for asset holding")
    }

    B = args[2]  // 账户B用户名
    Bval, err = strconv.Atoi(args[3])  // 账户B余额
    if err != nil {
        return shim.Error("Expecting integer value for asset holding")
    }
```

```
    fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

    // 将账户A的状态写入账本中
    err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
    if err != nil {
        return shim.Error(err.Error())
    }

    // 将账户B的状态写入账本中
    err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
    if err != nil {
        return shim.Error(err.Error())
    }

    return shim.Success(nil)
}

// 对账本数据进行操作时(query, invoke)被自动调用
func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response {
    fmt.Println("ex02 Invoke")
    // 获取用户传递给调用链代码的函数名称及参数
    function, args := stub.GetFunctionAndParameters()

    // 对获取到的函数名称进行判断
    if function == "invoke" {
        // 调用 invoke 函数实现转账操作
        return t.invoke(stub, args)
    } else if function == "delete" {
        // 调用 delete 函数实现账户注销
        return t.delete(stub, args)
    } else if function == "query" {
        // 调用 query 实现账户查询操作
        return t.query(stub, args)
    }
    // 传递的函数名出错，返回 shim.Error()
    return shim.Error("Invalid invoke function name. Expecting \"invoke\" \"delete\" \"query\"")
}

// 账户间转钱
func (t *SimpleChaincode) invoke(stub shim.ChaincodeStubInterface, args []string) pb.Response {
    var A, B string    // 账户A和B
    var Aval, Bval int // 账户余额
    var X int          // 转账金额
    var err error

    if len(args) != 3 {
        return shim.Error("Incorrect number of arguments. Expecting 3")
    }

    A = args[0]       // 账户A用户名
    B = args[1]       // 账户B用户名

    // 从账本中获取A的余额
    Avalbytes, err := stub.GetState(A)
    if err != nil {
        return shim.Error("Failed to get state")
    }
    if Avalbytes == nil {
        return shim.Error("Entity not found")
    }
    Aval, _ = strconv.Atoi(string(Avalbytes))

    // 从账本中获取B的余额
    Bvalbytes, err := stub.GetState(B)
    if err != nil {
        return shim.Error("Failed to get state")
    }
    if Bvalbytes == nil {
        return shim.Error("Entity not found")
```

```
    }
    Bval, _ = strconv.Atoi(string(Bvalbytes))

    // X为转账金额
    X, err = strconv.Atoi(args[2])
    if err != nil {
        return shim.Error("Invalid transaction amount, expecting a integer value")
    }
    // 转账
    Aval = Aval - X
    Bval = Bval + X
    fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

    // 更新转账后账本中A余额
    err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
    if err != nil {
        return shim.Error(err.Error())
    }

    // 更新转账后账本中B余额
    err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
    if err != nil {
        return shim.Error(err.Error())
    }

    return shim.Success(nil)
}

// 账户注销
func (t *SimpleChaincode) delete(stub shim.ChaincodeStubInterface, args []string) pb.Response {
    if len(args) != 1 {
        return shim.Error("Incorrect number of arguments. Expecting 1")
    }

    A := args[0]   // 账户用户名

    // 从账本中删除该账户状态
    err := stub.DelState(A)
    if err != nil {
        return shim.Error("Failed to delete state")
    }

    return shim.Success(nil)
}

// 账户查询
func (t *SimpleChaincode) query(stub shim.ChaincodeStubInterface, args []string) pb.Response {
    var A string
    var err error

    if len(args) != 1 {
        return shim.Error("Incorrect number of arguments. Expecting name of the person to query")
    }

    A = args[0]   // 账户用户名

    // 从账本中获取该账户余额
    Avalbytes, err := stub.GetState(A)
    if err != nil {
        jsonResp := "{\"Error\":\"Failed to get state for " + A + "\"}"
        return shim.Error(jsonResp)
    }

    if Avalbytes == nil {
        jsonResp := "{\"Error\":\"Nil amount for " + A + "\"}"
        return shim.Error(jsonResp)
    }

    jsonResp := "{\"Name\":\"" + A + "\",\"Amount\":\"" + string(Avalbytes) + "\"}"
```

```
    fmt.Printf("Query Response:%s\n", jsonResp)
    // 返回转账金额
    return shim.Success(Avalbytes)
}

func main() {
    err := shim.Start(new(SimpleChaincode))
    if err != nil {
        fmt.Printf("Error starting Simple chaincode: %s", err)
    }
}
```

2.3.4 链代码示例（2.0 风格）

如下是一个账户转账的链代码示例（2.0风格）仅供安装实例化，若您需要调测请参考**Fabric官方示例**中的链代码。

```
package main

import (
	"errors"
	"fmt"
	"strconv"

	"github.com/hyperledger/fabric-contract-api-go/contractapi"
)

// 链码实现
type ABstore struct {
	contractapi.Contract
}

// 初始化链码数据，实例化或者升级链码时自动调用
func (t *ABstore) Init(ctx contractapi.TransactionContextInterface, A string, Aval int, B string, Bval int) error {
	// 使用println函数输出的信息会记录在链码容器日志中
	fmt.Println("ABstore Init")
	var err error

	fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)
	// 将状态数据写入账本
	err = ctx.GetStub().PutState(A, []byte(strconv.Itoa(Aval)))
	if err != nil {
		return err
	}

	err = ctx.GetStub().PutState(B, []byte(strconv.Itoa(Bval)))
	if err != nil {
		return err
	}

	return nil
}

// A转账X给B
func (t *ABstore) Invoke(ctx contractapi.TransactionContextInterface, A, B string, X int) error {
	var err error
	var Aval int
	var Bval int

	// 从账本获取状态数据
	Avalbytes, err := ctx.GetStub().GetState(A)
	if err != nil {
		return fmt.Errorf("Failed to get state")
	}
	if Avalbytes == nil {
		return fmt.Errorf("Entity not found")
	}
	Aval, _ = strconv.Atoi(string(Avalbytes))
```

```
	Bvalbytes, err := ctx.GetStub().GetState(B)
	if err != nil {
		return fmt.Errorf("Failed to get state")
	}
	if Bvalbytes == nil {
		return fmt.Errorf("Entity not found")
	}
	Bval, _ = strconv.Atoi(string(Bvalbytes))

	// 执行转账
	Aval = Aval - X
	Bval = Bval + X
	fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

	// 将状态数据重新写回账本
	err = ctx.GetStub().PutState(A, []byte(strconv.Itoa(Aval)))
	if err != nil {
		return err
	}

	err = ctx.GetStub().PutState(B, []byte(strconv.Itoa(Bval)))
	if err != nil {
		return err
	}

	return nil
}

// 账户注销
func (t *ABstore) Delete(ctx contractapi.TransactionContextInterface, A string) error {

	// 从账本中删除账户状态
	err := ctx.GetStub().DelState(A)
	if err != nil {
		return fmt.Errorf("Failed to delete state")
	}

	return nil
}

// 账户查询
func (t *ABstore) Query(ctx contractapi.TransactionContextInterface, A string) (string, error) {
	var err error
	// 从账本获取状态数据
	Avalbytes, err := ctx.GetStub().GetState(A)
	if err != nil {
		jsonResp := "{\"Error\":\"Failed to get state for " + A + "\"}"
		return "", errors.New(jsonResp)
	}

	if Avalbytes == nil {
		jsonResp := "{\"Error\":\"Nil amount for " + A + "\"}"
		return "", errors.New(jsonResp)
	}

	jsonResp := "{\"Name\":\"" + A + "\",\"Amount\":\"" + string(Avalbytes) + "\"}"
	fmt.Printf("Query Response:%s\n", jsonResp)
	return string(Avalbytes), nil
}

func main() {
	cc, err := contractapi.NewChaincode(new(ABstore))
	if err != nil {
		panic(err.Error())
	}
	if err := cc.Start(); err != nil {
		fmt.Printf("Error starting ABstore chaincode: %s", err)
	}
}
```

2.3.5 链代码调测

链代码调测主要是使用shim包中的MockStub进行单元测试，所以您需要在测试代码中引入shim包。您可参考下文提供的示例测试代码，也可参考**Fabric官方示例**中的其它测试代码。

编写测试代码

帐户转账链代码示例的测试代码内容如下：

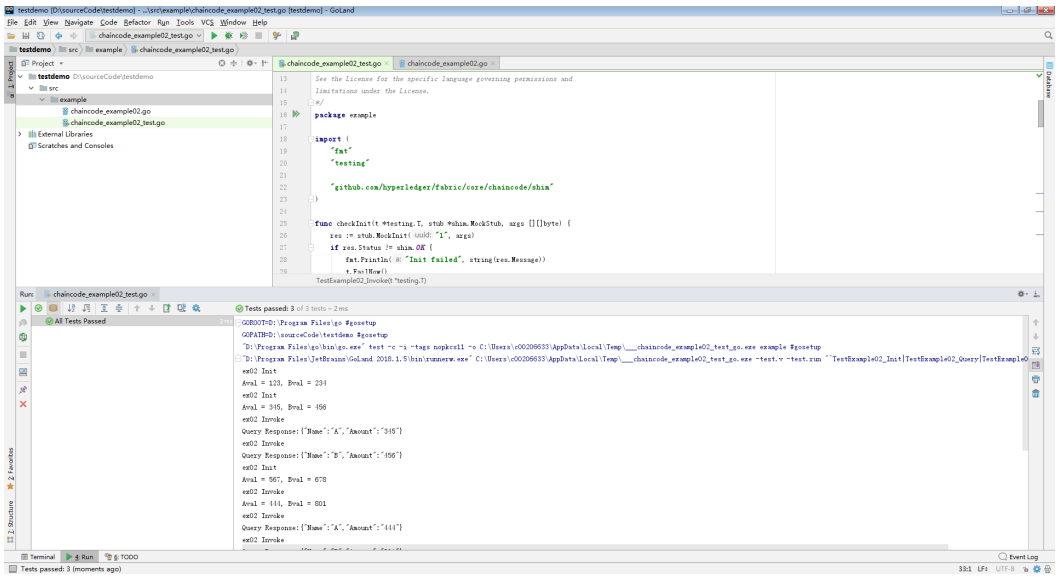
```
package main

import (
    "fmt"
    "testing"

    "github.com/hyperledger/fabric/core/chaincode/shim"
)

func checkInit(t *testing.T, stub *shim.MockStub, args [][]byte) {
    res := stub.MockInit("1", args)
    if res.Status != shim.OK {
        fmt.Println("Init failed", string(res.Message))
        t.FailNow()
    }
}

func checkState(t *testing.T, stub *shim.MockStub, name string, value string) {
    bytes := stub.State[name]
    if bytes == nil {
        fmt.Println("State", name, "failed to get value")
        t.FailNow()
    }
    if string(bytes) != value {
        fmt.Println("State value", name, "was not", value, "as expected")
        t.FailNow()
    }
}

func checkQuery(t *testing.T, stub *shim.MockStub, name string, value string) {
    res := stub.MockInvoke("1", [][]byte{[]byte("query"), []byte(name)})
    if res.Status != shim.OK {
        fmt.Println("Query", name, "failed", string(res.Message))
        t.FailNow()
    }
    if res.Payload == nil {
        fmt.Println("Query", name, "failed to get value")
        t.FailNow()
    }
    if string(res.Payload) != value {
        fmt.Println("Query value", name, "was not", value, "as expected")
        t.FailNow()
    }
}

func checkInvoke(t *testing.T, stub *shim.MockStub, args [][]byte) {
    res := stub.MockInvoke("1", args)
    if res.Status != shim.OK {
        fmt.Println("Invoke", args, "failed", string(res.Message))
        t.FailNow()
    }
}

func TestExample02_Init(t *testing.T) {
    scc := new(SimpleChaincode)
    stub := shim.NewMockStub("ex02", scc)
```

```
    // Init A=123 B=234
    checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"), []byte("123"), []byte("B"), []byte("234")})

    checkState(t, stub, "A", "123")
    checkState(t, stub, "B", "234")
}

func TestExample02_Query(t *testing.T) {
    scc := new(SimpleChaincode)
    stub := shim.NewMockStub("ex02", scc)

    // Init A=345 B=456
    checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"), []byte("345"), []byte("B"), []byte("456")})

    // Query A
    checkQuery(t, stub, "A", "345")

    // Query B
    checkQuery(t, stub, "B", "456")
}

func TestExample02_Invoke(t *testing.T) {
    scc := new(SimpleChaincode)
    stub := shim.NewMockStub("ex02", scc)

    // Init A=567 B=678
    checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"), []byte("567"), []byte("B"), []byte("678")})

    // Invoke A->B for 123
    checkInvoke(t, stub, [][]byte{[]byte("invoke"), []byte("A"), []byte("B"), []byte("123")})
    checkQuery(t, stub, "A", "444")
    checkQuery(t, stub, "B", "801")

    // Invoke B->A for 234
    checkInvoke(t, stub, [][]byte{[]byte("invoke"), []byte("B"), []byte("A"), []byte("234")})
    checkQuery(t, stub, "A", "678")
    checkQuery(t, stub, "B", "567")
    checkQuery(t, stub, "A", "678")
    checkQuery(t, stub, "B", "567")
}
```

执行调测

调测时在IDE中单击执行对应的测试函数即可。

图 2-1 执行调测



2.4 Java 语言链代码开发

2.4.1 链代码结构

本章以Java语言为例来介绍。链代码即一个Java项目，创建好文件后进行函数开发等操作。

约束与限制

Java链代码仅支持Fabric 2.2及以上版本。

链代码接口

链代码启动必须通过调用shim包中的start方法。实际开发中， 您需要自行定义一个类，来继承ChaincodeBase。以下为继承时必须重写的方法：

```
public class SimpleChaincodeSimple extends ChaincodeBase {
    @Override
    public Response init(ChaincodeStub stub) {
    }

    @Override
    public Response invoke(ChaincodeStub stub) {
    }
}
```

- **init方法**：在链代码实例化或升级时被调用，完成初始化数据的工作。
- **Invoke方法**：更新或查询帐本数据状态时被调用， 需要在此方法中实现响应调用或查询的业务逻辑。

链代码结构

Java语言的链代码结构如下：

```
package main
```



```
// 引入必要的包，系统自动操作，只要在maven或gradle中配置即可
import org.hyperledger.fabric.shim.ChaincodeBase;
import org.hyperledger.fabric.shim.ChaincodeStub;

public class SimpleChaincodeSimple extends ChaincodeBase {
    @Override
    public Response init(ChaincodeStub stub) {
        // 在该方法中实现链代码初始化或升级时的处理逻辑
        // 编写时可灵活使用stub中的API
    }

    @Override
    public Response invoke(ChaincodeStub stub) {
        // 在该方法中实现链代码运行中被调用或查询时的处理逻辑
        // 编写时可灵活使用stub中的API
    }

    //主函数，需要调用shim.Start()方法
    public static void main(String[] args) {
        new SimpleChaincode().start(args);
    }
}
```

2.4.2 链代码相关的 API

Fabric源码包中的shim包提供了如下几种类型的接口，您可以参考使用：

- 参数解析API：调用链代码时需要给被调用的目标函数/方法传递参数，该API提供解析这些参数的方法。
- 账本状态数据操作API：该API提供了对账本数据状态进行操作的方法，包括对状态数据的查询及事务处理等。
- 交易信息获取API：获取提交的交易信息的相关API。
- 其他API：其他的API，包括事件设置、调用其他链代码操作。

2.4.3 链代码示例

如下是一个读写数据的链代码示例，您也可以参考**Fabric官方示例**中其他链代码。

2.4.4 链代码调测

对链代码进行调测，主要是使用MockStub进行单元测试。本章中测试的链代码获取方法：登录区块链服务管理控制台，进入“应用案例”，单击“Java示例Demo-Java SDK Demo”中“Chaincode_Java_Local_Demo”的“下载”按钮。

添加依赖

使用mock()方法，需要添加mockito相关依赖。

- gradle版本：

  在build.gradle文件中的dependencies内添加如下配置依赖，注意不是buildscript内的dependencies：

  ```
  testCompile 'org.mockito:mockito-core:2.4.1'
  ```

- maven版本：

  在pom.xml文件中的dependencies（若无则添加）内添加如下配置依赖：

  ```
  <dependency>
      <groupId>org.mockito</groupId>
      <artifactId>mockito-core</artifactId>
  ```

```
    <version>2.4.1</version>
</dependency>
```

编写测试代码

若创建项目时没有test文件夹，在src下新建文件夹，并如图在Gradle Source Sets里面选择“test\java”，然后创建测试文件：SimpleChaincodeTest.java，如图所示：

图 2-2 创建测试文件

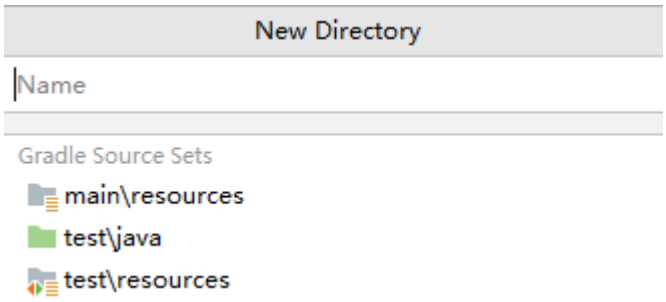
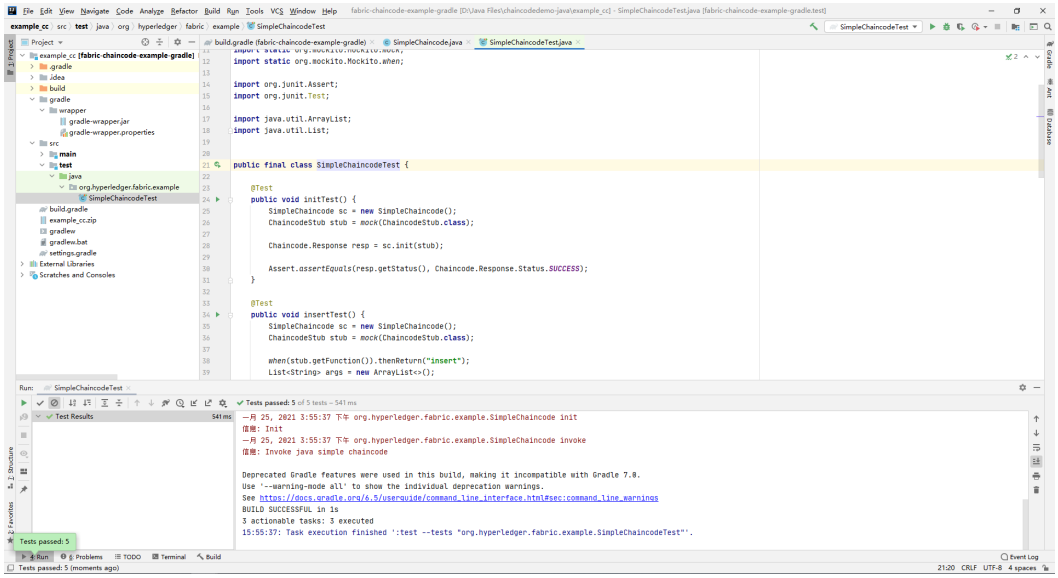


图 2-3 测试文件



SimpleChaincodeTest.java测试代码内容：

```
import org.hyperledger.fabric.example.SimpleChaincode;
import org.hyperledger.fabric.shim.Chaincode;
import org.hyperledger.fabric.shim.ChaincodeStub;
import org.junit.Assert;
import org.junit.Test;

import java.util.ArrayList;
import java.util.List;

import static org.mockito.Mockito.mock;
import static org.mockito.Mockito.when;

public final class SimpleChaincodeTest {

    @Test
    public void initTest() {
```

```
    SimpleChaincode sc = new SimpleChaincode();
    ChaincodeStub stub = mock(ChaincodeStub.class);
    Chaincode.Response resp = sc.init(stub);
    Assert.assertEquals(resp.getStatus(), Chaincode.Response.Status.SUCCESS);
  }

  @Test
  public void insertTest() {
    SimpleChaincode sc = new SimpleChaincode();
    ChaincodeStub stub = mock(ChaincodeStub.class);
    when(stub.getFunction()).thenReturn("insert");
    List<String> args = new ArrayList<>();
    args.add("a");
    args.add("100");
    when(stub.getParameters()).thenReturn(args);
    Chaincode.Response resp = sc.invoke(stub);
    Assert.assertEquals(resp.getStatus(), Chaincode.Response.Status.SUCCESS);
  }

  @Test
  public void insertTooManyArgsTest() {
    SimpleChaincode sc = new SimpleChaincode();
    ChaincodeStub stub = mock(ChaincodeStub.class);
    when(stub.getFunction()).thenReturn("insert");
    List<String> args = new ArrayList<>();
    args.add("a");
    args.add("100");
    args.add("b");
    args.add("100");
    when(stub.getParameters()).thenReturn(args);
    Chaincode.Response resp = sc.invoke(stub);
    Assert.assertEquals(resp.getMessage(), "Incorrect number of arguments. Expecting 2");
  }

  @Test
  public void queryTest() {
    SimpleChaincode sc = new SimpleChaincode();
    ChaincodeStub stub = mock(ChaincodeStub.class);
    when(stub.getFunction()).thenReturn("query");
    List<String> args = new ArrayList<>();
    args.add("a");
    when(stub.getParameters()).thenReturn(args);
    when(stub.getStringState("a")).thenReturn("100");
    Chaincode.Response resp = sc.invoke(stub);
    Assert.assertEquals(resp.getMessage(), "100");
  }

  @Test
  public void queryNoExistTest() {
    SimpleChaincode sc = new SimpleChaincode();
    ChaincodeStub stub = mock(ChaincodeStub.class);
    when(stub.getFunction()).thenReturn("query");
    List<String> args = new ArrayList<>();
    args.add("a");
    when(stub.getParameters()).thenReturn(args);
    when(stub.getStringState("a")).thenReturn(null);
    Chaincode.Response resp = sc.invoke(stub);
    Assert.assertEquals(resp.getMessage(), "{\"Error\":\"Null val for a\"}");
  }
}
```

执行调测

在SimpleChaincodeTest.java中，单击SimpleChaincodeTest方法前的“Run Test”按钮，执行测试。

图 2-4 执行测试

```
16
17  import java.util.ArrayList;
18  import java.util.List;
19
20
21  Run Test  final class SimpleChaincodeTest {
22
23      @Test
24      public void initTest() {
25          SimpleChaincode sc = new SimpleChaincode();
26          ChaincodeStub stub = mock(ChaincodeStub.class);
27
28          Chaincode.Response resp = sc.init(stub);
29
30          Assert.assertEquals(resp.getStatus(), Chaincode.Response.Status.SUCCESS);
31      }
```

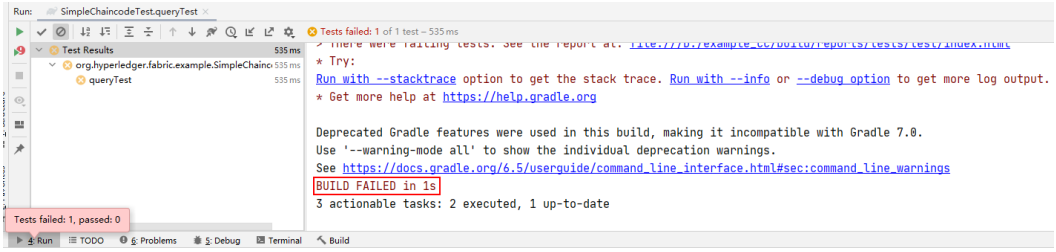
执行成功如图所示，表示链代码调测无问题：

图 2-5 执行成功

```
Deprecated Gradle features were used in this build, making it incompatible with Gradle 7.0.
Use '--warning-mode all' to show the individual deprecation warnings.
See https://docs.gradle.org/6.5/userguide/command_line_interface.html#sec:command_line_warnings
BUILD SUCCESSFUL in 1s
3 actionable tasks: 1 executed, 2 up-to-date
16:12:23: Task execution finished ':test --tests "org.hyperledger.fabric.example.SimpleChaincodeTest"'.
```

执行失败示例如图所示，请根据失败提示修改链代码或者检查调测代码的逻辑：

图 2-6 执行失败




```
expected:<[{"Error":"Null val for a"}]> but was:<[100]>
Expected :{"Error":"Null val for a"}
Actual   :100
<Click to see difference>
```

3 应用程序开发

3.1 概述

用户的应用程序通过链代码与账本数据进行交互。应用程序开发可使用的语言比较广泛，如Golang、Solidity、Java、C++、Python、Node.js等。应用程序和链代码开发语言无强对应关系，只要应用程序能通过SDK来调用链代码即可。

说明

Hyperledger Fabric增强版对应用程序开放的接口均为gRPC协议，与开源版本保持一致，通常使用SDK进行调用，详情可参考**Hyperledger Fabric增强版SDK接口定义**。

3.2 开发前准备

用户的应用程序通过链代码与账本数据进行交互。应用程序开发可使用的语言比较广泛，如Golang、Solidity、Java、C++、Python、Node.js等。应用程序和链代码开发语言无强对应关系，只要应用程序能通过SDK来调用链代码即可。

1. 您需要创建区块链实例。
   区块链实例支持在CCE集群上部署，具体可参见**基于CCE集群**。
2. 您需要获取对应实例的SDK配置文件，具体可参见**访问区块链**。

3.3 应用程序开发

您需要自行开发应用程序业务逻辑代码。应用开发过程中可使用BCS提供的**国密加密SDK**，也可使用Fabric官方社区提供的和您自身的实例版本匹配的SDK。

说明

Fabric源码包选择和创建的区块链实例版本对应，即如果创建区块链实例时，Hyperledger Fabric增强版内核是v2.2（4.X.X版本），则Fabric源码包对应选择2.2版本。

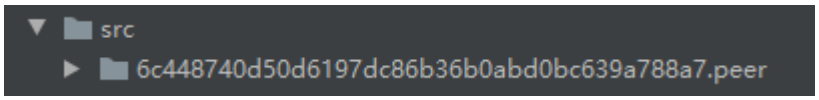
配置组织 ID

您需要修改应用程序中配置实例组织ID的相关代码，下载证书文件解压后的peer文件包括目录名和对应组织ID。

如下图所示，仅供示例参考，请以实际操作的证书文件为准。

证书文件解压后目录名是6c448740d50d6197dc86b36b0abd0bc639a788a7.peer，组织ID为6c448740d50d6197dc86b36b0abd0bc639a788a7。

图 3-1 证书文件解压



配置 SDK 文件

1. 您需要修改应用程序中SDK配置文件相关代码，如下面示例所示，您需要填写正确的SDK配置文件绝对路径。

```
var (
   configFile = "/root/gosdkdemo/config/go-sdk-demo-channel-sdk-config.yaml"
   org = " 6c448740d50d6197dc86b36b0abd0bc639a788a7"
)
```

2. 如果您下载SDK配置文件时填写的证书存放路径与实际不符，您需要修改SDK配置文件中所有证书相关路径。

4 示例 Demo

4.1 GO SDK Demo

本节提供了一个基于Go SDK的Demo，帮助用户开发自己的Go客户端应用程序。

准备工作

- 准备弹性云服务器。
- 在弹性云服务器上安装golang环境，Go版本要求：1.12及以上，1.16以下（>=1.12，<1.16）。
- 获取Go SDK源码，获取方法：登录区块链服务管理控制台，进入“体验中心 > 应用案例”，单击“GO示例Demo-GO SDK Demo”中Go应用程序源码的“下载”按钮。

创建区块链实例

创建区块链实例，具体请参见**基于CCE集群**。

安装及实例化链代码

本示例使用链代码文件获取方法：登录区块链服务管理控制台，进入“体验中心 > 应用案例”，单击“GO示例Demo-GO SDK Demo”中Go语言示例链代码的“下载”按钮。

参考章节：**用户指南-区块链管理-链代码管理**。

下载 SDK 和证书

步骤1 登录区块链服务管理控制台。

步骤2 在“实例管理”页面，在实例卡片中，单击“获取客户端配置”。

步骤3 在新打开的页面，勾选“SDK文件”，SDK配置参数如下：

参数	值
链代码名称	chaincode **说明** 链代码名称需要和链代码安装&实例化时的一致。
证书存放路径	/root/gosdkdemo/config
通道名称	channel
组织&Peer节点	保持系统默认

勾选“共识节点证书”。

勾选“Peer节点证书”，指定节点组织选择organization，勾选“管理员证书”。

步骤4 单击“下载”。下载SDK配置文件、orderer组织的管理员证书和organization组织的管理员证书。

----结束

部署应用

1. 将Go SDK源码下载至准备的弹性云服务器“/root”路径下并解压。

   下载方法：登录区块链服务管理控制台，进入“应用案例”，单击“GO示例Demo-GO SDK Demo”中Go应用程序源码的“下载”按钮。

2. 将**下载SDK和证书**步骤中的zip文件解压后，把configs文件夹中的orderer文件夹、peer文件夹、sdk-config.json、sdk-config.yaml文件全部复制到/root/gosdkdemo/config/目录下。

3. 在代码中找到“/gosdkdemo/src/main.go”文件，进行以下修改：

   a. 将configFile中的值修改为实际的SDK配置文件名称，例如：demo-channel-sdk-config.yaml。

   b. 将org的值修改为organization对应的组织哈希值。

   在通道管理页面，单击“查看节点”获取组织的哈希值（MSP标识去掉“MSP”后缀即为对应组织的哈希）。

   ```
   var (
       configFile = "/root/gosdkdemo/config/go-sdk-demo-channel-sdk-config.yaml"
       org = " 9103f17cb6b4f69d75982eb48bececcc51aa3125"
   )
   ```

4. 使用go mod方式配置GOPATH路径，请根据实际安装路径进行配置。

   a. 设置环境变量GO111MODULE为on。

   ```
   export GO111MODULE=on
   ```

   b. go.mod文件如图所示，用户需要根据实际安装路径修改replace代码。

   ```
   module main
   go 1.15
   // 指定导入的依赖包及其版本
   require (
           github.com/bitly/go-simplejson v0.5.0
           github.com/bmizerany/assert v0.0.0-20160611221934-b7ed37b82869// indirect
           github.com/ghodss/yaml v1.0.0
           github.com/hyperledger/fabric-sdk-go v1.0.0
           github.com/pkg/errors v0.9.1
           github.com/spf13/viper v1.7.1
   )
   ```

```
// 以项目路径为/root/gosdkdemo/src为例
replace github.com/hyperledger/fabric-sdk-go => /root/gosdkdemo/src/github.com/hyperledger/fabric-sdk-go
```

5. 找到“gosdkdemo/src”路径下的main.go文件，执行如下命令：

```
go run main.go
```

常用 API 接口

Fabric-sdk-go的主要入口是FabricSDK类，这个可以通过NewSDK()方法分别可以生成。

Fabric-sdk-go的常用操作基本都可以用这四个client实现：FabricClient，ChannelClient，ChannelMgmtClient，ResourceMgmtClient。

- **FabricSDK**

  FabricSDK在pkg\fabsdk\fabsdk.go中，通过New ()方法生成object。New ()方法支持可变参数Option，以下是生成FabricSDK的例子:

  var opts []fabsdk.Option

  opts = append(opts, fabsdk.WithOrgid(org))

  opts = append(opts, fabsdk.WithUserName("Admin"))

  sdk, err = fabsdk.New(config.FromFile(configFile), opts...)

  configFile是SDK配置文件的路径。OrgId是SDK配置文件中的组织id。

  FabricSDK在def/fabapi/fabapi.go中，通过NewSDK()方法生成object。NewSDK()方法有一个Options参数，以下是生成Options参数的例子:

  ```
  deffab.Options{ConfigFile: configFile, LoggerFa
  logging.LoggerProvider(), UserName: sysadmin}
  ```

  ConfigFile是SDK配置文件的路径。LoggerFactory是可选的，不提供的话default会log到console。

- **FabricClient**

  FabricClient主要有以下常用的接口。

接口名称	描述	参数值	返回值
CreateChannel	创建Channel的接口，用于创建channel。	request CreateChannelRequest	txn.TransactionID, error
QueryChannelInfo	查询Channel的接口，用于查询Channel的信息。	name string, peers []Peer	Channel, error

接口名称	描述	参数值	返回值
InstallChaincode	安装链码的接口，安装链码到区块链中。	request InstallChaincodeRequest	[]*txn.TransactionProposlResponse, string, error
InstallChaincode	安装链码的接口，安装链码到区块链中。	request InstallChaincodeRequest	[]*txn.TransactionProposlResponse, string, error
QueryChannels	查询channel的接口，查询区块链中已创建的通道。	peer Peer	*pb.ChannelQueryResponse, error
QueryInstalledChaincodes	查询已安装链码的接口，查询区块链中已安装的链码。	peer Peer	*pb.ChaincodeQueryResponse, error

- **ChannelClient**

  ChannelClient主要包括链码查询和链码调用两类接口。

接口名称	描述	参数值	返回值
Query	链码查询接口，调用链码进行查询。	request QueryRequest	[]byte, error
QueryWithOpts	带options的链码查询接口，与Query类似，但是可以通过QueryOpts指定notifier, peers,和timeout。	request QueryRequest, opt QueryOpts	[]byte, error
ExecuteTx	链码调用接口，用于链码的调用。	request ExecuteTxRequest	TransactionID, error
ExecuteTxWithOpts	带options的链码调用接口，与ExecuteTx类似，但是可以通过ExecuteTxOpts指定notifier, peers,和timeout。	request ExecuteTxRequestopt ExecuteTxOpts	TransactionID, error

- **ChannelMgmtCLient**

  ChannelMgmtClient 只有两个接口SaveChannel(req SaveChannelRequest) error 和SaveChannelWithOpts(req SaveChannelRequest, opts

SaveChannelOpts) error 这两个接口是用于创建channel用的，这两个接口里面具体实现会调用到FabricClient里createChannel()接口。

- **ResourceMgmtClient**

  ResourceMgmtClient主要就是与链码生命周期相关的接口和一个peer加入通道的接口。

  说明

  链码的删除接口为BCS增加的接口，目前只实现了删除链码安装包的功能。

接口名称	描述	参数值	返回值
InstallCC	安装链码，用于安装链码。	reqInstallCCRequest	[]InstallCCResponse, error
InstallCCWithOpts	带options的链码安装，与InstallCC类似，但是可以通过InstallCCOpts指定peers。	reqInstallCCRequest,opts InstallCCOpts	[]InstallCCResponse, error
InstantiateCC	实例化链码接口，用于实例化链码。	channelID string,reqInstantiateCCRequest	error
InstantiateCCWithOpts	带options的链码实例化，与InstantiateCC类似，但是可以通过InstantiateCCOpts指定peers和timeout。	channelID string,reqInstantiateCCRequest, optsInstantiateCCOpts	error
UpgradeCC	升级链码，用于链码的升级。	channelID string,reqUpgradeCCRequest	error
UpgradeCCWithOpts	带options的链码升级，与UpgradeCC类似，但是可以通过UpgradeCCOpts指定peers和timeout。	channelID string,reqUpgradeCCRequest, optsUpgradeCCOpts	error
DeleteCC	删除链码，用于链码的删除，目前只有删除安装包的功能。	channelID string,reqDeleteCCRequest	error
DeleteCCWithOpts	带options的链码删除，与DeleteCC类似，但是可以通过DeleteCCWithOpts指定peers和timeout。	channelID string,reqDeleteCCRequest,opts DeleteCCOpts	error

接口名称	描述	参数值	返回值
JoinChannel	Peers加入Channel的接口，用于peers加入Channel。	channelID string	error
JoinChannelWithOpts	带options的Peers加入Channel的接口，与JoinChannel类似，但是可以通过JoinChannelOpts指定peers。	channelID string,optsJoinChannelOpts	error

说明

带options的接口都可以指定peers，peers可通过def/fabapi/pkgfactory.go 里的NewPeer(userName string, orgName string, url string, certificate string, serverHostOverride string, config config.Config) (fab.Peer, error) 生成。这个method比原生的NewPeer多两个参数userName, orgName, 这两个参数用于peer双向tls找到对应的tls证书。

调用合约

Main.go是一个简单的客户端应用示例程序，主要是为了方便用户熟悉客户端开发的流程，主要包含以下步骤：

```
//1.导入相关包：Sdk包中提供了一些API，以便用户的应用程序能够访问链代码。
import (
   "fmt"
   "github.com/hyperledger/fabric-sdk-go/pkg/client/channel"
   "github.com/hyperledger/fabric-sdk-go/pkg/fabsdk"    ……
)
//2.创建文件配置：这部分封装了应用开发必要的一些公共配置，包括sdk配置文件路径、组织名
var (
   configFile = "/root/fabric-go-demo/config/go-sdk-demo-channel-sdk-config.yaml"
   org = "9103f17cb6b4f69d75982eb48bececcc51aa3125"
   ……
)
//3.加载配置文件
loadConfig()
//4. 初始化sdk
initializeSdk()
//5. 执行链代码，将数据写入账本，key = "testuser"，value= "100"
insert("insert",[][]byte{
   []byte("testuser"),
   []byte("100"),
})
//6.查询链代码，输出查询结果，key = "testuser"
query("query",
[][]byte{
[]byte("testuser"),
})
```

表 4-1 调用函数介绍

函数名	说明
getOptsToInitializeSDK	解析配置文件，创建并返回fabsdk.Option对象。
GetDefaultChaincodeId	解析配置文件，返回chaincodeID。
GetDefaultChannel	解析配置文件，返回channelID。
UserIdentityWithOrgAndName	用户身份验证，输入为组织名和用户名，返回为验证结果。
ChannelClient	创建*channel.Client对象，输入为组织名、用户名以及通道ID，返回*channel.Client对象。
insert	将数据写入账本，输入参数为链码的对应方法名称以及要插入的键值对，返回为写入的结果。
query	查询链上信息，输入参数为链码的对应方法名称以及要查询的数据，返回为查询的结果。

4.2 Java SDK Demo

以上章节是对链代码的开发、调测及其应用配置的介绍，本章节通过提供一个基于Java SDK的Demo（支持国密加密算法），以Java SDK Demo为例，让您了解区块链的相关知识以及体验应用开发的流程，助您快速上手并使用区块链服务。

说明

只用于场景体验，不用于实际应用。

准备工作

操作指导	说明
安装开发工具（IDE）	本地需提前安装JDK、maven和eclipse（可以使用您习惯的IDE）。 JDK需要安装1.8版本64位，如果本地已安装JDK，可以通过cmd命令java -version来查看JDK版本。

创建区块链实例

步骤1 登录区块链服务管理控制台。

步骤2 单击页面右上角的“创建区块链实例”。

步骤3 根据界面提示，配置区块链基本信息，参数如表4-2所示。

须知

为了保证示例Demo成功运行，请在参数配置时按照表格中的参数值填写。

表 4-2 基本信息配置

参数	参数值
区域	使用默认区域
企业项目	选择已创建的企业项目，例如：default。 如果您没有开通企业管理服务，将无法看到企业项目选项。
区块链实例名称	java-sdk-demo
版本类型	专业版（若安全机制需要使用国密算法，请选择企业版）
区块链类型	私有链
Hyperledger Fabric增强版内核	v2.2
共识策略	Raft(CFT)
资源初始密码	请自行设置
资源初始密码确认	-

步骤4 单击“下一步：资源配置”，进行资源配置，参数如表4-3所示。

表 4-3 资源配置

参数	示例
环境资源	选择“自定义环境”。
集群	创建新的CCE集群
可用区	请自行选择
多可用区	否
是否使用CCE集群节点弹性IP	是

步骤5 单击“下一步：区块链配置”，进行区块链配置，参数如表4-4所示。

表 4-4 区块链配置

参数	示例
区块链配置	选择“自定义配置”。

参数	示例
区块链管理初始密码	如果填写该项，则以填写值为准，如果不填写，则以资源初始密码为准。
区块链管理确认密码	-
存储卷类型	极速文件存储卷，也可以根据界面提示配置。
节点组织存储容量(GB)	使用默认规格。
账本数据存储方式	选择“文件数据库（ GoLevelDB ）”
peer节点组织	系统已默认创建1个节点组织，名称为organization，将节点数量修改为1。
通道配置	organization节点组织已默认添加进至通道中，保持默认即可。
共识节点数量	使用默认。
安全机制	ECDSA **须知** 安全机制支持选择国密算法。若选择了国密算法，则Demo的部署需要做其他修改，请注意下文中的相关描述。
区块生成配置	否
添加RESTful API支持	否

步骤6 单击“下一步：确认规格”。

步骤7 确认配置信息无误后，根据界面提示创建区块链实例。

请等待数分钟，安装页面提示安装成功，查看实例状态变为“正常”后，表示区块链实例部署完成。

----结束

安装及实例化链代码

步骤1 登录区块链服务管理控制台。

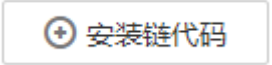
步骤2 单击左侧导航栏中的“实例管理”。

步骤3 在新创建的实例卡片中，单击“区块链管理”，登录链代码管理页面。

步骤4 在登录页面输入用户名、密码，单击“登录”。

说明

用户名为admin，密码为您在创建区块链实例时设置的区块链管理初始密码，如果没有设置区块链管理初始密码，则以资源初始密码为准。

步骤5 在链代码管理页面，单击页面左上角的 

。

安装参数如下：

表 4-5 安装参数

参数	值
链代码名称	chaincode
链代码版本	2.0
账本数据存储方式	文件数据库(goleveldb)
选择全部Peer节点	勾选
组织&Peer节点	peer-0
链代码语言	Golang
链代码文件	Chaincode_Go_Demo，获取方法：登录区块链服务管理控制台，进入“应用案例”，单击“Java示例Demo-Java SDK Demo”中Go语言示例链代码的“下载”按钮。
链代码描述	根据需要填写相关描述。
代码安全检查	链代码语言选择Golang，该功能才会显示。选择是否开启链代码安全检查。

步骤6 单击“安装”完成链代码安装。

步骤7 链代码安装完成后，在链代码列表的“操作”列，单击“实例化”。

实例化参数如下：

表 4-6 实例化参数

参数	值
链代码名称	chaincode
实例化通道	channel
链代码版本	2.0
初始化函数	init
链代码参数	a,200,b,250
背书策略	下列任意组织背书
背书组织列表	organization
隐私保护配置	否

步骤8 单击“实例化”，完成链代码在当前通道上的实例化。

稍等2-3分钟后刷新界面，单击“实例化”列的“查看更多”，查看链代码实例化进度。

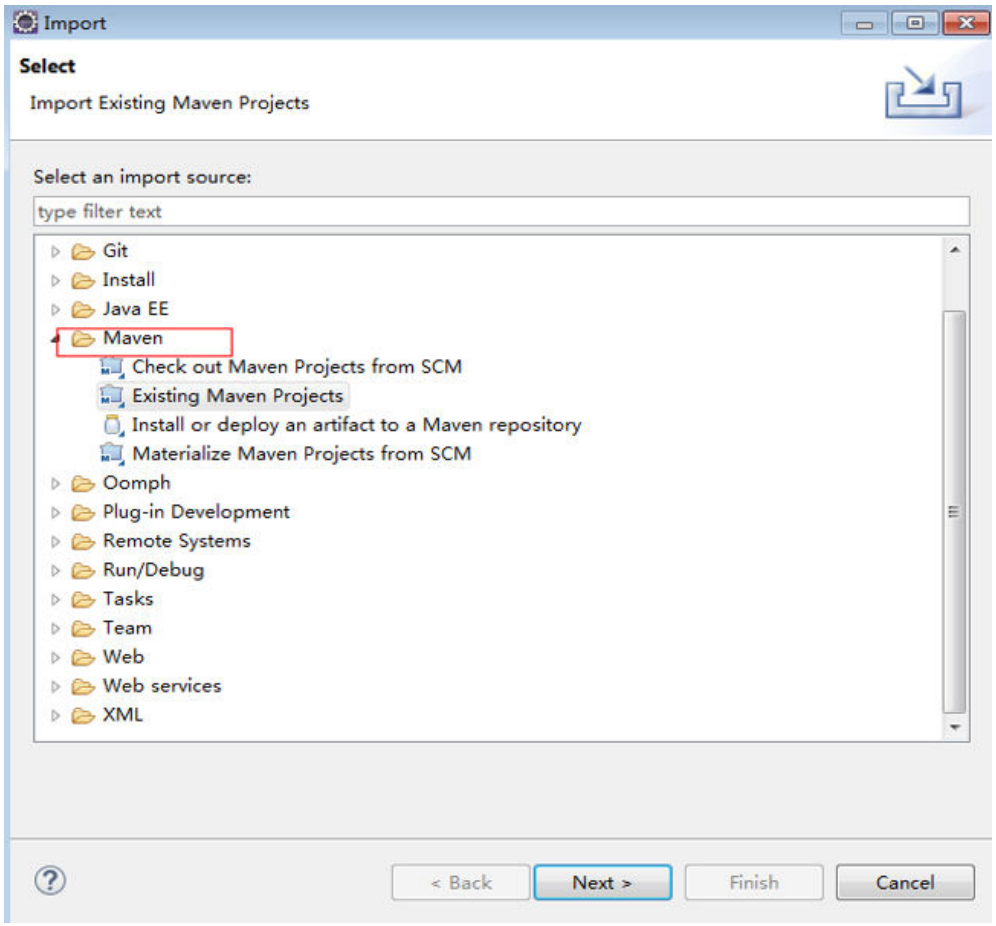
----结束

配置应用

步骤1 导入项目工程

获取项目代码并解压，获取方法：登录区块链服务管理控制台，进入“应用案例”，单击“Java示例Demo-Java SDK Demo”中Java项目源码的“下载”按钮。

在eclipse界面中右键选择import，将项目代码导入eclipse中（javasdkdemo为Maven工程）。



步骤2 下载SDK配置文件和证书。

1. 在“实例管理”界面，在实例卡片中，单击“获取客户端配置”。
2. 勾选“SDK文件”，SDK配置参数如下：

表 4-7 SDK 配置参数

参数名称	说明
链代码名称	chaincode **须知** 链代码名称需要和链代码安装&实例化时的一致。
证书存放路径	此处配置为下载的javasdkdemo工程的config文件夹路径。 **须知** 将windows的反斜杠“\”换为“/”。例如，正确的写法为D:/javasdkdemo/config
通道名称	channel
组织&Peer节点	选择通道中所有节点组织

勾选“共识节点证书”。

勾选“Peer节点证书”，指定节点组织选择organization，勾选“管理员证书”。

3. 单击“下载”，下载SDK配置文件、java-sdk-demo-orderer组织的管理员证书、organization组织的管理员证书，下载至demo工程的config路径中。

步骤3 复制并解压。

步骤4 将下载的demo-config.zip文件解压，将java-sdk-demo-orderer-admin-cert、organization-admin-cert和sdk-config文件夹中的内容复制至demo工程的config路径下。

----结束

部署应用

步骤1 在Maven工程中找到/javasdkdemo/src/main/java/handler/**Main.java**，将Main类中如下代码中的文件路径，修改为java-sdk-demo-sdk-config.yaml文件的绝对路径，路径请参见**步骤2.3**。需将windows的反斜杠**“\” 改为 “/”。**

```
helper.setConfigCtx("E:/yourdir/.yaml");
例如修改为：helper.setConfigCtx("D:/javasdkdemo/config/java-sdk-demo-channel-sdk-config.yaml");
```

须知

若您在订购区块链实例时安全机制选择“国密算法”，还需要做如下修改：国密算法需要引用自己的fabric-sdk依赖包，故需要在pom.xml中添加对工程lib文件夹“fabric-sdk-java-1.4.1-jar-with-dependencies.jar”中的依赖，如下图所示去掉该依赖的注释即可，否则可能会导致运行错误。

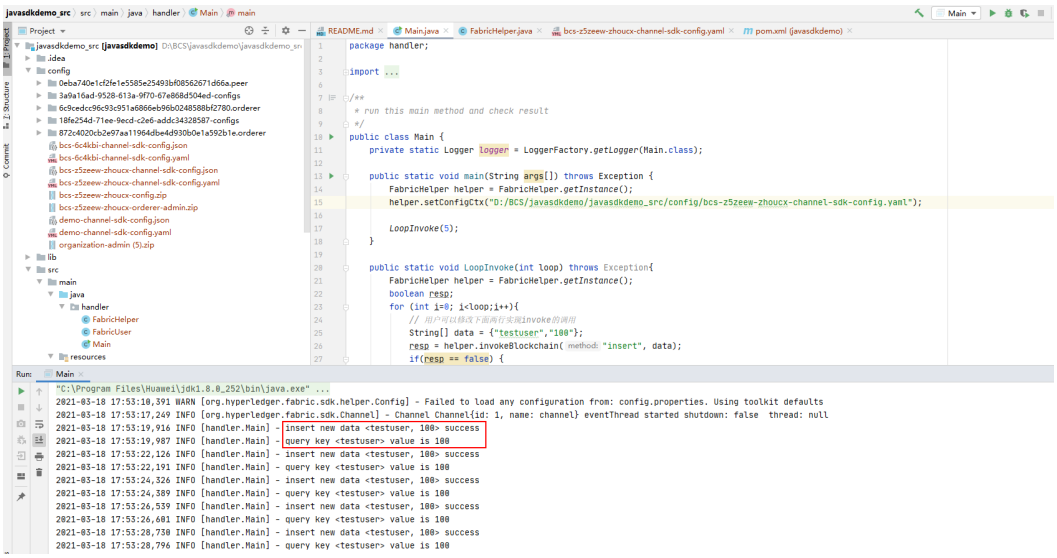
图 4-1 文件详情

```
        </dependency>
        <dependency>
            <groupId>org.hyperledger.fabric-sdk-java</groupId>
            <artifactId>fabric-sdk-java</artifactId>
            <version>1.4.1 </version>
            <!--
            <scope>system</scope>
            <systemPath>${project.basedir}/lib/fabric-sdk-java-1.4.1 -jar-with-dependencies.jar</systemPath>
            -->
        </dependency>
        <dependency>
```

步骤2 运行main函数。

每成功执行一次，表示向区块链存入一对键值对，<testuser,100>；在区块链上查询键值为testuser的value值为100。也可以通过区块链浏览器可以查看交易记录。

图 4-2 交易记录信息



----结束

4.3 Gateway Java Demo

本节提供一个基于Fabric Gateway Java的Demo，Fabric Gateway Java对Java SDK进行了封装，简化了代码量，帮助用户开发自己的Java客户端应用程序。

常用接口

使用Fabric-Gateway-Java发起交易和查询，主要用到Network和Contract两类的接口，**更多的api接口请参考Fabric官网**。

- **Network**

  主要有以下常用的接口：

接口名称	描述	参数值	返回值
getContract	获取Contract实例的接口	String chaincodeId	Contract
addBlockListener	设置监听器的接口，监听区块事件	Consumer<org.hyperledger.fabric.sdk.BlockEvent> listener	Consumer<org.hyperledger.fabric.sdk.BlockEvent>
getChannel	获取与network相关联channel的接口	/	org.hyperledger.fabric.sdk.Channel
removeBlockListener	移除监听器的接口	Consumer<org.hyperledger.fabric.sdk.BlockEvent> listener	void

- **Contract**

  主要有以下常用的接口：

接口名称	描述	参数值	返回值
submitTransaction	发起交易的接口，需要输入调用方法与参数	String name, String... args	byte[]
evaluateTransaction	发起查询的接口，需要输入调用方法与参数	String name, String... args	byte[]
createTransaction	创建交易的接口，需要submit以发起交易	String name	Transaction
addContractListener	设置监听器的接口，监听已经提交的交易发出的事件	Consumer<ContractEvent> listener	Consumer<ContractEvent> listener
removeContractListener	移除监听器的接口	Consumer<ContractEvent> listener	void

4.4 REST API Demo

区块链提供了REST API服务来简化用户访问区块链的学习成本。通过REST API服务，用户可以不需要学习fabric-go-sdk，fabric-Java-sdk, fabric-nodejs-sdk等，只需要开

发的应用支持RESTful接口，就可以轻松访问区块链。本Demo通过一个go语言的客户端来演示如何使用REST API服务调用链代码，供您学习参考。

说明

只用于场景体验，不用于实际应用。

创建区块链实例

步骤1 登录区块链服务管理控制台。

步骤2 单击页面右上角的“创建区块链实例”。

步骤3 根据界面提示，配置区块链基本信息，参数如表4-8所示。

须知

为了保证示例Demo成功运行，请在参数配置时按照表格中的参数值填写。

表 4-8 基本信息配置

参数	参数值
区域	使用默认区域
企业项目	default
区块链实例名称	demo
版本类型	专业版
区块链类型	私有链
Hyperledger Fabric增强版内核	v2.2
共识策略	Raft(CFT)
资源初始密码	请自行设置
资源初始密码确认	请自行设置

步骤4 单击“下一步：资源配置”，进行资源配置，参数如表4-9所示。

表 4-9 资源配置

参数	示例
环境资源	选择“自定义环境”。
集群	创建新的CCE集群
可用区	请自行选择
云主机规格	4核/8GB

参数	示例
云主机个数	1
高可用	否
虚拟私有云	系统自动创建VPC
所在子网	系统自动创建子网
云主机登录方式	密码
root密码	如果填写该项，则以填写值为准，如果不填写，则以资源初始密码为准。
确认密码	-
是否使用CCE集群节点弹性IP	是
弹性IP计费方式	使用默认规格
弹性IP带宽	5 Mbit/s

步骤5 单击“下一步：区块链配置”，进行区块链配置，参数如表4-10所示。

表 4-10 区块链配置

参数	示例
区块链配置	选择“自定义配置”。
区块链管理初始密码	如果填写该项，则以填写值为准，如果不填写，则以资源初始密码为准。
区块链管理确认密码	-
存储卷类型	极速文件存储卷
节点组织存储容量 (GB)	使用默认规格。
账本数据存储方式	选择“文件数据库（GoLevelDB）”
peer节点组织	系统已默认创建1个节点组织，名称为organization，将节点数量修改为1。
通道配置	organization节点组织已默认添加进至通道中，保持默认即可。
共识节点数量	使用默认值。
安全机制	ECDSA **须知** 安全机制仅支持选择ECDSA。
区块生成配置	否

参数	示例
添加RESTful API支持	请选择“是”。若您选择了“否”，则需要在实例创建完毕后，执行以下步骤安装RESTful API： 1. 左侧导航栏选择“插件管理”。 2. 在“插件仓库”页签下，鼠标移至baas-restapi插件卡片右上角。 3. 单击“安装”，选择已创建的区块链实例，安装baas-restapi插件。

步骤6 单击“下一步：确认订单”。

步骤7 确认配置信息无误后，根据界面提示创建区块链实例。

请等待数分钟，安装页面提示安装成功，查看实例状态变为“正常”后，表示区块链实例部署完成。

----结束

安装及实例化链代码

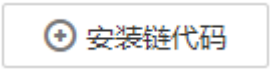
步骤1 登录区块链服务管理控制台。

步骤2 在新创建的实例卡片中，单击“区块链管理”，登录链代码管理页面。

步骤3 在登录页面输入用户名、密码，单击“登录”。

说明

用户名为admin，密码为您在创建区块链实例时设置的区块链管理初始密码，如果没有设置区块链管理初始密码，则以资源初始密码为准。

步骤4 在链代码管理页面，单击页面左上角的 

。

安装参数如下：

参数	值
链代码名称	bcsysq
链代码版本	1.0
账本数据存储方式	文件数据库(goleveldb)
选择全部Peer节点	勾选
组织&Peer节点	peer-0
链代码语言	Golang
链代码文件	下载示例链代码文件：**chaincode_example02.zip**。
链代码描述	根据需要填写相关描述。

参数	值
代码安全检查	链代码语言选择Golang，该功能才会显示。选择是否开启链代码安全检查。

步骤5 单击“安装”完成链代码安装。

步骤6 链代码安装完成后，在链代码列表的“操作”列，单击“实例化”。

实例化参数如下：

参数	值
实例化通道	channel
链代码版本	1.0
初始化函数	init
链代码参数	a,200,b,250
背书策略	任意组织背书
背书组织列表	organization
隐私保护配置	否

----结束

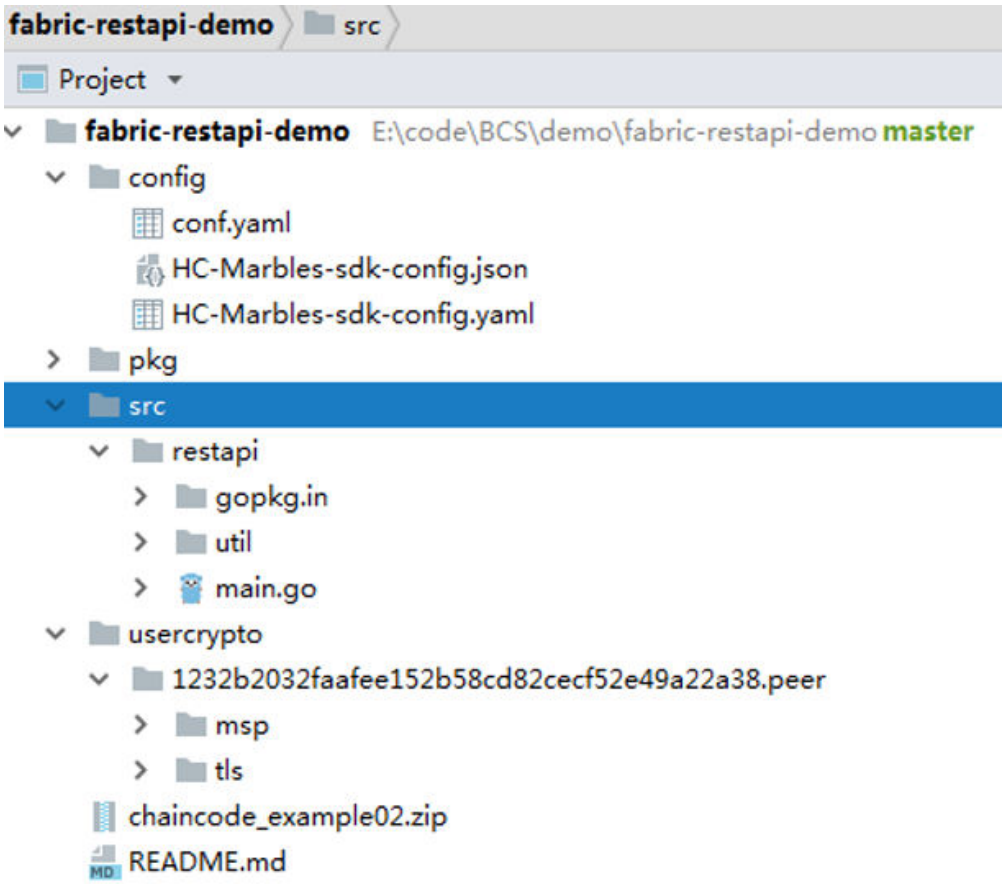
配置应用

步骤1 在“实例管理”界面，在实例卡片中，单击“获取客户端配置”。

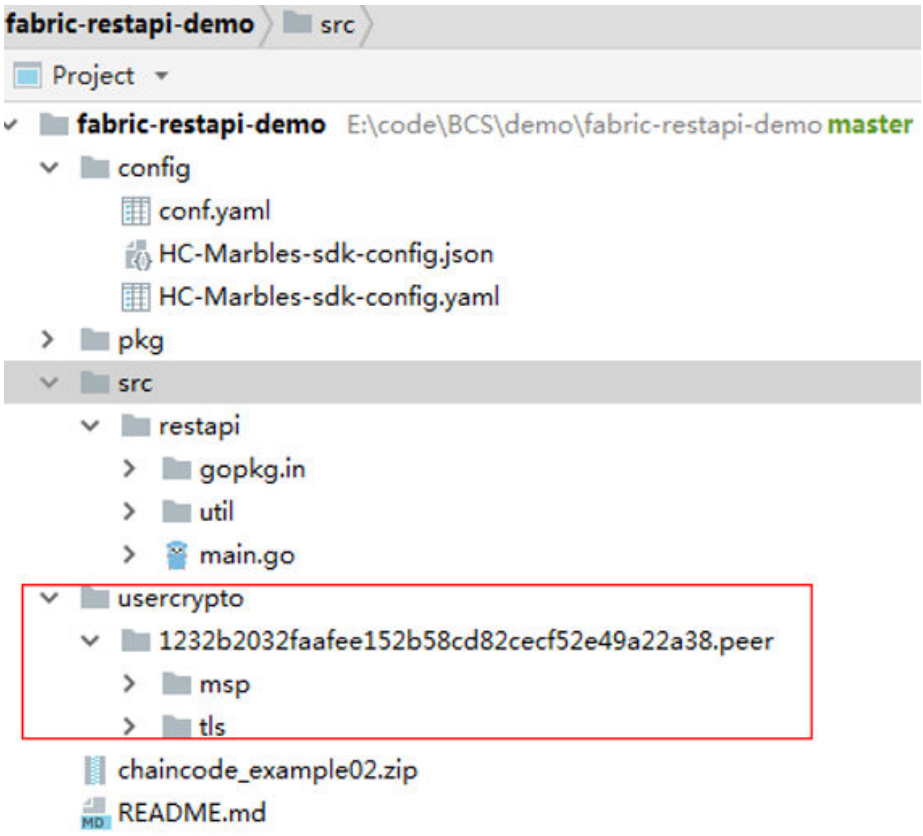
步骤2 勾选“Peer节点证书”，指定节点组织选择organization，勾选“用户证书”并下载。

步骤3 下载Demo项目工程：**fabric-restapi-demo.zip**，并将Demo项目代码工程包下载解压到本地并使用IDE打开。

本Demo是一个golang编写的REST客户端，通过RESTAPI服务来调用链代码，实现链代码a给b转账的功能，请用GoLand等个人喜欢的IDE打开。项目的内容如下图所示：



步骤4 将下载的用户证书解压到项目的usercrypto目录下。如图所示：



步骤5 修改参数配置。

1. 修改config目录下conf.yaml中的各项参数，参考如下截图及表格。

2. 修改src/restapi目录下的main.go文件，参考如下截图及表格。

说明

针对需要参与背书的每个peer节点，对其构造一个OrgPeer结构体，将组织ID和peer节点的域名传入，并将该结构体添加进OrgPeer类型的数组中，经json.Marshal()方法转换为字节数组，最后会转换成字符串类型传入。其中OrgPeer结构体定义如下：

```
type OrgPeer struct {
OrgId string `json:"orgId"`
PeerDomainName string `json:"peerDomainName"`
}
```

表 4-11 参数表

参数	说明
Endpoint	RESTAPI服务的访问IP和端口。具体获取方法如下： 1. 在已创建区块链实例卡片中，单击“容器集群”名称，进入云容器引擎CCE页面。 2. 在左侧导航栏，选择“资源管理 > 节点管理”。 3. 在对应实例的IP地址列，获取弹性公网IP地址。端口固定为32621。
Path	RESTAPI服务的访问路径，保持不变。
CryptoMethod	加密算法，如果是ECDSA算法填写“SW”。
SignCert	用户下载的证书中的签名证书路径。
PrvKey	用户下载证书中的签名私钥。
InvokeReq	请求body的参数，请按照部署的链码实际情况填写，可参考如下InvokeReq参数表。
QueryReq	与InvokeReq类似，按照部署链码的实际情况填写。

表 4-12 InvokeReq 参数表

参数	说明	范例
SignGzipSwitch	Sign是否选择Gzip压缩。0表示否，1表示是。	"1"
ChannelId	区块链通道名称。	"channel"
ChaincodeId	链代码名称。	"testcode"
Chaincode Version	链代码版本。	"1.0"
UserId	由组织CA签发的用户ID，目前区块链实例默认生成的都为User1。	"User1"

参数	说明	范例
OrgId	区块链组织ID。 **说明** 在“通道管理”页面中，单击通道名称后的“查看节点”，查看“MSP标识”，去掉MSP即为区块链组织ID。例如：MSP标识为"1232b2032faafee152b58cd82cecf52e49a22a38MSP"，区块链组织ID为"1232b2032faafee152b58cd82cecf52e49a22a38”。	"1232b2032faafee152b58cd82cecf52e49a22a38"
OrgPeers	各个Peer节点的组织ID和域名。	"[{OrgId:" 1232b2032faafee152b58cd82cecf52e49a22a38", PeerDomainName:"peer-1232b2032faafee152b58cd82cecf52e49a22a38-0 .peer-1232b2032faafee152b58cd82cecf52e49a22a38.default.svc.cluster.local "}]"
Opmethod	区块链链码调用类型，目前只有两类：invoke和query。	"invoke"
Args	链码调用参数。	'["invoke","a","b","100"]'（第一个参数为invoke的调用函数，可以是其他值如：move，query等等）

步骤6 配置完成后，构建并运行main()来运行该Demo项目。

代码中将读取conf.yaml以及main.go中的参数QueryReq和InvokeReq请求，并调用RESTAPI的接口"/v1/chaincode/operation"来调用链代码，实现a给b转账功能。运行结果如下：

```
go build main.go (2)
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
The result resp of query is "MTAwMDA="
After  query the count of a is 10000  b is  9000
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
The result resp of query is "OTAwMA=="
After  query the count of a is 10000  b is  9000
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
The result resp of query is "MTAwMDA="
After  query the count of a is 10000  b is  9000
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Process finished with exit code 0
```

说明

本Demo用一个简单的REST客户端调用RESTAPI服务实现了调用链代码，返回的invoke结果为一个base64加密的TransactionID，query结果为base64加密的数据值。代码仅供参考，可以通过该项目代码理解如何调用RESTAPI服务。

----结束

4.5 Nodejs SDK Demo

本Demo提供了一个nodejs的链代码，并提供基于fabric-nodejs-sdk的程序调用链代码，向您演示如何使用nodejs的sdk来访问区块链。有关Fabric nodejs SDK的接口使用说明请参考官方文档：**https://hyperledger.github.io/fabric-sdk-node/release-1.4/index.html**。

说明

只用于场景体验，不用于实际应用。

准备工作

准备顺序	操作指导	说明
1	安装开发工具	请从官网下载并安装nodejs运行环境，地址为**https://nodejs.org/en/download/**
2	下载Demo项目代码	项目代码：**nodejs-demo.zip**

创建区块链实例

步骤1 登录区块链服务管理控制台。

步骤2 单击页面右上角的“创建区块链实例”。

步骤3 根据界面提示，配置区块链基本信息，参数如**表4-13**所示。

须知

为了保证示例Demo成功运行，请在参数配置时按照表格中的参数值填写。

表 4-13 基本信息配置

参数	参数值
区域	使用默认区域
企业项目	default
区块链实例名称	node-sdk-demo

参数	参数值
版本类型	专业版
区块链类型	私有链
Hyperledger Fabric增强版内核	v2.2
共识策略	Raft(CFT)
资源初始密码	请自行设置
资源初始密码确认	请自行设置

步骤4 单击“下一步：资源配置”，进行资源配置，参数如表4-14所示。

表 4-14 资源配置

参数	示例
环境资源	选择“自定义环境”。
集群	创建新的CCE集群
可用区	请自行选择
云主机规格	4核/8GB
云主机个数	1
高可用	否
虚拟私有云	系统自动创建VPC
所在子网	系统自动创建子网
云主机登录方式	密码
root密码	如果填写该项，则以填写值为准，如果不填写，则以资源初始密码为准。
确认密码	-
是否使用CCE集群节点弹性IP	是
弹性IP计费方式	使用默认规格
弹性IP带宽	5 Mbit/s

步骤5 单击“下一步：区块链配置”，进行区块链配置，参数如表4-15所示。

表 4-15 区块链配置

参数	示例
区块链配置	选择“自定义配置”。
区块链管理初始密码	如果填写该项，则以填写值为准，如果不填写，则以资源初始密码为准。
区块链管理确认密码	-
存储卷类型	极速文件存储卷
节点组织存储容量(GB)	使用默认规格。
账本数据存储方式	选择“文件数据库（GoLevelDB）”
peer节点组织	系统已默认创建1个节点组织，名称为organization，将节点数量修改为1。
通道配置	organization节点组织已默认添加进至通道中，保持默认即可。
共识节点数量	使用默认。
安全机制	ECDSA **须知** 安全机制仅支持选择ECDSA。
区块生成配置	否
添加RESTful API支持	否

步骤6 单击“下一步：确认规格”。

步骤7 确认配置信息无误后，根据界面提示创建区块链实例。

请等待数分钟，安装页面提示安装成功，查看实例状态变为“正常”后，表示区块链实例部署完成。

----结束

安装及实例化链代码

步骤1 登录区块链服务管理控制台。

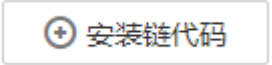
步骤2 单击左侧导航栏中的“实例管理”。

步骤3 在新创建的实例卡片中，单击“区块链管理”，登录链代码管理页面。

步骤4 在登录页面输入用户名、密码，单击“登录”。

说明

用户名为admin，密码为您在创建区块链实例时设置的区块链管理初始密码，如果没有设置区块链管理初始密码，则以资源初始密码为准。

步骤5 在链代码管理页面，单击页面左上角的 

。

安装参数如下：

参数	值
链代码名称	bcsysq
链代码版本	1.0
账本数据存储方式	文件数据库(goleveldb)
选择全部Peer节点	勾选
组织&Peer节点	peer-0
链代码语言	nodejs
链代码文件	下载示例链代码文件：**nodejs-chaincode.zip**
链代码描述	根据需要填写相关描述。

步骤6 单击“安装”完成链代码安装。

步骤7 链代码安装完成后，在链代码列表的“操作”列，单击“实例化”。

实例化参数如下：

参数	值
链代码名称	bcsysq
实例化通道	channel
链代码版本	1.0
初始化函数	init
链代码参数	a,200,b,250
背书策略	任意组织背书
背书组织列表	organization
隐私保护配置	否

----结束

部署应用

步骤1 下载Demo项目工程：**nodejs-demo.zip**，并将Demo项目代码包下载解压到本地并使用IDE打开。

本Demo是一个nodejs编写的js脚本，它包含了fabric-client库，实现链代码a给b转账的功能。请使用个人喜欢的IDE打开。项目包含如下内容：

表 4-16 项目内容

文件	说明
invoke.js	调用链代码，实现a给b转账 。
query.js	查询链代码，查询a的账户余额。
sdk-config.js	解析从BCS控制台下载的区块链网络配置文件sdk-config.json。
sdk-config.json	区块链网络配置信息

步骤2 下载SDK和证书。

1. 在“实例管理”界面，在实例卡片中，单击“获取客户端配置”。
2. 勾选“SDK文件”，SDK配置参数如下：

参数名称	说明
链代码名称	bcsysq **须知** 链代码名称需要和链代码安装&实例化时的一致。
证书存放路径	请指定一个证书存放路径，本例中为：E:\code\temp
通道名称	channel
组织&Peer节点	保持系统默认

   勾选“共识节点证书”。

   勾选“Peer节点证书”，指定节点组织选择organization，勾选“管理员证书”。
3. 单击“下载”，下载SDK配置文件、node-sdk-demo-orderer组织的管理员证书和organization组织的管理员证书。

步骤3 复制并解压。

将**下载SDK和证书**步骤中的zip文件解压，将orderer文件夹、peer文件夹内容全部复制到证书的存放路径下；将sdk-config.json文件复制到证书的存放路径下，并且命名为node-sdk-demo-sdk-config.json。

----结束

调试应用

步骤1 打开sdk-config.js，修改SDK配置文件的路径为node-sdk-demo-sdk-config.json的路径。

```
sdk-config.js | package.json | sdk-test.js | sdk-config.json
1  var fs = require('fs');
2  var path = require("path");
3  // 请在此配置sdk文件的路径和用户名
4  var sdkfile = "E:\\code\\temp\\node-sdk-demo-sdk-config.json";
5  var tmp_user_id = "Admin";
6
7  // 解析sdk-config-yaml文件获取配置信息
8  var sdkconfig = JSON.parse(fs.readFileSync(sdkfile));
9
10 var tmp_orgid = sdkconfig.client.organization;
```

步骤2 在项目工程所在路径执行node query.js，查询a用户的余额（如图所示a的余额为10000）。

```
MINGW64 /e/code/BCS/demo/nodejsdemo (master)
$ node query.js
(node:11224) DeprecationWarning: grpc.load: Use the @grpc/proto-loader
Load peer privateKey and signedCert
Make query
Assigning transaction_id:  e0cfdf8808f64e96aeb1b7a24defb71752255a4ffd46
Query has completed, checking results
Query result count =  1
Response is  10000
```

步骤3 在项目所在的路径执行node invoke.js，执行链码调用，实现a给b转账。如图所示即调用成功。

```
MINGW64 /e/code/BCS/demo/nodejsdemo (master)
$ node invoke.js
(node:12160) DeprecationWarning: grpc.load: Use the @grpc/proto-loader module with grpc.load
Load peer privateKey and signedCert
Assigning transaction_id:  7dc161c1bfb281dbcae23a67ea56a8b6ef5a1c874b9461f07c41115b5b2e724a
Transaction proposal was good
Successfully sent Proposal and received ProposalResponse: Status - 200, message - "OK"
The transaction has been committed on peer 10.154.197.169:30605
Send transaction promise and event listener promise have completed
Successfully sent transaction to the orderer.
Successfully committed the change to the ledger by the peer
```

步骤4 再次查询a的余额（如图所示a的余额为9990），可以看到转账成功。

```
MINGW64 /e/code/BCS/demo/nodejsdemo (master)
$ node query.js
(node:12792) DeprecationWarning: grpc.load: Use the @grpc/proto-loader modul
Load peer privateKey and signedCert
Make query
Assigning transaction_id:  80fe189b2aa9399ae207d09e3bf10ae2e2996b8caf1ec4aa5
Query has completed, checking results
Query result count =  1
Response is  9990
```

----结束

说明

- 本Demo借鉴了Fabric社区的Demo，更多的样例可以参考**https://github.com/hyperledger/fabric-samples**。

- 如何在链代码中自定义npm仓库?
  - 在链代码包的文件路径下创建一个名为 .npmrc 的个人配置文件，内容为：registry=https://registry.npm.taobao.org/
  - 这样在链代码容器实例化的时候就会从指定的仓库里面拉取类库。
  - 通过npm config get registry查看是否正确：

5 区块链中间件接口

5.1 概述

本章节主要介绍数据面的API接口，管理面的API接口请参见《API接口参考》。

数据面请求EndPoint可以通过查询服务实例详细信息接口返回结果中basic_info->agent_portal_addrs字段的值获取，请求示例：https://192.168.0.90:30603/v2/agent/apis/tokens。

5.2 链代码调用

功能介绍

对已经部署并已经实例化的区块链链代码进行调用（invoke）和查询（query）。

URI

POST /v1/chaincode/operation

请求消息

表 5-1 请求参数

参数	是否必选	参数类型	描述
channelId	是	String	区块链通道ID
chaincodeId	是	String	链代码ID
chaincodeVersion	否	String	链代码版本
userId	是	String	由组织CA签发的用户ID，目前区块链服务默认生成的都为User1
orgId	是	String	区块链组织ID

参数	是否必选	参数类型	描述
orgPeers	是	String	由组织中每个节点的组织ID和域名组成的数组，形如： [{"orgId":"7258adda1803f4137eff4813e7aba323018200c5","peerDomainName":"peer-7258adda1803f4137eff4813e7aba323018200c5-0.peer-7258adda1803f4137eff4813e7aba323018200c5.default.svc.cluster.local"}]
opmethod	是	String	区块链链码调用类型，目前只有两类调用方法：invoke和query。
args	是	String	链码调用参数，形如：["invoke","a","b","1"]
timestamp	是	String	格式为2018-10-31T17:28:16+08:00
cert	是	String	用户的证书文件，以字符串形式上传

说明

以上参数获取方式，详细参见用户指南中的**链代码管理**和**区块浏览器**章节。

- 在链代码管理页面中，单击链代码名称前的 ，展开链代码详细信息，您可以查看当前链代码的版本列表、安装列表和实例化情况。
- 在区块浏览器页面中，在通道下拉框中选择一个通道，下方的数据即可实时刷新供您查看多项数据。区块链相关信息的查询功能，包括区块数量、交易数量、区块详细信息、交易详细信息、性能数据及节点状态等。
- 为了保证交易安全性，需要使用Fabric用户证书（证书获取方式请见**下载用户证书**）中的私钥对请求消息体进行签名（目前只支持椭圆曲线，暂不支持国密等其他加密算法），并将签名结果放到消息头部x-bcs-signature-sign字段。

链码REST API自定义了一些消息头，请参见**表5-2**。

表 5-2 自定义消息头

名称	是否必选	描述
x-bcs-signature-sign	是	链码调用请求消息体签名。
x-bcs-signature-method	是	加密类型，目前固定是SW。
x-bcs-signature-sign-gzip	是	Sign是否选择Gzip压缩。0表示否，1表示是。

说明

x-bcs-signature-sign：为了保证只允许有权限的调用端才能够进行合法的链码调用，需要使用**下载用户证书**中下载的用户私钥以ECDSA椭圆曲线的加密方式对整个请求消息体的SHA256摘要进行加密签名，x-bcs-signature-sign值即为签名结果。

下载用户证书

进行API调用前，需要下载区块链服务中已经配置生成的用户证书。

步骤1 登录区块链服务控制台。

步骤2 进入“实例管理”界面，可以看到已经创建的区块链实例卡片。

步骤3 单击卡片上的“获取客户端配置”，勾选“Peer节点证书”。选择指定节点组织，勾选“用户证书”。

步骤4 单击“下载”，下载相应组织的用户证书。

步骤5 解压证书，其中msp文件夹中，keystore文件夹存储的是组织用户私钥，signcerts文件夹存储的是用户证书（公钥）。

----结束

响应消息

- 当opmethod为invoke时，返回值是base64加密的transactionID。
- 当opmethod为query时，返回值是base64加密的链代码的返回值。

示例

下面是调用invoke类型的链代码示例。

- 请求示例

```
{
  "channelId": "testchannel",
  "chaincodeId": "zmmcode",
  "chaincodeVersion": "1.0",
  "userId": "User1",
  "orgId": "7258adda1803f4137eff4813e7aba323018200c5",
  "orgPeers": "[{\"orgId\":\"7258adda1803f4137eff4813e7aba323018200c5\",\"peerDomainName\":
\"peer-7258adda1803f4137eff4813e7aba323018200c5-0.peer-7258adda1803f4137eff4813e7aba32301
8200c5.default.svc.cluster.local\"}]",
  "opmethod": "invoke",
  "args": "[\"invoke\",\"a\",\"b\",\"1\"]",
  "timestamp": "2018-10-31T17:28:16+08:00",
  "cert": "-----BEGIN CERTIFICATE-----
\nMIIDBzCCAq2gAwIBAgIQEXPZlMsReamxVtVNnKwCCzAKBggqhkjOPQQDAjCCAQQx
\nDjAMBgNVBAYTBUNISU5BMRAwDgYDVQQIEwdCRUlKSU5HMRAwDgYDVQQHEwdCRUlK
\nSU5HMXkwdwYDVQQKE3A3MjU4YWRkYTE4MDNmNDEzN2VmZjQ4MTNlN2FiYTMyMzAx
\nODIwMGM1LnBlZXItNzI1OGFkZGExODAzZjQxMzdlZmY0ODEzZTdhYmEzMjMwMTgy
\nMDBjNS5kZWZhdWx0LnN2Yy5jbHVzdGVyLmxvY2FsMVMwUQYDVQQDE0pjYS5wZWVy
\nLTcyNThhZGRhMTgwM2Y0MTM3ZWZmNDgxM2U3YWJhMzIzMDE4MjAwYzUuZGVmYXVs
\ndC5zdmMuY2x1c3Rlci5sb2NhbDAeFw0xODEwMzAwMjQ5MjZaFw0yODEwMjcwMjQ5\nMjZaMIG1M
Q4wDAYDVQQGEwVDSElOQTEQMA4GA1UECBMHQkVJSklORzEQMA4GA1UE\nBxMHQkVJSklORzF/
MH0GA1UEAwx2VXNlcjFANzI1OGFkZGExODAzZjQxMzdlZmY0\nODEzZTdhYmEzMjMwMTgyMDBjNS5
wZWVyLTcyNThhZGRhMTgwM2Y0MTM3ZWZmNDgx
\nM2U3YWJhMzIzMDE4MjAwYzUuZGVmYXVsdC5zdmMuY2x1c3Rlci5sb2NhbDBZMBMG
\nByqGSM49AgEGCCqGSM49AwEHA0IABPMrzoJL/MHeSFPFOJWLqnJ0sqB0it7wDIOq\n
+eTSvvPpGk1BIDmb2n13K5V04RO8xNezDQ7I6rW4LF2elq14eH+jTTBLMA4GA1Ud\nDwEB/
wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIFBXQ5TC4acFeTlT
\nJuDZg62XkXCdnOfvbejSeKI2TXoIMAoGCCqGSM49BAMCA0gAMEUCIQCadHIKl0Mk
\nYn0WZizyDZYR4rT2q0nzjFaiW+YfV5FBjAIgNalKUe3rIwXJvXORV4ZXurEua2Ag\nQmhcjRnVwPTjpTE=
```

```
\n-----END CERTIFICATE-----\n"
}
```

- 响应示例

```
After  invoke the count of a is 188  b is  262
```

错误码

请参见**错误码**页面。

5.3 链代码管理

5.3.1 获取 Token

功能介绍

根据区块链浏览器用户名密码获取Token。

URI

GET /v2/agent/apis/tokens

请求参数

表 5-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-User	是	String	用户名
X-Auth-Pass	是	String	密码

响应参数

状态码： 200

表 5-4 响应 Body 参数

参数	参数类型	描述
user	String	用户名
token	String	token
expire_time	Long	失效时间

状态码： 400

表 5-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/tokens
```

响应示例

状态码： 200

Success

```
{
  "user" : "admin",
  "token" :
"ACiUxRlk2Jcu2d5bp2SyfP7abHV1nVKlo9Mm2AxI9dDN7mdHXXHBdg=kaIglHeZLTu6UeIu5YRcXTlpUPXylEG3
ESNu0vIukrikG6SVhO393ds8rG+aRnZ+mMyookApP",
  "expire_time" : 1605867860701
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.2 安装链代码

功能介绍

在区块链节点上安装链代码，部分场景只支持go语言链码

URI

POST /v2/agent/apis/chaincode/install

请求参数

表 5-6 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

表 5-7 FormData 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，以小写字母开头，支持小写字母和数字 最小长度：**6** 最大长度：**25**
chaincode_version	是	String	链代码版本，只允许使用数字、点（.）、横杠（-），必须以数字开头和结尾，且点和横杠不能相邻 最小长度：**1** 最大长度：**14**
target_peers	是	String	链代码安装Peer列表信息，例如： [{"org_id":"9fb42c91458763990a45b62af92546a21f168dae","org_name":"organization3","peer_id":"peer-9fb42c91458763990a45b62af92546a21f168dae-0.peer-9fb42c91458763990a45b62af92546a21f168dae.default.svc.cluster.local","peer_name":"peer-0"}]
description	否	String	链代码描述
chaincode_language	是	String	链代码编程语言，例如 golang
db_type	否	String	账本数据存储方式，请按照实例创建时的账本数据存储方式填写。例如 goleveldb，couchdb
security_check	否	String	链代码安全检查选项，目前只对golang语言链代码有效，true表示开启，false表示关闭，默认为false
file	是	File	链代码zip文件

响应参数

状态码： 200

表 5-8 响应 Body 参数

参数	参数类型	描述
total_peer_num	Integer	链代码操作Peer总数
success_peer_num	Integer	操作成功Peer数量
fail_peer_num	Integer	操作失败Peer数量
fail_peers	Array of strings	操作失败Peer信息

状态码： 400

表 5-9 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
POST https://192.168.0.90:30603/v2/agent/apis/chaincode/install

{chaincode_name:gochaincode3chaincode_version:2.0target_peers:
[{"org_id":"9802af57cfab764dc12b860c44b01969575e83c9","org_name":"organization","peer_id":"peer-9802af57cfab764dc12b860c44b01969575e83c9-1.peer-9802af57cfab764dc12b860c44b01969575e83c9.default.svc.cluster.local","peer_name":"node-1"}]description:
22222222chaincode_language:golangdb_type:goleveldbsecurity_check:true}
```

响应示例

状态码： 200

Success

```
{
  "total_peer_num" : 4,
  "success_peer_num" : 4,
  "fail_peer_num" : 0,
  "fail_peers" : [ ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.3 实例化链代码

功能介绍

实例化链代码

URI

POST /v2/agent/apis/chaincode/instantiate

请求参数

表 5-10 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

表 5-11 请求 Body 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，以小写字母开头，支持小写字母和数字，长度6-25位 最小长度：**6** 最大长度：**25**
chaincode_version	是	String	链代码版本，只允许使用数字、点（.）、横杠（-），必须以数字开头和结尾，且点和横杠不能相邻
channel_name	是	String	实例化通道名称

参数	是否必选	参数类型	描述
endorsement_policy	是	**Policy** object	背书策略
init_variable	是	**InitArgs** object	初始化函数及参数
private_data	否	String	隐私保护配置数据，json数据的string格式，json数据结构举例：[{"name":"kvstore-collection","policy":"OR('9b4ea44913e8eed42cb056177b46191e1d316678MSP.member','9b4ea44913e8eed42cb056177b46191e1d316678MSP.member')","requiredPeerCount":0,"maxPeerCount":2,"blockToLive":0,"memberOnlyRead":true}]

表 5-12 Policy

参数	是否必选	参数类型	描述
operation	是	String	操作符。 OR：任意组织背书 AND：全部组织背书
group	是	Array of **OrgPolicy** objects	组织背书成员

表 5-13 OrgPolicy

参数	是否必选	参数类型	描述
org_id	是	String	组织ID
category	否	String	成员类型：member、admin

表 5-14 InitArgs

参数	是否必选	参数类型	描述
func_name	是	String	初始化函数名
args	是	Array of strings	初始化参数列表

响应参数

状态码： 400

表 5-15 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
POST https://192.168.0.90:30603/v2/agent/apis/chaincode/instantiate

{
  "chaincode_name" : "gochaincode2",
  "chaincode_version" : "1.0",
  "channel_name" : "channel001",
  "endorsement_policy" : {
    "operation" : "AND",
    "group" : [ {
      "org_id" : "8cc7155fb1f26ebac1743f481386c14223be511f",
      "category" : "member"
    } ]
  },
  "init_variable" : {
    "func_name" : "init",
    "args" : [ "a", "200", "b", "100" ]
  },
  "private_data" : "[{\"name\": \"kvstore-collection\",\"policy\":
\"OR('8cc7155fb1f26ebac1743f481386c14223be511fMSP.member','8cc7155fb1f26ebac1743f481386c14223b
e511fMSP.member')\",\"requiredPeerCount\": 0,\"maxPeerCount\": 2,\"blockToLive\": 0,\"memberOnlyRead
\": true}]"
}
```

响应示例

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.4 获取安装的链码列表

功能介绍

获取已经安装的链码列表

URI

GET /v2/agent/apis/chaincodes

表 5-16 Query 参数

参数	是否必选	参数类型	描述
offset	否	Integer	查询链代码列表的起始位置，默认为0
limit	否	Integer	查询链代码列表的数量，默认为10

请求参数

表 5-17 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-18 响应 Body 参数

参数	参数类型	描述
count	Integer	链码总数
chaincodes	Array of **ChaincodeInfo** objects	链码列表

表 5-19 ChaincodeInfo

参数	参数类型	描述
chaincode_name	String	链码名称
chaincode_language	String	链码开发语言
update_time	String	链码更新时间
chaincode_version	String	链码版本，多个链码之间以逗号(,)分割
install_org_infos	Array of **PeerInfo** objects	链码的安装信息
instantiated_channel	**instantiated_channel** object	链码通道信息
instantiated_info	**instantiated_info** object	实例化信息

表 5-20 PeerInfo

参数	参数类型	描述
org_name	String	组织名称
org_id	String	组织id
peer_name	String	节点名称
peer_id	String	节点id
status	String	节点状态
channels	Array of strings	未实例化的peer信息
url	String	Peer节点的url信息
peer	String	Peer节点的内部域名

表 5-21 instantiated_channel

参数	参数类型	描述
error	Array of **CCInstantiatedChannelError** objects	实例化错误信息
success	Array of strings	成功的通道
inprogress	Array of strings	实例化进度

表 5-22 CCInstantiatedChannelError

参数	参数类型	描述
channel_name	String	错误通道名
error_detail	String	错误详情

表 5-23 instantiated_info

参数	参数类型	描述
channels	Array of **channels** objects	通道信息

表 5-24 channels

参数	参数类型	描述
channel_id	String	通道名称
orgs	Array of **orgs** objects	通道内组织信息
versions	Array of strings	版本信息

表 5-25 orgs

参数	参数类型	描述
org_name	String	组织名
org_id	String	组织id

状态码： 400

表 5-26 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/chaincodes
```

响应示例

状态码： 200

Success

```
{
  "count" : 12,
  "chaincodes" : [ {
    "chaincode_name" : "test001",
    "chaincode_version" : "1.0",
    "update_time" : "2021-01-12T11:32:04.193358708+08:00",
    "instantiated_info" : {
      "channels" : [ {
        "channel_id" : "channel",
        "versions" : [ "1.0" ]
      }, {
        "channel_id" : "testchannel",
        "orgs" : null,
        "versions" : [ "1.0" ]
      } ]
    },
    "chaincode_language" : "golang"
  } ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.5 查询指定链码版本信息

功能介绍

查询指定链码版本信息

URI

GET /v2/agent/apis/chaincode/versions

表 5-27 Query 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，以小写字母开头，支持小写字母和数字，长度6-25位 最小长度：**6** 最大长度：**25**

请求参数

表 5-28 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-29 响应 Body 参数

参数	参数类型	描述
versions	Array of **ChaincodeVersion** objects	链码版本信息

表 5-30 ChaincodeVersion

参数	参数类型	描述
version	String	链码版本
hash_code	String	链码版本哈希值
description	String	链码版本描述
install_time	String	链码版本安装时间
update_time	String	链码版本更新时间
instantiate_status	Boolean	链码版本实例化状态
security_check_status	Integer	链代码安全检查状态（0：不存在，1：运行中，2：完成，3：失败）
uninstantiated_peer_infos	Array of **PeerInfo** objects	未实例化的peer信息

表 5-31 PeerInfo

参数	参数类型	描述
org_name	String	组织名称
org_id	String	组织id
peer_name	String	节点名称
peer_id	String	节点id
status	String	节点状态
channels	Array of strings	未实例化的peer信息
url	String	Peer节点的url信息
peer	String	Peer节点的内部域名

状态码： 400

表 5-32 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/chaincode/versions?chaincode_name=chaincode
```

响应示例

状态码： 200

Success

```
{
  "versions" : [ {
    "version" : "1.0",
    "hash_code" : "1473b4807fe9f970d1ba56192e41d39c7d621d07d80e603cf75ed3982b81034d",
    "description" : "",
    "install_time" : "2021-01-11T11:27:12.093454567+08:00",
    "update_time" : "2021-01-11T11:27:12.093454789+08:00",
    "instantiate_status" : false,
    "uninstantiated_peer_infos" : [ {
      "org_name" : "organization",
      "org_id" : "57e7914450b098771f5106acaf02be8a61894fae",
      "peer_name" : "peer-0",
      "peer_id" :
"peer-57e7914450b098771f5106acaf02be8a61894fae-0.peer-57e7914450b098771f5106acaf02be8a61894fae
.default.svc.cluster.local"
    }, {
      "org_name" : "organization",
      "org_id" : "57e7914450b098771f5106acaf02be8a61894fae",
      "peer_name" : "peer-1",
      "peer_id" :
"peer-57e7914450b098771f5106acaf02be8a61894fae-1.peer-57e7914450b098771f5106acaf02be8a61894fae
.default.svc.cluster.local"
    } ],
    "security_check_status" : 2
  } ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.6 查询链代码安装信息

功能介绍

查询某个链代码在节点上的安装信息

URI

GET /v2/agent/apis/chaincode/install

表 5-33 Query 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，以小写字母开头，支持小写字母和数字，长度6-25位 最小长度：**6** 最大长度：**25**

请求参数

表 5-34 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-35 响应 Body 参数

参数	参数类型	描述
result	Array of PeerInstallInfo objects	节点链代码安装信息

表 5-36 PeerInstallInfo

参数	参数类型	描述
org_name	String	节点所在组织名称
org_id	String	节点所在组织ID
peer_name	String	节点名称
peer_id	String	节点ID
install_status	String	链代码安装情况：installed / uninstalled
version	String	链代码版本号

状态码： 400

表 5-37 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/chaincode/install?chaincode_name=chaincode
```

响应示例

状态码： 200

Success

```
{
  "result" : [ {
    "org_name" : "org1",
    "org_id" : "65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3",
    "peer_name" : "peer-0",
    "peer_id" :
```

```
"peer-65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3-0.peer-65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3.default.svc.cluster.local",
    "install_status" : "uninstalled",
    "version" : ""
  }, {
    "org_name" : "org1",
    "org_id" : "65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3",
    "peer_name" : "peer-1",
    "peer_id" :
"peer-65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3-1.peer-65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3.default.svc.cluster.local",
    "install_status" : "installed",
    "version" : "1.0"
  } ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.7 查询链代码实例化信息

功能介绍

查询某个链代码在区块链通道上的实例化信息

URI

GET /v2/agent/apis/chaincode/instantiate

表 5-38 Query 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，以小写字母开头，支持小写字母和数字，长度6-25位 最小长度：**6** 最大长度：**25**

请求参数

表 5-39 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-40 响应 Body 参数

参数	参数类型	描述
result	Array of **ChannelInstantiateInfo** objects	通道链代码实例化信息

表 5-41 ChannelInstantiateInfo

参数	参数类型	描述
channel_name	String	通道名称
instantiate_info	**InstantiateInfo** object	实例化信息
endorsement_policy	String	背书策略
version	String	链代码版本
orgs_info	Array of **OrgInfo** objects	通道组织信息
has_private_data	Integer	是否有隐私数据，1表示有，0表示无

表 5-42 InstantiateInfo

参数	参数类型	描述
status	String	实例化状态，取值有CHAINCODE_INSTANTIATED（实例化成功）,CHAINCODE_INSTANTIATION_INPROGRESS（实例化进行中）,CHAINCODE_INSTANTIATION_FAILED（实例化失败）
code	String	实例化结果编码
reason	String	实例化结果理由
detail	String	实例化结果详情

表 5-43 OrgInfo

参数	参数类型	描述
org_name	String	组织名称
org_id	String	组织ID

状态码： 400

表 5-44 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/chaincode/instantiate?chaincode_name=chaincode
```

响应示例

状态码： 200

Success

```
{
  "result" : [ {
    "channel_name" : "channel",
    "instantiate_info" : {
      "status" : "CHAINCODE_INSTANTIATED",
```

```
    "code" : "1000",
    "reason" : "1000",
    "detail" : ""
  },
  "endorsement_policy" : "OR,org1,org2",
  "version" : "2.0",
  "orgs_info" : [ {
    "org_name" : "org1",
    "org_id" : "65cfb1c760f24058c865ffcfd8ce1cdb690bf2a3"
  }, {
    "org_name" : "org2",
    "org_id" : "a48c4ed995238eceaee3fe738f1871b2e58db350"
  } ],
  "has_private_data" : 0
 } ]
}
```


状态码： 400

Bad Request


```
{
 "error_code" : "BCS.4000013",
 "error_message" : "request body is too large"
}
```


状态码

状态码	描述
200	Success
400	Bad Request

5.3.8 查询应用链信息

功能介绍

通道概要信息查询

URI

GET /v2/agent/apis/channel/{channel_name}/summary

表 5-45 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称

请求参数

表 5-46 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。通过调用获取TokenAPI获取Token。

响应参数

状态码： 200

表 5-47 响应 Body 参数

参数	参数类型	描述
peer_num	Number	peer数量
chaincodes	Number	合约数量
block_num	Number	区块数量
transaction_num	Number	交易数量
bph	Array of **bph** objects	区块产生数量统计，按小时统计
bpm	Array of **bpm** objects	区块产生数量统计，按每5分钟统计
tph	Array of **tph** objects	合约交易数量统计，按小时统计
tpm	Array of **tpm** objects	合约交易数量统计，按每5分钟统计
orgs_map	Array of **orgs_map** objects	组织交易统计
peers_list	Array of **peers_list** objects	peer列表

表 5-48 bph

参数	参数类型	描述
block	Number	区块数量

参数	参数类型	描述
time	String	时间

表 5-49 bpm

参数	参数类型	描述
block	Number	区块数量
time	String	时间

表 5-50 tph

参数	参数类型	描述
tx	Number	交易数量
time	String	时间

表 5-51 tpm

参数	参数类型	描述
tx	Number	交易数量
time	String	时间

表 5-52 orgs_map

参数	参数类型	描述
org_name	String	组织名称
tx_num	Number	交易数量

表 5-53 peers_list

参数	参数类型	描述
org_name	String	组织名称
org_id	String	组织ID
peer	String	Peer名称
url	String	PeerURL

参数	参数类型	描述
status	String	Peer状态

状态码： 400

表 5-54 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET https://192.168.0.90:30603/v2/agent/apis/channel/channel/summary
```

响应示例

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.9 查询区块列表

功能介绍

查询区块列表

URI

GET /v2/agent/apis/channel/{channel_name}/blocks

表 5-55 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称，名称长度限制：4-24，不能与系统通道名称（ testchainid ）相同

表 5-56 Query 参数

参数	是否必选	参数类型	描述
is_pagination	否	Boolean	是否分页，默认为否 缺省值：**false**
offset	否	Integer	偏移量，默认为0 最小值：**0** 缺省值：**0**
limit	否	Integer	每页显示的条目数量，默认为10，取值范围为1-50 最小值：**1** 最大值：**50** 缺省值：**10**

请求参数

表 5-57 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-58 响应 Body 参数

参数	参数类型	描述
count	Integer	区块总数
data	Array of **BlockHeader** objects	区块

表 5-59 BlockHeader

参数	参数类型	描述
block_number	Integer	区块号
block_hash	String	区块哈希
transaction_count	Integer	交易总数
data_hash	String	数据哈希
previous_hash	String	前一个区块哈希
timestamp	String	时间戳
organization_maps	Map<String,Integer>	key:creatorMSP，value:数量

状态码： 400

表 5-60 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

- 默认模式请求

  ```
  GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/blocks
  ```

- 分页模式请求

  ```
  GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/blocks?
  is_pagination=true&offset=1&limit=50
  ```

响应示例

状态码： 200

Success

```
{
  "count" : 1,
  "data" : [ {
    "block_number" : 0,
    "block_hash" : "Yux2Ea0RNZM95+3R95mvdlI8mH1dvmTSTylPwwzMsbY",
    "transaction_count" : 1,
    "data_hash" : "+3B6RFfbQisE8zt2BeWTvCwP1JbmQLIPeQoiKxoCQVA",
    "previous_hash" : "FlEClYDQJ4cHaEslex9usupte0EqbHyymQ+zUaQcjyE",
```



```
    "timestamp" : "2021-01-20T14:38:39+08:00",
    "organization_maps" : "{\"d565e95144ae53b9b3f556de613513f257e720ecMSP\":49}"
  } ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_msg" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.10 查询交易列表

功能介绍

查询交易列表

URI

GET /v2/agent/apis/channel/{channel_name}/transactions

表 5-61 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称，名称长度限制：4-24，不能与系统通道名称（testchainid）相同

表 5-62 Query 参数

参数	是否必选	参数类型	描述
is_pagination	否	Boolean	是否分页，默认为否 缺省值：**false**
offset	否	Integer	偏移量，默认为0 最小值：**0** 缺省值：**0**

参数	是否必选	参数类型	描述
limit	否	Integer	每页显示的条目数量，默认为10，取值范围为1-50 最小值：**1** 最大值：**50** 缺省值：**10**

请求参数

表 5-63 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-64 响应 Body 参数

参数	参数类型	描述
count	Integer	交易总数
data	Array of **TransactionSummary** objects	交易

表 5-65 TransactionSummary

参数	参数类型	描述
organization_name	String	创建者组织
type	String	交易类型
transaction_id	String	交易id
chaincode_name	String	链码名称
timestamp	String	时间戳

参数	参数类型	描述
channel_name	String	通道名称
creator_msp	String	身份信息
chaincode_version	String	链码版本
block_number	Integer	区块号

状态码： 400

表 5-66 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

- 默认模式请求

```
GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/transactions
```

- 分页模式请求

```
GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/transactions?
is_pagination=true&offset=1&limit=50
```

响应示例

状态码： 200

Success

```
{
  "count" : 1,
  "data" : [ {
    "block_number" : 0,
    "transaction_id" : "",
    "channel_name" : "channel",
    "creator_msp" : "e784724be5ed75f59b2809e4f0965a10679ae113MSP",
    "type" : "CONFIG",
    "chaincode_name" : "",
    "chaincode_version" : "",
    "timestamp" : "2021-01-20T14:38:27+08:00",
    "organization_name" : "orderer"
  } ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_msg" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.11 查询交易总数

功能介绍

查询交易总数

URI

GET /v2/agent/apis/channel/{channel_name}/transactions/count

表 5-67 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称

请求参数

表 5-68 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-69 响应 Body 参数

参数	参数类型	描述
channel_id	String	通道id
block_height	Integer	区块高度
transaction_num	Integer	交易数量

状态码： 400

表 5-70 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/transactions/count
```

响应示例

状态码： 200

Success

```
{
  "channel_id" : "channel",
  "block_height" : 2,
  "transaction_num" : 2
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success

状态码	描述
400	Bad Request

5.3.12 查询区块交易列表

功能介绍

查询区块交易列表

URI

GET /v2/agent/apis/channel/{channel_name}/blocks/{block_num}/transactions

表 5-71 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称，名称长度限制：4-24，不能与系统通道名称（ testchainid ）相同
block_num	是	String	区块号

请求参数

表 5-72 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-73 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array of **ShowTransactionDetailRes** objects	Success

表 5-74 ShowTransactionDetailRes

参数	参数类型	描述
read_set	Map<String,Array<**KVRead**>>	读集 "map[string][]KVRead key:chaincode value:键值对数组"
write_set	Map<String,Array<**KVWrite**>>	写集 "map[string][]KVWrite key:chaincode value:键值对数组"
validation_code	String	验证代码
endorser_organizations	Array of strings	背书组织
proposal_hash	String	请求数据哈希
transaction_summary	Object	交易详情

表 5-75 KVRead

参数	参数类型	描述
key	String	读键
version	**version** object	读集键的版本

表 5-76 version

参数	参数类型	描述
block_num	Integer	区块数
tx_num	Integer	交易数

表 5-77 KVWrite

参数	参数类型	描述
key	String	写键
is_delete	Boolean	是否删除
value	String	写值

表 5-78 TransactionSummary

参数	参数类型	描述
organization_name	String	创建者组织
type	String	交易类型
transaction_id	String	交易id
chaincode_name	String	链码名称
timestamp	String	时间戳
channel_name	String	通道名称
creator_msp	String	身份信息
chaincode_version	String	链码版本
block_number	Integer	区块号

状态码： 400

表 5-79 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/blocks/1/transactions
```

响应示例

状态码： 200

Success

```
[ {
  "transaction_summary" : {
    "block_number" : 29,
    "transaction_id" : "6d704b217e17e16de71029b70f17a1ced35c055279f655dfd096bebf978a0546",
    "channelName" : "channel",
    "creator_msp" : "282f3c713ea1cec646aa7c640defca9c4f64bd88MSP",
    "type" : "ENDORSER_TRANSACTION",
```



```
  "chaincode_name" : "kvtest",
  "chaincode_version" : "1.0",
  "timestamp" : "2021-01-20T19:30:28+08:00",
  "organization_name" : "organization"
 },
 "validation_code" : "VALID",
 "endorser_organizations" : [ "282f3c713ea1cec646aa7c640defca9c4f64bd88MSP" ],
 "proposal_hash" : "k1h2ewweWGrWNmmcu7UvzJ8Aw2G190SQzV+lBAAl4gw=",
 "read_set" : {
  "kvtest" : null,
  "lscc" : [ {
   "key" : "kvtest",
   "version" : {
    "block_num" : 2
   }
  } ]
 },
 "write_set" : {
  "kvtest" : [ {
   "key" : "a1",
   "is_delete" : false,
   "value" : "1"
  } ],
  "lscc" : [ ]
 }
} ]
```


状态码： 400

Bad Request


```
{
 "error_code" : "BCS.4000013",
 "error_msg" : "request body is too large"
}
```


状态码

状态码	描述
200	Success
400	Bad Request

5.3.13 查询交易详情

功能介绍

查询交易详情

URI

GET /v2/agent/apis/channel/{channel_name}/transactions/{transaction_id}/detail

表 5-80 路径参数

参数	是否必选	参数类型	描述
channel_name	是	String	通道名称
transaction_id	是	String	交易id号

请求参数

表 5-81 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-82 响应 Body 参数

参数	参数类型	描述
read_set	Map<String,Array<**KVRead**>>	读集 "map[string][]KVRead key:chaincode value:键值对数组"
write_set	Map<String,Array<**KVWrite**>>	写集 "map[string][]KVWrite key:chaincode value:键值对数组"
validation_code	String	验证代码
endorser_organizations	Array of strings	背书组织
proposal_hash	String	请求数据哈希
transaction_summary	Object	交易详情

表 5-83 KVRead

参数	参数类型	描述
key	String	读键

参数	参数类型	描述
version	**version** object	读集键的版本

表 5-84 version

参数	参数类型	描述
block_num	Integer	区块数
tx_num	Integer	交易数

表 5-85 KVWrite

参数	参数类型	描述
key	String	写键
is_delete	Boolean	是否删除
value	String	写值

表 5-86 TransactionSummary

参数	参数类型	描述
organization_name	String	创建者组织
type	String	交易类型
transaction_id	String	交易id
chaincode_name	String	链码名称
timestamp	String	时间戳
channel_name	String	通道名称
creator_msp	String	身份信息
chaincode_version	String	链码版本
block_number	Integer	区块号

状态码： 400

表 5-87 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET  https://192.168.0.90:30603/v2/agent/apis/channel/channel/transactions/1111111/detail
```

响应示例

状态码： 200

Success

```
{
  "transaction_summary" : {
    "block_number" : 29,
    "transaction_id" : "6d704b217e17e16de71029b70f17a1ced35c055279f655dfd096bebf978a0546",
    "channelName" : "channel",
    "creator_msp" : "282f3c713ea1cec646aa7c640defca9c4f64bd88MSP",
    "type" : "ENDORSER_TRANSACTION",
    "chaincode_name" : "kvtest",
    "chaincode_version" : "1.0",
    "timestamp" : "2021-01-20T19:30:28+08:00",
    "organization_name" : "organization"
  },
  "validation_code" : "VALID",
  "endorser_organizations" : [ "282f3c713ea1cec646aa7c640defca9c4f64bd88MSP" ],
  "proposal_hash" : "k1h2ewweWGrWNmmcu7UvzJ8Aw2G190SQzV+lBAAl4gw=",
  "read_set" : {
    "kvtest" : null,
    "lscc" : [ {
      "key" : "kvtest",
      "version" : {
        "block_num" : 2
      }
    } ]
  },
  "write_set" : {
    "kvtest" : [ {
      "key" : "a1",
      "is_delete" : false,
      "value" : "1"
    } ],
    "lscc" : [ ]
  }
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.14 查询节点状态

功能介绍

查询节点状态

URI

GET /v2/agent/apis/peers

请求参数

表 5-88 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 5-89 响应 Body 参数

参数	参数类型	描述
peers	Map<String,PeerInfo>	key:节点域名，value:节点详情

表 5-90 PeerInfo

参数	参数类型	描述
org_name	String	组织名称
org_id	String	组织id
peer_name	String	节点名称
peer_id	String	节点id

参数	参数类型	描述
status	String	节点状态
channels	Array of strings	未实例化的peer信息
url	String	Peer节点的url信息
peer	String	Peer节点的内部域名

状态码： 400

表 5-91 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
GET  https://192.168.0.90:30603/v2/agent/apis/peers
```

响应示例

状态码： 200

Success

```
{
  "peers" : {
    "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-0.peer-
a9e940a9e947e8af0c9c2fb98d0129e56210d6b5.default.svc.cluster.local" : {
      "org_name" : "organization",
      "org_id" : "a9e940a9e947e8af0c9c2fb98d0129e56210d6b5",
      "peer" : "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-0.peer-
a9e940a9e947e8af0c9c2fb98d0129e56210d6b5.default.svc.cluster.local",
      "peer_name" : "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-0",
      "url" : "100.95.146.117:30610",
      "channels" : [ "channel" ],
      "status" : "running"
    },
    "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-1.peer-
a9e940a9e947e8af0c9c2fb98d0129e56210d6b5.default.svc.cluster.local" : {
      "org_name" : "organization",
      "org_id" : "a9e940a9e947e8af0c9c2fb98d0129e56210d6b5",
      "peer" : "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-1.peer-
a9e940a9e947e8af0c9c2fb98d0129e56210d6b5.default.svc.cluster.local",
      "peer_name" : "peer-a9e940a9e947e8af0c9c2fb98d0129e56210d6b5-1",
      "url" : "100.95.146.117:30611",
      "channels" : [ "channel" ],
      "status" : "running"
    }
```

```
  }
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.15 删除链代码

功能介绍

删除区块链节点上的链代码

URI

DELETE /v2/agent/apis/chaincode/uninstall

请求参数

表 5-92 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

表 5-93 请求 Body 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称,以小写字母开头，支持小写字母和数字，长度6-25位 最小长度：**6** 最大长度：**25**

参数	是否必选	参数类型	描述
chaincode_version	是	String	链代码版本,只允许使用数字、点（.）、横杠（-），必须以数字开头和结尾，且点和横杠不能相邻
target_peers	是	Array of **TargetPeer** objects	卸载链代码peer信息

表 5-94 TargetPeer

参数	是否必选	参数类型	描述
org_id	是	String	peer所属组织ID
peer_id	是	String	Peer ID

响应参数

状态码： 200

表 5-95 响应 Body 参数

参数	参数类型	描述
total_peer_num	Integer	链代码操作Peer总数
success_peer_num	Integer	操作成功Peer数量
fail_peer_num	Integer	操作失败Peer数量
fail_peers	Array of strings	操作失败Peer信息

状态码： 400

表 5-96 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述

参数	参数类型	描述
error_msg	String	错误描述

请求示例

```
DELETE https://192.168.0.90:30603/v2/agent/apis/chaincode/uninstall

{
  "chaincode_name" : "chaincode1",
  "chaincode_version" : "1.0",
  "target_peers" : [ {
    "org_id" : "9802af57cfab764dc12b860c44b01969575e83c9",
    "peer_id" :
"peer-9802af57cfab764dc12b860c44b01969575e83c9-1.peer-9802af57cfab764dc12b860c44b01969575e83c9
.default.svc.cluster.local"
  } ]
}
```

响应示例

状态码： 200

Success

```
{
  "total_peer_num" : 4,
  "success_peer_num" : 4,
  "fail_peer_num" : 0,
  "fail_peers" : [ ]
}
```

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4000013",
  "error_message" : "request body is too large"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

5.3.16 下载报告

功能介绍

下载链代码安全检查报告

URI

POST /v2/agent/apis/chaincode/report

请求参数

表 5-97 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

表 5-98 请求 Body 参数

参数	是否必选	参数类型	描述
chaincode_name	是	String	链代码名称，与安装链代码时填写一致，需开启链代码检查选项
chaincode_version	是	String	链代码版本，与安装链代码时填写一致

响应参数

状态码： 200

表 5-99 响应 Body 参数

参数	参数类型	描述
-	File	Success

状态码： 400

表 5-100 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_message	String	错误描述
error_msg	String	错误描述

请求示例

```
POST https://192.168.0.90:30603/v2/agent/apis/chaincode/report
```

```
{
  "chaincode_name" : "chaincode1",
  "chaincode_version" : "1.0"
}
```

响应示例

状态码： 400

Bad Request

```
{
  "error_code" : "BCS.4002068",
  "error_message" : "failed to get report, not existed"
}
```

状态码

状态码	描述
200	Success
400	Bad Request

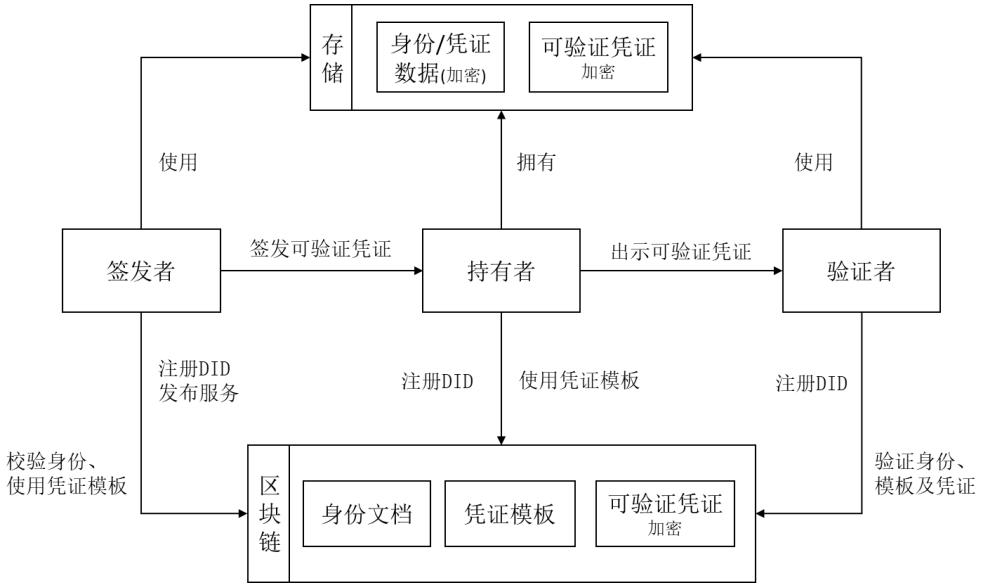
5.4 分布式身份

5.4.1 概述

分布式身份是一种基于区块链的分布式身份管理技术，提供用户身份的创建、可验证凭证的注册、签发、验证等功能，该特性基于W3C分布式身份(Decentralized Identifiers, DIDs)和可验证凭证(Verifiable Credentials, VC)的标准实现，为个人和企业用户提供统一的、可自解释的、移植性强的分布式身份标识，有效解决跨部门、跨企业、跨地域的身份认证难和隐私泄露等问题。

本文为您介绍分布式身份（ DID ）管理的实现流程和使用方式。详细请参见图5-1、图5-2、图5-3和图5-4。

图 5-1 分布式身份实现架构图



实现流程

1. 各个角色可通过企业身份注册(带有service)和注册DID生成完全由自己控制的分布式身份，并将身份文档发布到区块链上完成身份的注册。企业用户的DID身份中可以包含其所能提供的服务信息，以支持多样的应用场景。

   说明

   主要有三种角色：签发者、持有者和验证者，其中每个角色都可以是设备、应用、个人或者组织。

2. 具备基础的身份标识之后，通过可验证凭证架设起身份与身份之间的认证体系。凭证的模板会由相关主体注册发布到区块链上，并持续维护。持有者便可以向签发者发起认证申请，获得凭证后出示给验证者完成校验。
3. 验证者可以通过接口验证持有者出示的“可验证凭证”，确保其是否有权限和资质开展后续业务。

使用方式

分布式身份中间件是部署在用户侧的一套微服务，简化用户调用区块链相关接口的复杂操作。因此接口调用时需要传入用户私钥和被fabric组织根证书签名的证书。

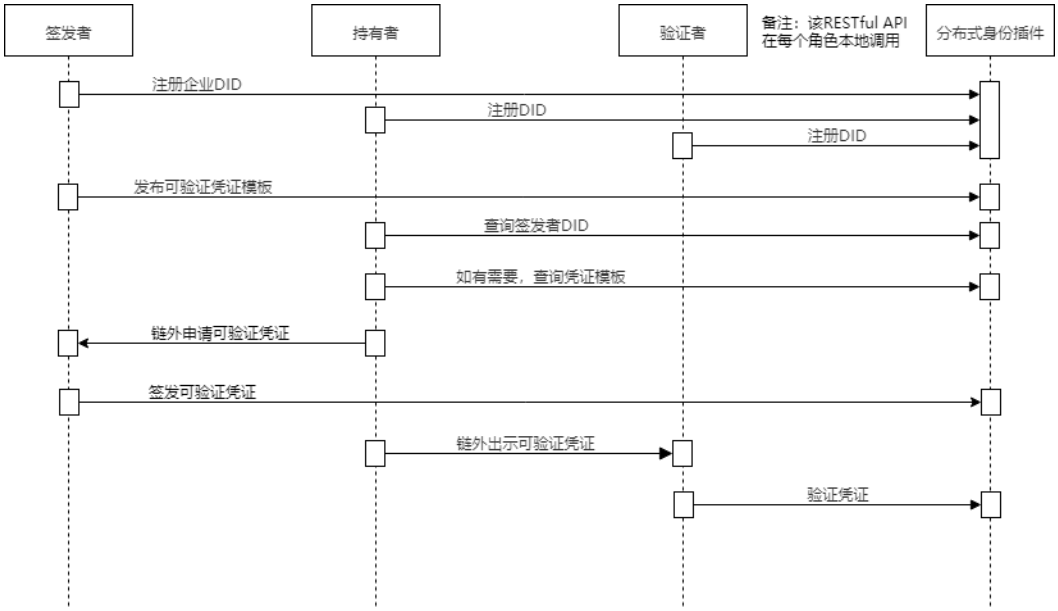
说明

获取用户私钥和证书的方式有两种，BCS区块链管理界面下载和使用openssl工具生成，详细方法请参见《区块链服务用户指南》常见问题中的“获取fabric用户私钥及证书的方法”章节。

根据持有者申请可验证凭证的方式，将分布式身份服务的使用分为链外申请模式和链上申请模式。

- 链外申请模式中，持有者将申请可验证凭证的身份/凭证数据直接发送给签发者。
- 链上申请模式中，持有者将申请可验证凭证的身份/凭证数据加密存储于区块链。

图 5-2 分布式身份使用时序图(链外申请模式)



链上申请模式中，根据持有者与签发者之间是否需要通信信道，分为在线申请和离线申请。

图 5-3 分布式身份使用时序图(链上申请-在线申请模式)

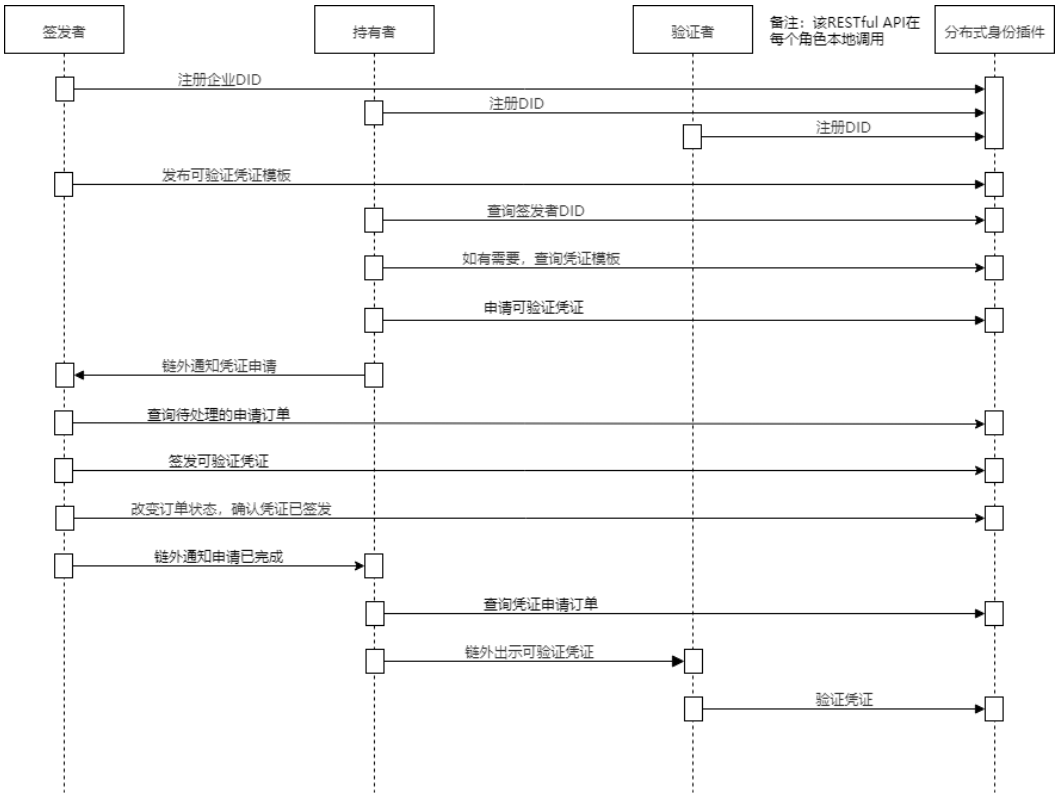
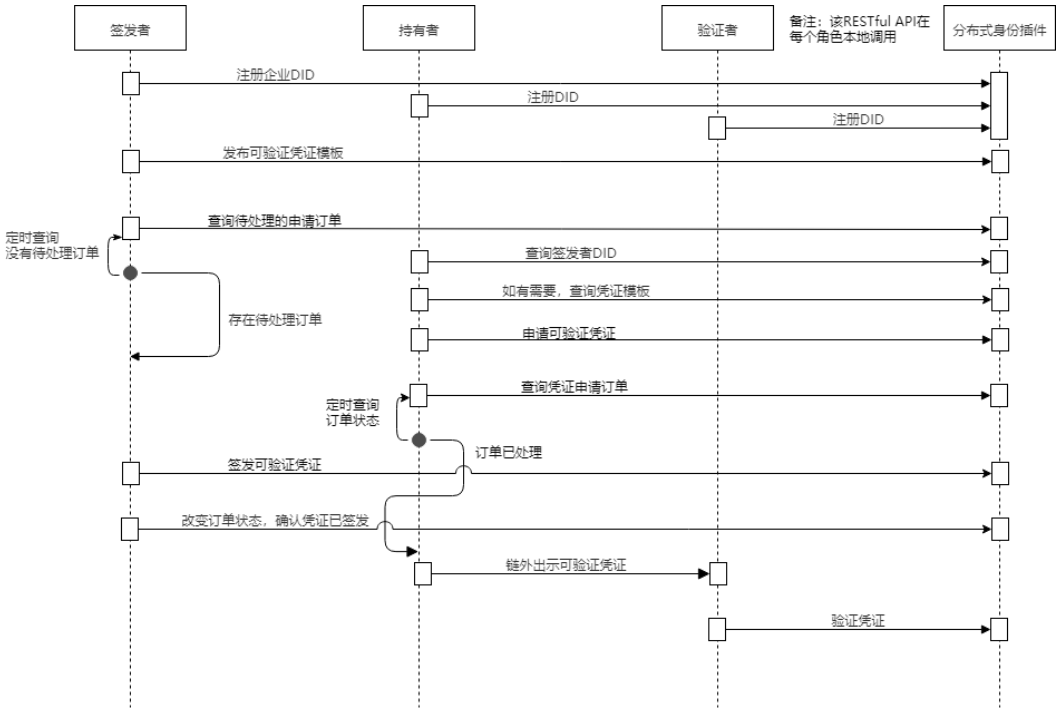


图 5-4 分布式身份使用时序图(链上申请-离线申请模式)



5.4.2 分布式身份(DID)管理

5.4.2.1 企业身份注册(带有 service)

功能介绍

分布式身份注册方法。在使用该方法前需要先使用openssl工具生成每个用户的私钥和被fabric组织根证书签名的证书(或通过BCS区块链管理界面下载用户证书)。注册时需声明可提供的服务列表。

URI

POST /v1/identity/firm-did

请求参数

表 5-101 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n

参数	是否必选	参数类型	描述
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
service	是	Array of **DIDService** objects	提供的服务

表 5-102 DIDService

参数	是否必选	参数类型	描述
type	是	String	类型
serviceEndpoint	是	String	接入点
credentialApplySchema	否	**CredentialApplySchema** object	申请凭证所需数据的Schema

表 5-103 CredentialApplySchema

参数	是否必选	参数类型	描述
type	否	String	类型
name	否	String	名称
description	否	String	描述信息
attributes	否	Array of **Attribute** objects	属性列表

表 5-104 Attribute

参数	是否必选	参数类型	描述
name	否	String	名称
type	否	String	类型
description	否	String	描述信息

响应参数

状态码： 200

表 5-105 响应 Body 参数

参数	参数类型	描述
did	String	分布式身份标识

状态码： 500

表 5-106 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "service" : [ {
    "type" : "VerifiableCredentialService",
    "serviceEndpoint" : "https://example.com/vc/",
    "credentialApplySchema" : {
      "type" : "file",
      "name" : "Test Enterprise Certification",
      "description" : "this is test apply info",
      "attributes" : [ {
        "name" : "bob",
        "type" : "string",
        "description" : "Attribute's description"
      } ]
    }
  } ]
}
```

响应示例

状态码： 200

分布式身份标识

```
{
  "did" : "did:example:2THjdfbKMLDVcoYvkiepr9"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002033",
  "errorMsg" : "Service Type and ServiceEndpoint Can not Null"
}
```

状态码

状态码	描述
200	分布式身份标识
500	失败响应

5.4.2.2 注册 DID

功能介绍

DID快速注册方法。可以方便的注册发布一个DID，该DID不提供服务，只拥有一个公钥。

URI

POST /v1/identity/did

请求参数

表 5-107 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳

响应参数

状态码： 200

表 5-108 响应 Body 参数

参数	参数类型	描述
did	String	分布式身份标识

状态码： 500

表 5-109 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "channel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PUBLIC KEY-----\n...\n...\n-----END PUBLIC KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00"
}
```

响应示例

状态码： 200

分布式身份标识

```
{
  "did" : "did:example:2THjdfbKMLDVcoYvkiepr9"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002033",
  "errorMsg" : "Service Type and ServiceEndpoint Can not Null"
}
```

状态码

状态码	描述
200	分布式身份标识
500	失败响应

5.4.2.3 更新 DID

功能介绍

更新DID文档中发布的服务。

URI

PUT /v1/identity/did

请求参数

表 5-110 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
did	是	String	分布式身份标识
service	否	Array of **DIDService** objects	服务列表

表 5-111 DIDService

参数	是否必选	参数类型	描述
type	是	String	类型

参数	是否必选	参数类型	描述
serviceEndpoint	是	String	接入点
credentialApplySchema	否	**CredentialApplySchema** object	申请凭证所需数据的Schema

表 5-112 CredentialApplySchema

参数	是否必选	参数类型	描述
type	否	String	类型
name	否	String	名称
description	否	String	描述信息
attributes	否	Array of **Attribute** objects	属性列表

表 5-113 Attribute

参数	是否必选	参数类型	描述
name	否	String	名称
type	否	String	类型
description	否	String	描述信息

响应参数

状态码： 200

表 5-114 响应 Body 参数

参数	参数类型	描述
context	String	context
id	String	分布式身份标识
publicKey	Array of **DocPublicKey** objects	公钥列表

参数	参数类型	描述
authentication	Array of strings	did主公钥标识
recovery	String	备用公钥标识，可用于修改主密钥
service	Array of **Service** objects	服务列表
proof	**Proof** object	证明结构，可为空
created	String	创建时间
updated	String	更新时间
status	String	状态

表 5-115 DocPublicKey

参数	参数类型	描述
id	String	公钥标识
type	String	公钥类型
controller	String	公钥的控制者标识
publicKeyPem	String	公钥证书

表 5-116 Service

参数	参数类型	描述
id	String	服务标识
type	String	服务类型
serviceEndpoint	String	服务介绍网址
credentialApplySchema	**CredentialApplySchema** object	申请凭证所需数据的Schema

表 5-117 CredentialApplySchema

参数	参数类型	描述
type	String	类型

参数	参数类型	描述
name	String	名称
description	String	描述信息
attributes	Array of **Attribute** objects	属性列表

表 5-118 Attribute

参数	参数类型	描述
name	String	名称
type	String	类型
description	String	描述信息

表 5-119 Proof

参数	参数类型	描述
creator	String	创建者身份标识
type	String	签名类型
created	String	签名创建时间
signatureValue	String	签名值

状态码： 500

表 5-120 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "did" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "service" : [ {
    "type" : "VerifiableCredentialService",
    "serviceEndpoint" : "https://example.com/vc/",
    "credentialApplySchema" : {
      "type" : "file",
      "name" : "Test Enterprise Certification",
      "description" : "this is test apply info",
      "attributes" : [ {
        "name" : "bob",
        "type" : "string",
        "description" : "Attribute's description"
      } ]
    }
  } ]
}
```

响应示例

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002033",
  "errorMsg" : "Service Type and ServiceEndpoint Can not Null"
}
```

状态码

状态码	描述
200	分布式身份文档结构体
500	失败响应

5.4.2.4 查询 DID

功能介绍

查询指定DID的文档

URI

POST /v1/identity/query-did

请求参数

表 5-121 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
did	是	String	分布式身份标识

响应参数

状态码： 200

表 5-122 响应 Body 参数

参数	参数类型	描述
context	String	context
id	String	分布式身份标识
publicKey	Array of **DocPublicKey** objects	公钥列表
authentication	Array of strings	did主公钥标识
recovery	String	备用公钥标识，可用于修改主密钥
service	Array of **Service** objects	服务列表
proof	**Proof** object	证明结构，可为空
created	String	创建时间
updated	String	更新时间
status	String	状态

表 5-123 DocPublicKey

参数	参数类型	描述
id	String	公钥标识
type	String	公钥类型
controller	String	公钥的控制者标识
publicKeyPem	String	公钥证书

表 5-124 Service

参数	参数类型	描述
id	String	服务标识
type	String	服务类型
serviceEndpoint	String	服务介绍网址
credentialApplySchema	**CredentialApplySchema** object	申请凭证所需数据的Schema

表 5-125 CredentialApplySchema

参数	参数类型	描述
type	String	类型
name	String	名称
description	String	描述信息
attributes	Array of **Attribute** objects	属性列表

表 5-126 Attribute

参数	参数类型	描述
name	String	名称
type	String	类型
description	String	描述信息

表 5-127 Proof

参数	参数类型	描述
creator	String	创建者身份标识
type	String	签名类型
created	String	签名创建时间
signatureValue	String	签名值

状态码： 500

表 5-128 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "did" : "did:example:ebfeb1f712ebc6f1c276e12ec21"
}
```

响应示例

状态码： 200

分布式身份文档结构体

```
{
  "context" : "https://www.w3.org/ns/did/v1",
  "id" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "publicKey" : [ {
    "id" : "string",
    "type" : "string",
    "controller" : "string",
    "publicKeyPem" : "string"
  } ],
  "authentication" : [ "did:example:ebfeb1f712ebc6f1c276e12ec21#key-0" ],
  "recovery" : "string",
```

```
 "service" : [ {
   "id" : "did:example:ebfeb1f712ebc6f1c276e12ec21#xdi",
   "type" : "XdiService",
   "serviceEndpoint" : "https://xdi.example.com/8377464",
   "credentialApplySchema" : {
     "type" : "LegalCitizen",
     "name" : "LegalCitizen",
     "description" : "Certified citizens",
     "attributes" : [ {
       "name" : "name",
       "type" : "someType",
       "description" : "Identity number"
     } ]
   }
 } ],
 "proof" : {
   "creator" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
   "type" : "RsaSignature2018",
   "created" : "1606720551",
   "signatureValue" : "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0I
PAYuNzVBAh4vGHSrQyHUdBBPM"
 },
 "created" : "1588921521",
 "updated" : "",
 "status" : "active"
}
```


状态码： 500

失败响应


```
{
  "errorCode" : "BCS.5002034",
  "errorMsg" : "request timed out or been cancelled"
}
```


状态码

状态码	描述
200	分布式身份文档结构体
500	失败响应

5.4.3 可验证凭证(VC)管理

5.4.3.1 发布可验证凭证的模板

功能介绍

发布凭证模板数据结构。签发professional类型的凭证时会使用，因为字段较多容易不统一，需要提前有模板来约束和标明。

URI

POST /v1/identity/credential-schema

请求参数

表 5-129 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
title	否	String	名称
identifier	是	String	标识
attributes	否	Array of **Attribute** objects	属性信息
issuer	是	String	签发者身份标识

表 5-130 Attribute

参数	是否必选	参数类型	描述
name	否	String	名称
type	否	String	类型
description	否	String	描述信息

响应参数

状态码： 200

表 5-131 响应 Body 参数

参数	参数类型	描述
schemaIndex	String	模板存储在链上的索引
credentialSchema	**CredentialSchema** object	CredentialSchema

表 5-132 CredentialSchema

参数	参数类型	描述
creator	String	创建者身份标识
title	String	名称
identifier	String	凭证模板标识
attributes	Array of **Attribute** objects	属性信息
version	Integer	版本

表 5-133 Attribute

参数	参数类型	描述
name	String	名称
type	String	类型
description	String	描述信息

状态码： 500

表 5-134 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "title" : "string",
  "identifier" : "string",
  "attributes" : [ {
```

```
    "name" : "name",
    "type" : "someType",
    "description" : "Identity number"
  } ],
  "issuer" : "did:example:ebfeb1f712ebc6f1c276e12ec21"
}
```

响应示例

状态码： 200

VCSchemaResponeParams

```
{
  "schemaIndex" : "string",
  "credentialSchema" : {
    "creator" : "string",
    "title" : "string",
    "identifier" : "string",
    "attributes" : [ {
      "name" : "name",
      "type" : "someType",
      "description" : "Identity number"
    } ],
    "version" : 0
  }
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002035",
  "errorMsg" : "Schema Already Exist"
}
```

状态码

状态码	描述
200	VCSchemaResponeParams
500	失败响应

5.4.3.2 查询凭证模板

功能介绍

根据索引查询凭证模板。

URI

POST /v1/identity/query-credential-schema

请求参数

表 5-135 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
schemaIndex	是	String	schema索引

响应参数

状态码： 200

表 5-136 响应 Body 参数

参数	参数类型	描述
creator	String	创建者身份标识
title	String	名称
identifier	String	凭证模板标识
attributes	Array of **Attribute** objects	属性信息
version	Integer	版本

表 5-137 Attribute

参数	参数类型	描述
name	String	名称
type	String	类型
description	String	描述信息

状态码： 500

表 5-138 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "schemaIndex" : "1"
}
```

响应示例

状态码： 200

CredentialSchema Information

```
{
  "creator" : "string",
  "title" : "string",
  "identifier" : "string",
  "attributes" : [ {
    "name" : "name",
    "type" : "someType",
    "description" : "Identity number"
  } ],
  "version" : 0
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	CredentialSchema Information

状态码	描述
500	失败响应

5.4.3.3 申请可验证凭证

功能介绍

申请可验证凭证。根据在服务中声明的必要字段，申请者需要提供相关数据。

URI

POST /v1/identity/apply-vc

请求参数

表 5-139 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
applyer	是	String	申请者的身份标识
serviceID	是	String	服务的标识符。服务提供者did中声明的service的id
data	否	String	data

响应参数

状态码： 200

表 5-140 响应 Body 参数

参数	参数类型	描述
orderIndex	String	订单索引

状态码： 500

表 5-141 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "applyer" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "serviceID" : "did:example:ebfeb1f712ebc6f1c276e12ec21#service1",
  "data" : "abcdefg"
}
```

响应示例

状态码： 200

VCOrderResponeParams Information

```
{
  "orderIndex" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	VCOrderResponeParams Information
500	失败响应

5.4.3.4 签发者确认凭证已签发

功能介绍

签发者确认申请订单，将签发的凭证索引更新到订单中。

URI

POST /v1/identity/vc-order

请求参数

表 5-142 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
orderIndex	是	String	订单索引
vcIndex	是	String	凭证索引

响应参数

状态码： 200

表 5-143 响应 Body 参数

参数	参数类型	描述
orderIndex	String	订单索引
result	String	结果

状态码： 500

表 5-144 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "orderIndex" : 1,
  "vcIndex" : 0
}
```

响应示例

状态码： 200

ConfirmOrderResponeParams Information

```
{
  "orderIndex" : "string",
  "result" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	ConfirmOrderResponeParams Information
500	失败响应

5.4.3.5 查询凭证申请订单

功能介绍

查询凭证的申请订单

URI

POST /v1/identity/query-vc-order

请求参数

表 5-145 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
orderIndex	是	String	订单索引

响应参数

状态码： 200

表 5-146 响应 Body 参数

参数	参数类型	描述
applyer	String	申请者身份标识
orderSeq	String	订单序列号
applyTime	String	申请时间
price	String	价格
status	String	状态
service	String	服务
reason	String	理由信息

参数	参数类型	描述
dataUri	String	凭证数据URI
encryptedAesKey	String	加密的AES密钥
uriType	String	URI类型
dataHash	String	数据的hash
lockProof	String	锁定证明
vcIndex	String	凭证索引

状态码： 500

表 5-147 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "orderIndex" : 1
}
```

响应示例

状态码： 200

VCOrder Information

```
{
  "applyer" : "string",
  "orderSeq" : "string",
  "applyTime" : "string",
  "price" : "string",
  "status" : "string",
  "service" : "string",
  "reason" : "string",
```

```
  "dataUri" : "string",
  "encryptedAesKey" : "string",
  "uriType" : "string",
  "dataHash" : "string",
  "lockProof" : "string",
  "vcIndex" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	VCOrder Information
500	失败响应

5.4.3.6 查询待处理的申请订单

功能介绍

根据服务标识符，查询待处理的凭证申请订单，仅有服务提供者有权限

URI

POST /v1/identity/query-vc-orders

请求参数

表 5-148 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
serviceID	否	String	服务的标识符

响应参数

状态码： 200

表 5-149 响应 Body 参数

参数	参数类型	描述
items	Array of **VCOrder** objects	列表

表 5-150 VCOrder

参数	参数类型	描述
applyer	String	申请者身份标识
orderSeq	String	订单序列号
applyTime	String	申请时间
price	String	价格
status	String	状态
service	String	服务
reason	String	理由信息
dataUri	String	凭证数据URI
encryptedAesKey	String	加密的AES密钥
uriType	String	URI类型
dataHash	String	数据的hash
lockProof	String	锁定证明
vcIndex	String	凭证索引

状态码： 500

表 5-151 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "orderIndex" : 1
}
```

响应示例

状态码： 200

UntreatedVCOrder Information List

```
{
  "items" : [ {
    "applyer" : "string",
    "orderSeq" : "string",
    "applyTime" : "string",
    "price" : "string",
    "status" : "string",
    "service" : "string",
    "reason" : "string",
    "dataUri" : "string",
    "encryptedAesKey" : "string",
    "uriType" : "string",
    "dataHash" : "string",
    "lockProof" : "string",
    "vcIndex" : "string"
  } ]
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	UntreatedVCOrder Information List
500	失败响应

5.4.3.7 签发可验证凭证

功能介绍

签发可验证凭证,签发时间默认为当前时间。

URI

POST /v1/identity/issue-vc

请求参数

表 5-152 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
credentialInfo	是	CredentialSubjectInfo object	凭证主题信息

表 5-153 CredentialSubjectInfo

参数	是否必选	参数类型	描述
applyer	是	String	申请者身份标识
issuer	是	String	签发者身份标识
sequence	是	String	凭证的序列号

参数	是否必选	参数类型	描述
schemaIndex	是	String	凭证模板索引
expirationDate	否	String	过期时间
data	是	String	凭证数据明文

响应参数

状态码： 200

表 5-154 响应 Body 参数

参数	参数类型	描述
vcIndex	String	凭证索引

状态码： 500

表 5-155 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "org1",
  "channelID" : "mychannel",
  "cryptoMethod" : "this is a demo",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "applyer" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "issuer" : "did:example:fdsafre767f8a3hr773j4h1jkhr",
  "sequence" : "10025469331",
  "schemaIndex" : "did:example:ebfeb1f712ebc6f1c276e12ec21_IDCard",
  "data" : "{\"name\": \"xm\",\"age\": 18}"
}
```

响应示例

状态码： 200

VCResponeParams Information

```
{
  "vcIndex" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	VCResponeParams Information
500	失败响应

5.4.3.8 根据索引查询可验证凭证

功能介绍

根据索引查询可验证凭证

URI

POST /v1/identity/query-vc

请求参数

表 5-156 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n

参数	是否必选	参数类型	描述
timestamp	是	String	时间戳
vcIndex	是	String	凭证索引

响应参数

状态码： 200

表 5-157 响应 Body 参数

参数	参数类型	描述
context	String	内容
sequence	String	颁发机构对应凭证的序列号
type	Array of strings	可验证凭证类型
issuer	String	签发者身份标识
issuanceDate	String	签发日期
expirationDate	String	凭证有效期
credentialSubject	**CredentialSubject** object	凭证主题
revocation	**Revocation** object	撤销

表 5-158 CredentialSubject

参数	参数类型	描述
owner	String	申请者的身份标识
type	String	凭证类型
schemaID	String	schema ID
dataURI	String	数据URI
encryptedAeskey	String	加密对称密钥
uriType	String	数据索引类型
dataHash	String	数据hash值

表 5-159 Revocation

参数	参数类型	描述
id	String	撤销API或者撤销列表的url
type	String	撤销类型

状态码： 500

表 5-160 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "vcIndex" : 0
}
```

响应示例

状态码： 200

VerifiableCredential Information

```
{
  "context" : "https://www.w3.org/2018/credentials/v1",
  "sequence" : "x00123456",
  "type" : [ "VerifiableCredential", "AlumniCredential" ],
  "issuer" : "https://example.edu/issuers/565049",
  "issuanceDate" : "1606720551",
  "expirationDate" : "1606720551",
  "credentialSubject" : {
    "owner" : "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "type" : "professional",
    "schemaID" : "did:example:ebfeb1f712ebc6f1c276e12ec21_IDCard",
    "dataURI" : "string",
    "encryptedAeskey" : "string",
    "uriType" : "index",
    "dataHash" : "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b85"
  },
  "revocation" : {
```

```
    "id" : "string",
    "type" : "string"
  }
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "stringst",
  "errorMsg" : "string"
}
```

状态码

状态码	描述
200	VerifiableCredential Information
500	失败响应

5.4.3.9 验证凭证

功能介绍

验证可验证凭证是否存在以及合法性。

URI

POST /v1/identity/verify-vc

请求参数

表 5-161 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	用户证书，每行末尾均需要增加显式换行符\n
sk	是	String	用户私钥，每行末尾均需要增加显式换行符\n
timestamp	是	String	时间戳
vcIndex	是	String	凭证索引
owner	是	String	凭证所有者身份标识

响应参数

状态码： 200

表 5-162 响应 Body 参数

参数	参数类型	描述
isOwned	Boolean	是否拥有

状态码： 400

表 5-163 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

状态码： 500

表 5-164 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码 最小长度：**8** 最大长度：**36**
errorMsg	String	错误描述 最小长度：**2** 最大长度：**512**

请求示例

```
{
  "orgID" : "4f1439758ebb41f7411b5f684b67713c08b89198",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\n...\n...\n-----END CERTIFICATE-----",
  "sk" : "------BEGIN PRIVATE KEY-----\n...\n...\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "vcIndex" : 1,
```

```
  "owner" : "did:example:ebfeb1f712ebc6f1c276e12ec21"
}
```

响应示例

状态码： 200

VCVerifyResponeParams Information

```
{
  "isOwned" : true
}
```

状态码： 400

失败响应

```
{
  "errorCode" : "BCS.4002030",
  "errorMsg" : "Owner(bol) does not have credential ..."
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002014",
  "errorMsg" : "Internal Server Error"
}
```

状态码

状态码	描述
200	VCVerifyResponeParams Information
400	失败响应
500	失败响应

5.5 可信数据交换

5.5.1 概述

在商业实践中，数据是重要的生产要素。基于区块链的可信数据交换，实现了分布式场景中业务数据的隐私保护与可信共享，有效打破“数据孤岛”，最大化数据价值。可信数据共享中间件集成在Rest API插件中，可快速插拔，支持弹性伸缩。用户可通过RESTful API的方式访问区块链系统，快速集成，实现数据的发布、授权、分享、加密、解密、细粒度访问控制等能力。

功能介绍

- 可信数据涉及两个主要数据结构：数据集和数据订单。数据集包括数据描述信息、访问控制信息（属性加密中的策略）等。数据订单包括数据申请、审核信息等。

- 可信数据交换支持三种模式：申请-授权、主动分享和细粒度访问控制。详细请参见**模式介绍**。

  说明

  可信交换中的数据加密后支持多种存储服务，用户可以根据业务需要自己选择。调用者负责将密文数据存储到公开可访问的存储设备中。

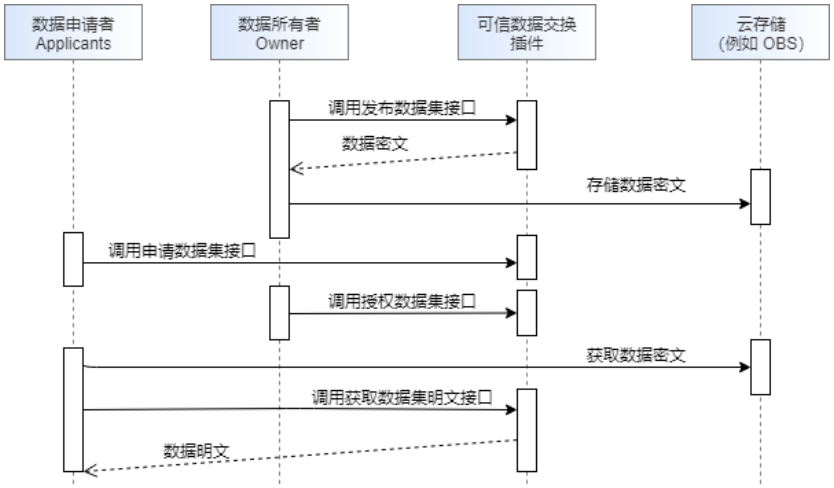
角色定义

数据所有者Owner和数据申请者Applicants，每一个用户既可以是数据所有者角色，也可以是数据申请者角色。

模式介绍

- 申请-授权模式，授权流程图请参见**图5-5**。
  - 数据所有者通过“发布数据集”接口完成用户明文数据的加密和数据描述等信息的注册发布。
  - 数据申请者可通过“申请数据集”接口调用链代码触发申请-授权流程。
  - 数据所有者可根据申请信息和申请者的did、vc等信息决定授权或者拒绝。

  图 5-5 申请-授权模式使用流程

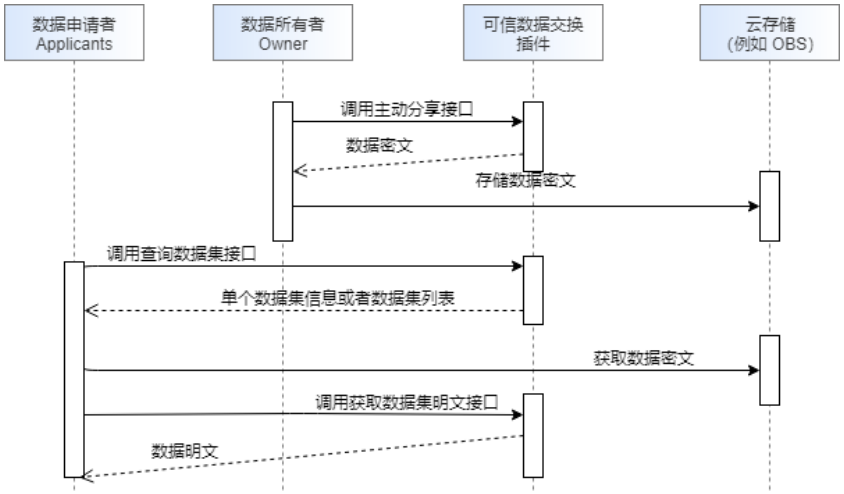
  

- 主动分享模式，使用流程请参见**图5-6**。

  “主动分享数据集”接口相当于“发布数据集”接口和“授权数据集”接口的组合。数据所有者将数据集发布到区块链，同时授权某申请者解密数据权限，被授权者可以直接解密数据集。此时，其他参与者均可以通过“查询指定数据集”和“查询数据集列表”接口获得数据相关描述信息，并通过申请-授权模式获取数据解密权限。

  接口使用方法请参考数据集管理和数据订单管理。

图 5-6 主动分享模式使用流程



- 细粒度访问控制模式，使用流程请参见图5-7。

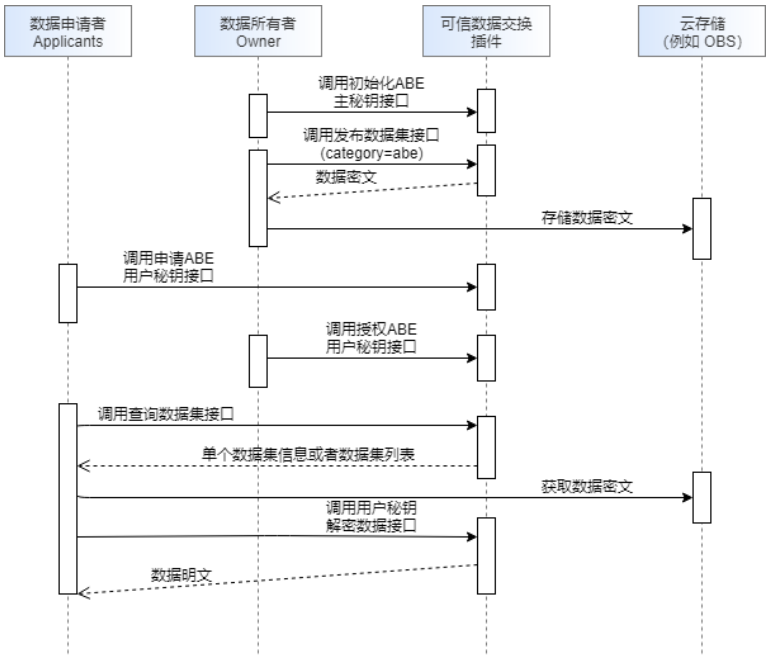
  说明

  基于属性加密(Attribute-Based Encryption, ABE)细粒度控制的数据交换模式，通过为每个数据集配置合理的、自定义的共享策略，实现对数据申请者属性级别的访问控制，即拥有某些属性组合的申请者可以访问密文。

  细粒度访问控制模式采用CP-ABE（基于密文策略的属性加密，Ciphertext-Policy ABE）方式实现。将策略嵌入到密文中，属性嵌入到用户密钥中。

  - 每个数据所有者都只需初始化一次自己的主公钥和私钥。
  - 数据申请者需要使用某数据所有者数据时，需要向其申请用户密钥，当属性没有变化的情况下只需申请一次。
  - 当拥有用户密钥且属性满足密文访问策略时，数据申请者可以异步的，随时解密数据所有者发布的所有相应数据。
  - 接口使用方法请参考属性加密的密钥管理。

图 5-7 细粒度访问控制模式使用流程



说明

基本概念介绍：

- 属性（Attribute）：描述一个实体的性质与实体之间关系的统称。
- 策略（Policy)：策略是属性集合与逻辑关系的组合，由数据所有者制定，将嵌入到数据密文中。例如 “age>26 && gender=man”代表策略要求年龄大于26岁且性别是男性。
- ABE主密钥包含主公钥（Master Public Key, MPK）和主私钥（Master Secret Key, MSK）：数据拥有者所有，用于加密密文和生成用户（数据申请者）密钥。
- 用户密钥（User Key）: 数据申请者通过提交属性列表，向数据所有者申请获得。密钥中包含用户的属性信息，用于解密密文。

5.5.2 数据集管理

5.5.2.1 发布数据集

功能介绍

发布数据集。（不支持水印功能）

URI

POST /v1/datashare/dataset

请求参数

表 5-165 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
provider	是	String	数据发布者身份标识
providerName	否	String	数据发布者名称
productName	是	String	产品名称
productID	是	String	产品ID
sampleUrl	是	String	样例数据存放地址
sampleSize	否	String	样例数据大小

参数	是否必选	参数类型	描述
sampleType	否	String	样例数据类型
sampleName	否	String	样例数据名称
fileType	否	String	数据文件类型
dataUrl	是	String	数据存放地址
dataHash	否	String	数据哈希
dataSize	否	String	数据大小
dataName	否	String	数据名称
description	否	String	数据集描述信息，在使用abe加密时，需要对加密策略进行详细描述
plainData	是	String	base64编码的明文数据
category	否	String	加密类型，abe/symmetric，默认为symmetric，如果使用abe加密，则policy必填
watermarkType	否	String	水印类型，visible明水印，blind暗水印，嵌入水印时，必须填写；不填写时默认不嵌入水印。嵌入的水印内容为：发布人did_productID。
file	否	File	加水印的文件，当对文件加水印时，plainData无效果
policy	否	**policy** object	abe策略

表 5-166 policy

参数	是否必选	参数类型	描述
Threshold	是	Integer	策略需要满足的属性阈值
Children	是	Array of **policy-children** objects	子属性列表

表 5-167 policy-children

参数	是否必选	参数类型	描述
name	是	String	属性名

参数	是否必选	参数类型	描述
type	是	String	属性类型（ plain，comparable和policy ）
value	是	policy-children-comparablevalue object	属性值，根据type确定具体参数。 当type为plain时，value为string类型属性值。 当type为policy时，value为子policy。 当type为comparable时，value包含了op、value和maxValue三部分。

表 5-168 policy-children-comparablevalue

参数	是否必选	参数类型	描述
op	否	String	type为comparable时填写，比较类型符号（ >,<或= ）
value	是	String	比较类型属性值，必须为整数
maxValue	否	String	type为comparable时填写，比较类型中value上限值

响应参数

状态码： 200

表 5-169 响应 Body 参数

参数	参数类型	描述
provider	String	数据集提供者身份标识
providerName	String	数据集提供者名称
productName	String	数据集产品名称
productID	String	数据集产品id
sampleUrl	String	样例数据url
sampleSize	String	样例数据大小
sampleType	String	样例数据类型
sampleName	String	样例数据名称
fileType	String	文件类型
dataUrl	String	数据url

参数	参数类型	描述
dataHash	String	数据哈希值
dataSize	String	数据大小
dataName	String	数据名称
description	String	数据描述
price	String	数据价格
encryptedAesKey	String	密钥
status	String	状态
publishTime	String	数据发布时间
dataFiles	Array of **DataFile** objects	数据文件列表
sampleFiles	Array of **DataFile** objects	样例文件列表
category	String	加密类型
encryptData	String	加密后的数据

表 5-170 DataFile

参数	参数类型	描述
fileType	String	文件类型
dataUrl	String	数据url
dataHash	String	数据哈希
dataSize	String	数据大小
dataName	String	数据名称

状态码： 500

表 5-171 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

abe示例策略（policy）。该示例策略中，解密者需要符合属性att1=="hello"或符合子策略才能解密；解密者需要属性att3的值大于16才能符合子策略。

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productName" : "prodname",
  "productID" : "product2",
  "sampleUrl" : "http://sdcloud.com/sample.com/prodname2",
  "sampleSize" : "10KB",
  "sampleType" : "csv",
  "sampleName" : "data_sub1",
  "fileType" : "csv",
  "dataUrl" : "http://sdcloud.com/prodname2",
  "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
  "dataSize" : "100MB",
  "dataName" : "mydata1",
  "description" : "this is my second prod",
  "plainData" : "base64 encoding string",
  "category" : "abe",
  "watermarkType" : "string",
  "file" : "string",
  "policy" : "{\"threshold\" : 1,\"children\" : [{\"name\" : \"att1\", \"type\" : \"plain\",\"value\" : \"hello\"},
{\"name\" : \"policy\", \"type\" : \"policy\",\"value\" : {\"Threshold\" : 1,\"Children\" : [{\"name\" : \"att3\",
\"type\" :\"comparable\", \"value\" : {\"op\" : \">\", \"value\" : \"16\", \"maxValue\" : \"10000\"}}]}}]}"
}
```

响应示例

状态码： 200

数据集信息

```
{
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "xxx",
  "productName" : "prodname2",
  "productID" : "product2",
  "sampleUrl" : "http://sdcloud.com/sample.com/prodname2",
  "sampleSize" : "10KB",
  "sampleType" : "csv",
  "sampleName" : "data_sub1",
  "fileType" : "csv",
  "dataUrl" : "http://sdcloud.com/prodname2",
  "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
  "dataSize" : "100MB",
  "dataName" : "mydata",
  "description" : "this is second prod",
  "price" : "0",
  "encryptedAesKey" : "BA4Ub3t3IskN8uKcEMa+4cbtsDS8OzF4V/qqb4OcPMeMvp7IL+HClzAbL6lPnhbDg/
AnrStBlf0qFzRj+qvk6ZH0c7wP0aS48fSoNtecG79aFpFx0dg7rFdVYXWWzgeyI03eD3gFdXlQ/
ovpxKJG5ALK39OCazUqDrawZHSDGyIlw0hGh88Q+GVORVSp+6V5Ag==",
  "status" : "ready",
  "publishTime" : "1607157244",
  "dataFiles" : [ {
    "fileType" : "csv",
    "dataUrl" : "http://sdcloud.com/prodname2",
    "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
    "dataSize" : "100MB",
    "dataName" : "mydata"
```



```
 } ],
 "sampleFiles" : [ {
   "fileType" : "csv",
   "dataUrl" : "http://sdcloud.com/prodname2",
   "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
   "dataSize" : "100MB",
   "dataName" : "mydata"
 } ],
 "category" : "string",
 "encryptData" : "string"
}
```

状态码： 500

失败响应

```
{
 "errorCode" : "BCS.5002046",
 "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	数据集信息
500	失败响应

5.5.2.2 删除数据集

功能介绍

删除数据集

URI

DELETE /v1/datashare/dataset

请求参数

表 5-172 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳

参数	是否必选	参数类型	描述
provider	是	String	数据集提供者身份标识
productID	是	String	数据产品id

响应参数

状态码： 200

表 5-173 响应 Body 参数

参数	参数类型	描述
result	String	操作结果

状态码： 500

表 5-174 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "productID" : "product1"
}
```

响应示例

状态码： 200

操作结果

```
{
  "result" : "success"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
```

```
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	操作结果
500	失败响应

5.5.2.3 关闭数据集

功能介绍

关闭数据集

URI

PUT /v1/datashare/dataset

请求参数

表 5-175 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
provider	是	String	数据集提供者身份标识
productID	是	String	数据产品id

响应参数

状态码： 200

表 5-176 响应 Body 参数

参数	参数类型	描述
result	String	操作结果

状态码： 500

表 5-177 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "productID" : "product1"
}
```

响应示例

状态码： 200

操作结果

```
{
  "result" : "success"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	操作结果
500	失败响应

5.5.2.4 查询指定数据集

功能介绍

查询指定数据集，根据数据集发布者身份标识和数据产品id信息查询。

URI

POST /v1/datashare/query-dataset

请求参数

表 5-178 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
provider	是	String	数据集发布者身份标识
productID	是	String	数据集产品id

响应参数

状态码： 200

表 5-179 响应 Body 参数

参数	参数类型	描述
provider	String	数据集提供者身份标识
providerName	String	数据集提供者名称
productName	String	数据集产品名称
productID	String	数据集产品id
sampleUrl	String	样例数据url
sampleSize	String	样例数据大小
sampleType	String	样例数据类型
sampleName	String	样例数据名称

参数	参数类型	描述
fileType	String	文件类型
dataUrl	String	数据url
dataHash	String	数据哈希值
dataSize	String	数据大小
dataName	String	数据名称
description	String	数据描述
price	String	数据价格
encryptedAesKey	String	密钥
status	String	状态
publishTime	String	数据发布时间
dataFiles	Array of **DataFile** objects	数据文件列表
sampleFiles	Array of **DataFile** objects	样例文件列表
category	String	加密类型

表 5-180 DataFile

参数	参数类型	描述
fileType	String	文件类型
dataUrl	String	数据url
dataHash	String	数据哈希
dataSize	String	数据大小
dataName	String	数据名称

状态码： 500

表 5-181 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码

参数	参数类型	描述
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "productID" : "product2"
}
```

响应示例

状态码： 200

操作结果

```
{
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "xxx",
  "productName" : "prodname2",
  "productID" : "product2",
  "sampleUrl" : "http://sdcloud.com/sample.com/prodname2",
  "sampleSize" : "10KB",
  "sampleType" : "csv",
  "sampleName" : "data_sub1",
  "fileType" : "csv",
  "dataUrl" : "http://sdcloud.com/prodname2",
  "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
  "dataSize" : "100MB",
  "dataName" : "mydata",
  "description" : "this is second prod",
  "price" : "0",
  "encryptedAesKey" : "BA4Ub3t3IskN8uKcEMa+4cbtsDS8OzF4V/qqb4OcPMeMvp7IL+HClzAbL6lPnhbDg/
AnrStBlf0qFzRj+qvk6ZH0c7wP0aS48fSoNtecG79aFpFx0dg7rFdVYXWWzgeyI03eD3gFdXlQ/
ovpxKJG5ALK39OCazUqDrawZHSDGyIlw0hGh88Q+GVORVSp+6V5Ag==",
  "status" : "ready",
  "publishTime" : "1607157244",
  "dataFiles" : [ {
    "fileType" : "csv",
    "dataUrl" : "http://sdcloud.com/prodname2",
    "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
    "dataSize" : "100MB",
    "dataName" : "mydata"
  } ],
  "sampleFiles" : [ {
    "fileType" : "csv",
    "dataUrl" : "http://sdcloud.com/prodname2",
    "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
    "dataSize" : "100MB",
    "dataName" : "mydata"
  } ],
  "category" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	操作结果
500	失败响应

5.5.2.5 查询数据集列表

功能介绍

查询数据集，支持分页和按条件过滤查询

URI

POST /v1/datashare/query-datasets

请求参数

表 5-182 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
currentPage	否	String	分页参数：当前页码（默认1）
pageSizeNum	否	String	分页参数：每页条数(默认100)
provider	否	String	过滤条件：发布者身份标识
searchText	否	String	过滤条件：匹配关键字（基于数据集中的产品名称、产品描述信息）
status	否	String	过滤条件：数据集状态（ready、closed）

响应参数

状态码： 200

表 5-183 响应 Body 参数

参数	参数类型	描述
items	Array of **DatasetResponse** objects	列表
pagination	**PaginationResp** object	分页信息

表 5-184 DatasetResponse

参数	参数类型	描述
provider	String	数据集提供者身份标识
providerName	String	数据集提供者名称
productName	String	数据集产品名称
productID	String	数据集产品id
sampleUrl	String	样例数据url
sampleSize	String	样例数据大小
sampleType	String	样例数据类型
sampleName	String	样例数据名称
fileType	String	文件类型
dataUrl	String	数据url
dataHash	String	数据哈希值
dataSize	String	数据大小
dataName	String	数据名称
description	String	数据描述
price	String	数据价格
encryptedAesKey	String	密钥
status	String	状态
publishTime	String	数据发布时间

参数	参数类型	描述
dataFiles	Array of **DataFile** objects	数据文件列表
sampleFiles	Array of **DataFile** objects	样例文件列表
category	String	加密类型

表 5-185 DataFile

参数	参数类型	描述
fileType	String	文件类型
dataUrl	String	数据url
dataHash	String	数据哈希
dataSize	String	数据大小
dataName	String	数据名称

表 5-186 PaginationResp

参数	参数类型	描述
currentPage	Integer	当前页码
pageSizeNum	Integer	每页条数
totalItems	Integer	总条数

状态码： 500

表 5-187 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
```

```
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "currentPage" : "string",
  "pageSizeNum" : "string",
  "provider" : "string",
  "searchText" : "string",
  "status" : "string"
}
```

响应示例

状态码： 200

数据集分页信息

```
{
  "items" : [ {
    "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
    "providerName" : "xxx",
    "productName" : "prodname2",
    "productID" : "product2",
    "sampleUrl" : "http://sdcloud.com/sample.com/prodname2",
    "sampleSize" : "10KB",
    "sampleType" : "csv",
    "sampleName" : "data_sub1",
    "fileType" : "csv",
    "dataUrl" : "http://sdcloud.com/prodname2",
    "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
    "dataSize" : "100MB",
    "dataName" : "mydata",
    "description" : "this is second prod",
    "price" : "0",
    "encryptedAesKey" : "BA4Ub3t3IskN8uKcEMa+4cbtsDS8OzF4V/qqb4OcPMeMvp7IL+HClzAbL6lPnhbDg/
AnrStBlf0qFzRj+qvk6ZH0c7wP0aS48fSoNtecG79aFpFx0dg7rFdVYXWWzgeyI03eD3gFdXlQ/
ovpxKJG5ALK39OCazUqDrawZHSDGyIlw0hGh88Q+GVORVSp+6V5Ag==",
    "status" : "ready",
    "publishTime" : "1607157244",
    "dataFiles" : [ {
      "fileType" : "csv",
      "dataUrl" : "http://sdcloud.com/prodname2",
      "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
      "dataSize" : "100MB",
      "dataName" : "mydata"
    } ],
    "sampleFiles" : [ {
      "fileType" : "csv",
      "dataUrl" : "http://sdcloud.com/prodname2",
      "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
      "dataSize" : "100MB",
      "dataName" : "mydata"
    } ],
    "category" : "string"
  } ],
  "pagination" : {
    "currentPage" : 1,
    "pageSizeNum" : 100,
    "totalItems" : 10
  }
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
```



```
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	数据集分页信息
500	失败响应

5.5.2.6 主动分享数据集

功能介绍

发布数据集并创建已授权的订单。（ 不支持水印功能 ）

URI

POST /v1/datashare/dataset/share

请求参数

表 5-188 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
provider	是	String	数据发布者身份标识
providerName	否	String	数据发布者名称
productName	是	String	产品名称
productID	是	String	产品ID
sampleUrl	是	String	样例数据存放地址
sampleSize	否	String	样例数据大小
sampleType	否	String	样例数据类型
sampleName	否	String	样例数据名称

参数	是否必选	参数类型	描述
fileType	否	String	数据文件类型
dataUrl	是	String	数据存放地址
dataHash	否	String	数据哈希
dataSize	否	String	数据大小
dataName	否	String	数据名称
description	否	String	数据集描述信息，在使用abe加密时，需要对加密策略进行详细描述
plainData	是	String	base64编码的明文数据
consumer	是	String	订单申请者身份标识
orderSeq	否	String	订单序列号，为空则随机生成
watermarkType	否	String	水印类型，visible明水印，blind暗水印，嵌入水印时，必须填写；不填写时默认不嵌入水印。嵌入的水印内容为：发布人did_productID。
file	否	String	加水印的文件，当对文件加水印时，plainData无效果
productIDKeywords	否	Array of **productIDKeywordsJson** objects	产品ID包含的索引关键字，用于按条件查询订单时使用，需要为设定的json格式
onChainStore	否	String	是否在链上存储加密后的数据，可设置为“true”或“false”，默认为“false”。当为“true”时，sampleUrl和dataUrl可以自动填充，不需要填写。水印功能暂不支持链上存储。
consumerName	否	String	使用者名称
creatorDID	否	String	数据集分享流程的创建者DID，action为“new”时，不需要填入
processID	否	String	数据集分享流程的流程ID，action为"new"时，不需要输入
stageName	否	String	本次数据集分享阶段的阶段名称
action	否	String	流程动作，新建："new"、追加："append"、不涉及：""，该字段不输入时视为不涉及

参数	是否必选	参数类型	描述
category	否	String	加密类型，symmetric/abe，默认为symmetric，如果使用abe加密，则policy必填
policy	否	**policy** object	abe策略

表 5-189 productIDKeywordsJson

参数	是否必选	参数类型	描述
value	否	String	关键字值

表 5-190 policy

参数	是否必选	参数类型	描述
Threshold	是	Integer	策略需要满足的属性阈值
Children	是	Array of **policy-children** objects	子属性列表

表 5-191 policy-children

参数	是否必选	参数类型	描述
name	是	String	属性名
type	是	String	属性类型（plain，comparable和policy）
value	是	**policy-children-comparablevalue** object	属性值，根据type确定具体参数。 当type为plain时，value为string类型属性值。 当type为policy时，value为子policy。 当type为comparable时，value包含了op、value和maxValue三部分。

表 5-192 policy-children-comparablevalue

参数	是否必选	参数类型	描述
op	否	String	type为comparable时填写，比较类型符号（ >,<或= ）
value	是	String	比较类型属性值，必须为整数
maxValue	否	String	type为comparable时填写，比较类型中value上限值

响应参数

状态码： 200

表 5-193 响应 Body 参数

参数	参数类型	描述
consumer	String	订单消费者身份标识
consumerName	String	订单消费者名称
orderSeq	String	订单序列号
provider	String	订单提供者身份标识
providerName	String	订单提供者名称
productID	String	数据集产品id
productName	String	数据集产品名称
price	String	订单价钱
applyTime	String	订单申请时间
encryptedAesKey	String	密钥
status	String	订单状态
reason	String	订单申请原因
lockProof	String	订单锁定证明
creatorDID	String	流程创建者DID，如果没有加入任何流程，为“”
processID	String	当前订单所属流程ID，如果没有加入任何流程，为“”
encryptData	String	base64编码的数据密文

状态码： 500

表 5-194 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productName" : "prodname",
  "productID" : "product2",
  "sampleUrl" : "http://sdcloud.com/sample.com/prodname2",
  "sampleSize" : "10KB",
  "sampleType" : "csv",
  "sampleName" : "data_sub1",
  "fileType" : "csv",
  "dataUrl" : "http://sdcloud.com/prodname2",
  "dataHash" : "2282ba7a1a2ef5700609214a997d3d4237a03bfd3632c6d089e57e7b6f467969",
  "dataSize" : "100MB",
  "dataName" : "mydata1",
  "description" : "this is my second prod",
  "plainData" : "base64 encoding string",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1",
  "watermarkType" : "string",
  "file" : "string",
  "productIDKeywords" : "[{\"value\":\"taiyuan\"},{\"value\":\"renmin_hospital\"},{\"value\":\"medicine\"}]",
  "onChainStore" : "string",
  "consumerName" : "string"
}
```

响应示例

状态码： 200

订单信息

```
{
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "consumerName" : "Tyler",
  "orderSeq" : "1",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productID" : "product1",
  "productName" : "prodname1",
  "price" : "0",
  "applyTime" : "1607332359",
  "encryptedAesKey" : "BNGhPwjaTgpM+V7czzw1i4mH21KKN+XLKXHLqVsRIfybUCncqZNfomkRfzX4WEHj
+oty1X9oCd4h6xMnRvs8BWE5Tvg6BJ6QTW/km9EO/FSYqzJf2GqQzAleAcLJrTBZ3LRbPaF87CgJ114ae7R
+VK9VvfXQ8exuH2KMRD305dXieGpM4VPVv9u1BbL15Jpd/g==",
  "status" : "ready",
  "reason" : "I want product1",
```

```
  "lockProof" : "",
  "encryptData" : "base64 encoding string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单信息
500	失败响应

5.5.2.7 获取数据解密后的明文

功能介绍

获取数据解密后的明文。（ 不支持水印功能 ）

URI

POST /v1/datashare/dataset/query-plaintext

请求参数

表 5-195 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识
orderSeq	是	String	订单序列号
encryptData	否	String	数据密文，当onChainStore设置为“true”时，可不输入

参数	是否必选	参数类型	描述
watermarkType	否	String	水印类型，visible明水印，blind暗水印，嵌入水印时，必须填写本字段。如果发布或主动分享数据集的时候嵌入的是暗水印，则无法再次嵌入暗水印。嵌入的水印内容为：使用人did_orderID。
onChainStore	否	String	数据密文是否在链上存储，可设置为“true”或“false”，默认为“false”。如果设置为“true”，则不需要输入encryptData，可自动在链上获取数据密文

响应参数

状态码： 200

表 5-196 响应 Body 参数

参数	参数类型	描述
provider	String	订单提供者身份标识
productID	String	数据集产品id
plaintext	String	base64编码的数据明文

状态码： 500

表 5-197 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
```

```
  "orderSeq" : "1",
  "encryptData" : "base64 encoding string",
  "watermarkType" : "string",
  "onChainStore" : "string"
}
```

响应示例

状态码： 200

订单信息。

```
{
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "productID" : "product1",
  "plaintext" : "base64 encoding string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单信息。
500	失败响应

5.5.2.8 提取文件中的暗水印

功能介绍

提取文件中的暗水印。（ 不支持水印功能 ）

URI

POST /v1/datashare/dataset/watermark/extract

请求参数

表 5-198 FormData 参数

参数	是否必选	参数类型	描述
file	是	File	需要提取暗水印的文档

响应参数

状态码： 200

表 5-199 响应 Body 参数

参数	参数类型	描述
watermark	String	文件中嵌入的暗水印内容，不嵌入暗水印时内容为空。

状态码： 500

表 5-200 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

无

响应示例

状态码： 200

提取暗水印的返回内容

```
{
  "watermark" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	提取暗水印的返回内容
500	失败响应

5.5.2.9 查询指定的数据集分享流程

功能介绍

查询指定的数据集分享流程

URI

POST /v1/datashare/dataset/query-process

请求参数

表 5-201 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
creatorDID	是	String	流程创建者身份标识
processID	是	String	流程ID

响应参数

状态码： 200

表 5-202 响应 Body 参数

参数	参数类型	描述
creatorDID	String	流程创建者身份标识
processID	String	流程ID
stages	Array of **StageInProcess** objects	流程中的阶段信息

表 5-203 StageInProcess

参数	参数类型	描述
stageName	String	阶段名称
createTime	String	阶段信息上链时间戳
consumer	String	消费者身份标识
orderSeq	String	订单序列号

状态码： 500

表 5-204 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
/v1/datashare/dataset/query-process

{
  "orgID" : "{{orgID}}",
  "channelID" : "{{channelID}}",
  "cryptoMethod" : "{{cryptoMethod}}",
  "cert" : "{{cert}}",
  "sk" : "{{sk}}",
  "timestamp" : "{{timestamp}}",
  "creatorDID" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
  "processID" : "a779dd88-f7a3-4ac9-bf1c-a0ed1d827632"
}
```

响应示例

状态码： 200

查询指定的数据集分享流程响应参数

```
{
  "creatorDID" : "did:example:YLgvmFcukyigJpRsqRbFMn",
  "processID" : "25f46489-29bb-4f58-8f4c-e12da0a4bd66",
  "stages" : [ {
    "stageName" : "transaction1",
    "createTime" : "1640574210",
    "consumer" : "did:example:My8PRB5dKDVvBKXT76oJoB",
    "orderSeq" : "8zLQUpyswA8kpiHEsAUhZN"
  }, {
    "stageName" : "transaction2",
    "createTime" : "1640574304",
    "consumer" : "did:example:T1kFDUQAqo2z2X7hJWiRtQ",
    "orderSeq" : "3oKCPSKLGvaebT6z3a4PvY"
  }, {
    "stageName" : "transaction2",
    "createTime" : "1640587323",
```

```
    "consumer" : "did:example:T1kFDUQAqo2z2X7hJWiRtQ",
    "orderSeq" : "GPSta7mTScEEaQVRB62wUF"
  }, {
    "stageName" : "transaction2",
    "createTime" : "1640680918",
    "consumer" : "did:example:T1kFDUQAqo2z2X7hJWiRtQ",
    "orderSeq" : "8kzahSLi2kBxY8GGBZdLhp"
  } ]
}
```

状态码

状态码	描述
200	查询指定的数据集分享流程响应参数
500	失败响应

5.5.2.10 查询指定流程创建者的所有流程

功能介绍

查询指定流程创建者的所有流程

URI

POST /v1/datashare/dataset/query-processes

请求参数

表 5-205 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
currentPage	否	String	分页参数：当前页码（默认1）
pageSizeNum	否	String	分页参数：每页条数(默认100)
creatorDID	是	String	流程创建者身份标识

响应参数

状态码： 200

表 5-206 响应 Body 参数

参数	参数类型	描述
items	Array of **DatasetShareProcessResponseBody** objects	数据集分享流程列表
pagination	**PaginationResp** object	分页信息

表 5-207 DatasetShareProcessResponseBody

参数	参数类型	描述
creatorDID	String	流程创建者身份标识
processID	String	流程ID
stages	Array of **StageInProcess** objects	流程中的阶段信息

表 5-208 StageInProcess

参数	参数类型	描述
stageName	String	阶段名称
createTime	String	阶段信息上链时间戳
consumer	String	消费者身份标识
orderSeq	String	订单序列号

表 5-209 PaginationResp

参数	参数类型	描述
currentPage	Integer	当前页码
pageSizeNum	Integer	每页条数
totalItems	Integer	总条数

状态码： 500

表 5-210 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
/v1/datashare/dataset/query-processes

{
  "orgID" : "{{orgID}}",
  "channelID" : "{{channelID}}",
  "cryptoMethod" : "{{cryptoMethod}}",
  "cert" : "{{cert}}",
  "sk" : "{{sk}}",
  "timestamp" : "{{timestamp}}",
  "creatorDID" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
  "currentPage" : "1",
  "pageSizeNum" : "100"
}
```

响应示例

状态码： 200

查询指定流程创建者的所有流程响应参数

```
{
  "items" : [ {
    "creatorDID" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
    "processID" : "442a6b42-82b7-415a-a0e6-deaaee59f582",
    "stages" : [ {
      "stageName" : "Seconde transaction",
      "createTime" : "1639824526",
      "consumer" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
      "orderSeq" : "N6UhspZ5cQY7NtHsxuFTZ"
    } ]
  }, {
    "creatorDID" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
    "processID" : "a779dd88-f7a3-4ac9-bf1c-a0ed1d827632",
    "stages" : [ {
      "stageName" : "First transaction",
      "createTime" : "1639824239",
      "consumer" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
      "orderSeq" : "FKCx1Cfatj7RRsKMWPQ7wQ"
    }, {
      "stageName" : "",
      "createTime" : "1639826471",
      "consumer" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
      "orderSeq" : "8PuuWWg521bZDXadzwPdMn"
    }, {
      "stageName" : "Seconde transaction",
      "createTime" : "1639826898",
      "consumer" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
      "orderSeq" : "SCkh1rQ6aD5SYTYkojn44W"
    } ]
  }, {
    "creatorDID" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
    "processID" : "d15b5213-5cb8-4af4-97dc-143379369f35",
```

```
  "stages" : [ {
    "stageName" : "564econde transaction",
    "createTime" : "1639827681",
    "consumer" : "did:example:8sAvsS4tB3NYgMJ4uqbVYj",
    "orderSeq" : "XCXtVZsdKFDLaErzpSZYAN"
  } ]
 } ],
 "pagination" : {
  "currentPage" : 1,
  "pageSizeNum" : 100,
  "totalItems" : 3
 }
}
```


状态码

状态码	描述
200	查询指定流程创建者的所有流程响应参数
500	失败响应

5.5.3 数据订单管理

5.5.3.1 申请数据集

功能介绍

申请数据集

URI

POST /v1/datashare/dataset/dataset-order

请求参数

表 5-211 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识

参数	是否必选	参数类型	描述
orderSeq	是	String	订单序列号
provider	是	String	数据集发布者身份标识
productID	是	String	数据集产品id
reason	否	String	申请原因
consumerName	否	String	数据集申请者名称

响应参数

状态码： 200

表 5-212 响应 Body 参数

参数	参数类型	描述
consumer	String	订单消费者身份标识
consumerName	String	订单消费者名称
orderSeq	String	订单序列号
provider	String	订单提供者身份标识
providerName	String	订单提供者名称
productID	String	数据集产品id
productName	String	数据集产品名称
price	String	订单价钱
applyTime	String	订单申请时间
encryptedAesKey	String	密钥
status	String	订单状态
reason	String	订单申请原因
lockProof	String	订单锁定证明
creatorDID	String	流程创建者DID，如果没有加入任何流程，为“”
processID	String	当前订单所属流程ID，如果没有加入任何流程，为“”

状态码： 500

表 5-213 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "productID" : "product2",
  "reason" : "apply dataset for AI",
  "consumerName" : "user1"
}
```

响应示例

状态码： 200

订单信息

```
{
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "consumerName" : "Tyler",
  "orderSeq" : "1",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productID" : "product1",
  "productName" : "prodname1",
  "price" : "0",
  "applyTime" : "1607332359",
  "encryptedAesKey" : "BNGhPwjaTgpM+V7czzw1i4mH21KKN+XLKXHLqVsRIfybUCncqZNfomkRfzX4WEHj
+oty1X9oCd4h6xMnRvs8BWE5Tvg6BJ6QTW/km9EO/FSYqzJf2GqQzAleAcLJrTBZ3LRbPaF87CgJ114ae7R
+VK9VvfXQ8exuH2KMRD305dXieGpM4VPVv9u1BbL15Jpd/g==",
  "status" : "ready",
  "reason" : "I want product1",
  "lockProof" : ""
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单信息
500	失败响应

5.5.3.2 授权数据集

功能介绍

授权数据集

URI

POST /v1/datashare/dataset/authorize-dataset

请求参数

表 5-214 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识
orderSeq	是	String	订单序列号

响应参数

状态码： 200

表 5-215 响应 Body 参数

参数	参数类型	描述
consumer	String	订单消费者身份标识

参数	参数类型	描述
consumerName	String	订单消费者名称
orderSeq	String	订单序列号
provider	String	订单提供者身份标识
providerName	String	订单提供者名称
productID	String	数据集产品id
productName	String	数据集产品名称
price	String	订单价钱
applyTime	String	订单申请时间
encryptedAesKey	String	密钥
status	String	订单状态
reason	String	订单申请原因
lockProof	String	订单锁定证明
creatorDID	String	流程创建者DID，如果没有加入任何流程，为“”
processID	String	当前订单所属流程ID，如果没有加入任何流程，为“”

状态码： 500

表 5-216 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1"
}
```

响应示例

状态码： 200

订单信息

```
{
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "consumerName" : "Tyler",
  "orderSeq" : "1",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productID" : "product1",
  "productName" : "prodname1",
  "price" : "0",
  "applyTime" : "1607332359",
  "encryptedAesKey" : "BNGhPwjaTgpM+V7czzw1i4mH21KKN+XLKXHLqVsRIfybUCncqZNfomkRfzX4WEHj
+oty1X9oCd4h6xMnRvs8BWE5Tvg6BJ6QTW/km9EO/FSYqzJf2GqQzAleAcLJrTBZ3LRbPaF87CgJ114ae7R
+VK9VvfXQ8exuH2KMRD305dXieGpM4VPVv9u1BbL15Jpd/g==",
  "status" : "ready",
  "reason" : "I want product1",
  "lockProof" : ""
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单信息
500	失败响应

5.5.3.3 修改订单状态

功能介绍

修改订单状态

URI

PUT /v1/datashare/dataset/order

请求参数

表 5-217 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识
orderSeq	是	String	订单序列号
orderStatus	是	Integer	设置订单状态（0完成，1失败，2取消）
reason	否	String	原因

响应参数

状态码： 200

表 5-218 响应 Body 参数

参数	参数类型	描述
result	String	操作结果

状态码： 500

表 5-219 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
```

```
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1",
  "orderStatus" : 0,
  "reason" : "string"
}
```

响应示例

状态码： 200

操作结果

```
{
  "result" : "success"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	操作结果
500	失败响应

5.5.3.4 删除订单

功能介绍

删除订单

URI

DELETE /v1/datashare/dataset/order

请求参数

表 5-220 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id

参数	是否必选	参数类型	描述
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识
orderSeq	是	String	订单序列号

响应参数

状态码： 200

表 5-221 响应 Body 参数

参数	参数类型	描述
result	String	操作结果

状态码： 500

表 5-222 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1"
}
```

响应示例

状态码： 200

操作结果


```
{
  "result" : "success"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	操作结果
500	失败响应

5.5.3.5 查询指定订单

功能介绍

查询指定订单，根据订单消费者did和订单序列号查询。

URI

POST /v1/datashare/dataset/query-order

请求参数

表 5-223 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
consumer	是	String	订单申请者身份标识
orderSeq	是	String	订单序列号

响应参数

状态码： 200

表 5-224 响应 Body 参数

参数	参数类型	描述
consumer	String	订单消费者身份标识
consumerName	String	订单消费者名称
orderSeq	String	订单序列号
provider	String	订单提供者身份标识
providerName	String	订单提供者名称
productID	String	数据集产品id
productName	String	数据集产品名称
price	String	订单价钱
applyTime	String	订单申请时间
encryptedAesKey	String	密钥
status	String	订单状态
reason	String	订单申请原因
lockProof	String	订单锁定证明
creatorDID	String	流程创建者DID，如果没有加入任何流程，为“”
processID	String	当前订单所属流程ID，如果没有加入任何流程，为“”

状态码： 500

表 5-225 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
```

```
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "orderSeq" : "1"
}
```

响应示例

状态码： 200

订单信息

```
{
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
  "consumerName" : "Tyler",
  "orderSeq" : "1",
  "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
  "providerName" : "sd",
  "productID" : "product1",
  "productName" : "prodname1",
  "price" : "0",
  "applyTime" : "1607332359",
  "encryptedAesKey" : "BNGhPwjaTgpM+V7czzw1i4mH21KKN+XLKXHLqVsRIfybUCncqZNfomkRfzX4WEHj
+oty1X9oCd4h6xMnRvs8BWE5Tvg6BJ6QTW/km9EO/FSYqzJf2GqQzAleAcLJrTBZ3LRbPaF87CgJ114ae7R
+VK9VvfXQ8exuH2KMRD305dXieGpM4VPVv9u1BbL15Jpd/g==",
  "status" : "ready",
  "reason" : "I want product1",
  "lockProof" : ""
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单信息
500	失败响应

5.5.3.6 查询订单列表

功能介绍

查询订单，支持分页和按条件过滤查询。

URI

POST /v1/datashare/dataset/query-orders

请求参数

表 5-226 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法，目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
currentPage	否	String	分页参数：当前页码（默认1）
pageSizeNum	否	String	分页参数：每页条数(默认100)
provider	否	String	过滤条件：订单中数据集发布者身份标识
searchText	否	String	过滤条件：匹配关键字（订单的产品名称）
status	否	String	过滤条件：订单状态（ready、finished、failed、canceled）
consumer	否	String	过滤条件：订单消费者身份标识

响应参数

状态码： 200

表 5-227 响应 Body 参数

参数	参数类型	描述
items	Array of **DataOrderResponse** objects	列表
pagination	**PaginationResp** object	分页信息

表 5-228 DataOrderResponse

参数	参数类型	描述
consumer	String	订单消费者身份标识
consumerName	String	订单消费者名称
orderSeq	String	订单序列号
provider	String	订单提供者身份标识
providerName	String	订单提供者名称
productID	String	数据集产品id
productName	String	数据集产品名称
price	String	订单价钱
applyTime	String	订单申请时间
encryptedAesKey	String	密钥
status	String	订单状态
reason	String	订单申请原因
lockProof	String	订单锁定证明
creatorDID	String	流程创建者DID，如果没有加入任何流程，为“”
processID	String	当前订单所属流程ID，如果没有加入任何流程，为“”

表 5-229 PaginationResp

参数	参数类型	描述
currentPage	Integer	当前页码
pageSizeNum	Integer	每页条数
totalItems	Integer	总条数

状态码： 500

表 5-230 响应 Body 参数

参数	参数类型	描述
errorCode	String	错误码

参数	参数类型	描述
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "currentPage" : "string",
  "pageSizeNum" : "string",
  "provider" : "string",
  "searchText" : "string",
  "status" : "string",
  "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG"
}
```

响应示例

状态码： 200

订单分页信息

```
{
  "items" : [ {
    "consumer" : "did:example:3TMWx8owKHARgNwbj4ywmG",
    "consumerName" : "Tyler",
    "orderSeq" : "1",
    "provider" : "did:example:DHkJjyD5wZwya6sd6BNBnG",
    "providerName" : "sd",
    "productID" : "product1",
    "productName" : "prodname1",
    "price" : "0",
    "applyTime" : "1607332359",
    "encryptedAesKey" : "BNGhPwjaTgpM+V7czzw1i4mH21KKN+XLKXHLqVsRIfybUCncqZNfomkRfzX4WEHj
+oty1X9oCd4h6xMnRvs8BWE5Tvg6BJ6QTW/km9EO/FSYqzJf2GqQzAleAcLJrTBZ3LRbPaF87CgJ114ae7R
+VK9VvfXQ8exuH2KMRD305dXieGpM4VPVv9u1BbL15Jpd/g==",
    "status" : "ready",
    "reason" : "I want product1",
    "lockProof" : ""
  } ],
  "pagination" : {
    "currentPage" : 1,
    "pageSizeNum" : 100,
    "totalItems" : 10
  }
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	订单分页信息
500	失败响应

5.5.4 属性加密的密钥管理

5.5.4.1 初始化 ABE 主密钥

功能介绍

初始化ABE主密钥，如果owner未初始化过ABE主密钥，则自动生成并存储在链上。如果owner已有ABE主密钥，不会覆盖。

URI

POST /v1/datashare/abe-setup

请求参数

表 5-231 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
owner	是	String	密钥生成者的身份标识
keyManagerMode	否	String	abe系统首次使用时，需选择中心模式“central”或者多中心模式“distributed”，该模式仅可选择一次，默认为多中心模式。

响应参数

状态码： 200

表 5-232 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥

状态码： 500

表 5-233 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "owner" : "did:example:8poVETnVCry9ecfHSDeQaR"
}
```

响应示例

状态码： 200

ABE主密钥信息

```
{
  "secretJson" : "{}",
  "publicKeyJson" : "{}"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	ABE主密钥信息
500	失败响应

5.5.4.2 更新 ABE 主密钥

功能介绍

更新ABE主密钥

URI

POST /v1/datashare/abe-update

请求参数

表 5-234 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
secretKeyJson	否	String	json格式的abe主私钥
publicKeyJson	否	String	json格式的abe主公钥
owner	是	String	密钥生成者的身份标识

响应参数

状态码： 200

表 5-235 响应 Body 参数

参数	参数类型	描述
oldSecretJson	String	旧ABE主私钥

参数	参数类型	描述
oldPublicKeyJson	String	旧ABE主公钥
newSecretJson	String	新ABE主私钥
newPublicKeyJson	String	新ABE主公钥

状态码： 500

表 5-236 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "owner" : "did:example:8poVETnVCry9ecfHSDeQaR",
  "secretKeyJson" : "string",
  "publicKeyJson" : "string"
}
```

响应示例

状态码： 200

原有及更新后的ABE主密钥信息

```
{
  "oldSecretJson": "{}",
  "oldPublicKeyJson": "{}",
  "newSecretJson": "{}",
  "newPublicKeyJson": "{}"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
```

```
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	原有及更新后的ABE主密钥信息
500	失败响应

5.5.4.3 查询 ABE 主密钥

功能介绍

查询ABE主密钥

URI

POST /v1/datashare/abekey

请求参数

表 5-237 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
owner	是	String	密钥生成者的身份标识
keyManagerMode	否	String	abe系统首次使用时，需选择中心模式“central”或者多中心模式“distributed”，该模式仅可选择一次，默认为多中心模式。

响应参数

状态码： 200

表 5-238 响应 Body 参数

参数	参数类型	描述
secretKeyJson	String	json格式的abe主私钥
publicKeyJson	String	json格式的abe主公钥

状态码： 500

表 5-239 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "owner" : "did:example:8poVETnVCry9ecfHSDeQaR"
}
```

响应示例

状态码： 200

ABE主密钥信息

```
{
  "secretKeyJson" : "string",
  "publicKeyJson" : "string"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	ABE主密钥信息
500	失败响应

5.5.4.4 申请 ABE 用户密钥

功能介绍

申请ABE用户密钥

URI

POST /v1/datashare/abekey-order

请求参数

表 5-240 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
applyer	是	String	申请者的身份标识
provider	是	String	授权者的身份标识
attrJson	是	Array of **attribute** objects	属性列表

表 5-241 attribute

参数	是否必选	参数类型	描述
name	是	String	属性名
type	是	String	属性类型（plain，comparable）

参数	是否必选	参数类型	描述
value	是	String	属性值（当type为plain时，value为属性值。当type为comparable时，value必须是整数。）
maxValue	否	String	属性值上限，value可能取到的最大值。只在type为comparable时可选使用

响应参数

状态码： 200

表 5-242 响应 Body 参数

参数	参数类型	描述
applyer	String	申请者的身份标识
applyerName	String	申请者的名称
provider	String	授权者的身份标识
providerName	String	授权者的名称
service	String	授权者的服务名
price	Integer	价格
applyTime	String	申请时间
encryptedABEKey	String	被加密的ABE密钥
status	String	申请状态，request表示未授权；ready表示申请已处理
reason	String	原因
lockProof	String	证明
attributesJson	String	属性

状态码： 500

表 5-243 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥

参数	参数类型	描述
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID": "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID": "mychannel",
  "cryptoMethod": "SW",
  "cert": "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk": "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp": “2020-10-27T17:28:16+08:00”,
  "applyer": "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "provider": "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "attrJson": "[{\"name\":\"att1\",\"type\":\"plain\",\"value\":\"att1name\"},{\"name\":\"att2\",\"type\":\"plain
\",\"value\":\"att2name\"},{\"name\":\"att3\",\"type\":\"plain\",\"value\":\"5\"}]"
}
```

响应示例

状态码： 200

ABE用户密钥订单信息

```
{
  "applyer" : "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "provider" : "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "applyTime" : "1622166512",
  "status" : "ready",
  "attributesJson" : "{\"att1\": \"YXR0MW5hbWU=\",\"att2\": \"YXR0Mm5hbWU=\",\"att3\": \"NQ==\"}"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	ABE用户密钥订单信息
500	失败响应

5.5.4.5 授权 ABE 用户密钥

功能介绍

授权ABE用户密钥

URI

PUT /v1/datashare/abekey-order

请求参数

表 5-244 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
applyer	是	String	申请者的身份标识
provider	是	String	授权者的身份标识
attrJson	否	Array of **attribute** objects	属性列表

表 5-245 attribute

参数	是否必选	参数类型	描述
name	是	String	属性名
type	是	String	属性类型（plain，comparable）
value	是	String	属性值（当type为plain时，value为属性值。当type为comparable时，value必须是整数。）
maxValue	否	String	属性值上限，value可能取到的最大值。只在type为comparable时可选使用

响应参数

状态码： 200

表 5-246 响应 Body 参数

参数	参数类型	描述
applyer	String	申请者的身份标识
applyerName	String	申请者的名称
provider	String	授权者的身份标识
providerName	String	授权者的名称
service	String	授权者的服务名
price	Integer	价格
applyTime	String	申请时间
encryptedABEKey	String	被加密的ABE密钥
status	String	申请状态，request表示未授权；ready表示申请已处理
reason	String	原因
lockProof	String	证明
attributesJson	String	属性

状态码： 500

表 5-247 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
```

```
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "applyer" : "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "provider" : "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "attrJson" : "[{\"name\":\"att1\",\"type\":\"plain\",\"value\":\"att1name\"},{\"name\":\"att2\",\"type\":
\"plain\",\"value\":\"att2name\"},{\"name\":\"att3\",\"type\":\"plain\",\"value\":\"5\"}]"
}
```

响应示例

状态码： 200

ABE用户密钥订单信息

```
{
  "applyer" : "did:example:mfqqdiW8V64JbPFgQsoiv",
  "applyerName" : "",
  "provider" : "did:example:FahQr32NgQZWjGRiCZc37C",
  "providerName" : "",
  "service" : "",
  "price" : 0,
  "applyTime" : "",
  "encryptedABEKey" : "",
  "status" : "ready",
  "reason" : "",
  "lockProof" : "",
  "attributesJson" : "[{\"name\":\"att3\",\"type\":\"comparable\",\"value\":\"3\",\"maxValue\":\"1000\"}]"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	ABE用户密钥订单信息
500	失败响应

5.5.4.6 查询 ABE 用户密钥申请

功能介绍

查询ABE用户密钥申请

URI

POST /v1/datashare/query-abekey

请求参数

表 5-248 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
provider	是	String	授权者的身份标识
applyer	是	String	申请者的身份标识

响应参数

状态码： 200

表 5-249 响应 Body 参数

参数	参数类型	描述
applyer	String	申请者的身份标识
applyerName	String	申请者的名称
provider	String	授权者的身份标识
providerName	String	授权者的名称
service	String	授权者的服务名
price	Integer	价格
applyTime	String	申请时间
encryptedABEKey	String	被加密的ABE密钥
status	String	申请状态，request表示未授权；ready表示申请已处理
reason	String	原因
lockProof	String	证明
attributesJson	String	属性

状态码： 500

表 5-250 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

无

响应示例

状态码： 200

ABE用户密钥订单信息

```
{
  "applyer" : "did:example:mfqqdiW8V64JbPFgQsoiv",
  "applyerName" : "",
  "provider" : "did:example:FahQr32NgQZWjGRiCZc37C",
  "providerName" : "",
  "service" : "",
  "price" : 0,
  "applyTime" : "1672985584",
  "encryptedABEKey" : "",
  "status" : "ready",
  "reason" : "",
  "lockProof" : "",
  "attributesJson" : "{\"att1\":\"YXR0MW5hbWU=\",\"att2\":\"YXR0Mm5hbWU=\",\"att3\":\"NQ==\"}"
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	ABE用户密钥订单信息
500	失败响应

5.5.4.7 ABE 用户密钥解密数据

功能介绍

ABE用户密钥解密数据

URI

POST /v1/datashare/abe-decrypt

请求参数

表 5-251 请求 Body 参数

参数	是否必选	参数类型	描述
orgID	是	String	组织id
channelID	是	String	通道id
cryptoMethod	是	String	加密方法,目前固定为SW
cert	是	String	证书
sk	是	String	私钥
timestamp	是	String	时间戳
encryptData	是	String	数据密文，当onChainStore设置为“true”时，可不输入
applyer	是	String	申请者的身份标识
provider	是	String	授权者的身份标识
orderSeq	否	String	订单序列号，当onChainStore为true时，订单序列号必填。
onChainStore	否	String	数据密文是否在链上存储，可设置为“true”或“false”，默认为“false”。如果设置为“true”，则不需要输入encryptData，可自动在链上获取数据密文

响应参数

状态码： 200

表 5-252 响应 Body 参数

参数	参数类型	描述
plainData	String	base64处理过的解密后数据

状态码： 500

表 5-253 响应 Body 参数

参数	参数类型	描述
secretJson	String	ABE主私钥
publicKeyJson	String	ABE主公钥
errorCode	String	错误码
errorMsg	String	错误描述

请求示例

```
{
  "orgID" : "ce0ac69b0c8648cd25b44a551780409767c8890b",
  "channelID" : "mychannel",
  "cryptoMethod" : "SW",
  "cert" : "-----BEGIN CERTIFICATE-----\\n...\\n-----END CERTIFICATE-----",
  "sk" : "-----BEGIN PRIVATE KEY-----\\n...\\n-----END PRIVATE KEY-----",
  "timestamp" : "2020-10-27T17:28:16+08:00",
  "encryptData" : "string",
  "onChainStore" : "false",
  "applyer" : "did:example:Mb4SshJeN5ukWXkbMJK8xC",
  "provider" : "did:example:Mb4SshJeN5ukWXkbMJK8xC"
}
```

响应示例

状态码： 200

base64编码的ABE解密后数据

```
{
  "plainData" : "aGVsbG8sdGhpcyBpcyBhbiBleGFtcGxlIGZvciBhYmU="
}
```

状态码： 500

失败响应

```
{
  "errorCode" : "BCS.5002046",
  "errorMsg" : "Incorrect number of arguments"
}
```

状态码

状态码	描述
200	base64编码的ABE解密后数据
500	失败响应

6 附录

6.1 国密加密

6.1.1 概述

国密是国家商用密码的简称，商用密码是指对不涉及国家秘密内容的信息进行加密保护或者安全认证所使用的密码技术和密码产品。

国密算法是国家密码管理局制定的自主可控的国产算法，可提高加密强度和加解密性能。使用国密加密，可以满足政府机构、事业单位、大型国企、金融银行等行业的改造和国密算法的需求。

区块链服务提供国密加密算法SDK供您使用，供用户开发客户端程序以及对私钥文件进行加密保护。

资源下载

表 6-1 SDK 列表

配套社区Hyperledger Fabric版本	语言	下载链接
Fabric 2.2	Go	登录区块链服务管理控制台，进入“应用案例”，单击“国密加密SDK”中Fabric_SDK_Go的“下载”按钮。
	Java	登录区块链服务管理控制台，进入“应用案例”，单击“国密加密SDK”中Fabric_SDK_Gateway_Java或Fabric_SDK_Java的“下载”按钮。
说明 • Go版本要求：1.12及以上，1.16以下（>=1.12，<1.16）。 • 国密SDK涵盖了普通SDK的所有功能，并在此基础上增加了对国密算法的支持。 • Fabric_SDK_Gateway_Java对SDK的部分接口进行了封装，涵盖Fabric_SDK_Java同时更加简便易用，推荐选用。		

将下载的压缩包解压后，得到如下目录，目录的功能如下表：

目录	说明
src（仅Go）	存放Go SDK的源码文件。
jar（仅Java）	存放Java SDK的Jar包。

6.1.2 SDK 的使用

安装 SDK

如何获取GO、JAVA压缩包、Jar文件请参考**概述**。

- GO：将下载的压缩包解压到用户的$GOPATH目录下。
- Java：将下载的压缩包中的Jar文件添加到项目的依赖中，可按以下方式添加：
  a. 将下载的SDK Jar包注册至Maven本地仓库，可参考以下命令：
  ```
  Mvn install:install-file -Dfile=fabric-sdk-java-2.2.6-jar-with-dependencies.jar -DgroupId=org.hyperledger.fabric-sdk-java -DartifactId=fabric-sdk-java -Dversion=2.2.6-BCS -Dpackaging=jar
  ```
  b. 在项目中依赖SDK，可参考以下代码：
  ```
  <dependency>
      <groupId>org.hyperledger.fabric-sdk-java</groupId>
      <artifactId>fabric-sdk-java</artifactId>
      <version>2.2.6-BCS</version>
  </dependency>
  ```

运行 Client 程序

Client程序运行时一般要设定其使用的配置文件路径、通道名称、链代码名称、组织ID等。

- 配置文件路径即用户下载配置文件的存放路径。
- 通道名称即BCS实例中的通道名称。
- 链代码名称即BCS实例中安装链代码时设定的名称。
- 组织ID，以如下示例配置文件内容为例，组织ID为02f23ab00f6e1ffcde8a27bfd3ac2290edc18127
  ```
  client:
  organization: 02f23ab00f6e1ffcde8a27bfd3ac2290edc18127
  ```

6.1.3 附录

fabric-sdk-client/go依赖的第三方包列表：

序号	包名
1	github.com/Knetic/govaluate
2	github.com/VividCortex/gohistogram

序号	包名
3	github.com/cloudflare/cfssl
4	github.com/go-kit/kit
5	github.com/golang/mock
6	github.com/golang/protobuf
7	github.com/hashicorp/hcl
8	github.com/hyperledger/fabric-config
9	github.com/hyperledger/fabric-lib-go
10	github.com/hyperledger/fabric-protos-go
11	github.com/magiconair/properties
12	github.com/miekg/pkcs11
13	github.com/mitchellh/mapstructure
14	github.com/pelletier/go-toml
15	github.com/pkg/errors
16	github.com/prometheus/client_golang
17	github.com/spf13/afero
18	github.com/spf13/cast
19	github.com/spf13/jwalterweatherman
20	github.com/spf13/pflag
21	github.com/spf13/viper
22	github.com/stretchr/testify
23	github.com/tjfoc/gmsm
24	google.golang.org/grpc
25	gopkg.in/yaml.v2

6.2 错误码

调用接口出错后，调用方可根据每个接口对应的错误码和错误描述信息来定位错误原因。 当调用出错时，HTTPS请求返回一个4xx或5xx的HTTPS状态码。返回的消息体中是具体的错误代码及错误信息。在调用方找不到错误原因时，可以联系运维人员，并提供错误码，以便运维人员尽快帮您解决问题。

错误响应 Body 体格式说明

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如表6-2所示。

表 6-2 错误响应参数说明

参数	类型	说明
error_msg	String	错误码
error_code	String	错误描述信息

示例：

```
{
  "error_code": "BCS.4006009",
  "error_msg": "one of parameters is nil"
}
```

错误码说明

表 6-3 错误码说明

状态码	错误码	错误信息	处理措施
200	BCS.2001002	OneStepPurchase4Need successfully.	请求成功
400	BCS.4006009	one of parameters is nil	请给相应的参数赋值
400	BCS.4001051	Input org org1 is not exist	请检查相应参数
400	BCS.4030403	forbidden to project [project_id]	请检查project_id
400	BCS.4006003	json: cannot unmarshal object into Go value of type []apis.PeersToChannelAdd	请检查相应参数是否符合要求
400	BCS.4006005	ActOnDetailNotification fail	请联系技术支持解决

7 修订记录

发布日期	修订记录
2023-03-25	更新**区块链中间件接口**。
2021-01-15	第一次正式发布。