# 准备

修改本地Hosts文件

127.0.0.1 server.smart-sso.com

127.0.0.1 demo.smart-sso.com

数据库准备

数据库 ssodb

-- ssodb.auth\_history definition

CREATE TABLE ssodb.`auth\_history` (

`id` varchar(50) NOT NULL,

`user\_id` varchar(50) NOT NULL,

`client\_id` varchar(50) NOT NULL,

`create\_time` datetime DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- ssodb.password\_history definition

CREATE TABLE ssodb.`password\_history` (

`id` varchar(50) NOT NULL,

`user\_id` varchar(50) NOT NULL,

`password` varchar(50) NOT NULL,

`create\_time` datetime DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- ssodb.sso\_app definition

CREATE TABLE ssodb.`sso\_app` (

`id` varchar(50) NOT NULL,

`client\_name` varchar(20) NOT NULL,

`client\_id` varchar(50) NOT NULL,

`client\_secret` varchar(50) NOT NULL,

`redirect\_uri` varchar(200) NOT NULL,

`create\_time` datetime DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- ssodb.sso\_user definition

CREATE TABLE ssodb.`sso\_user` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`username` varchar(50) NOT NULL,

`password` varchar(50) NOT NULL,

`create\_time` datetime DEFAULT NULL,

`update\_time` datetime DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4;

INSERT INTO ssodb.sso\_app (id,client\_name,client\_id,client\_secret,redirect\_uri,create\_time) VALUES

('cd7825f3-3251-11eb-9476-3c91802a3758','服务端1','server1','123456','http://server.smart-sso.com:8080',NULL)

,('e7a7913a-3251-11eb-9476-3c91802a3758','客户端1','client1','123456','http://demo.smart-sso.com:8082/',NULL);

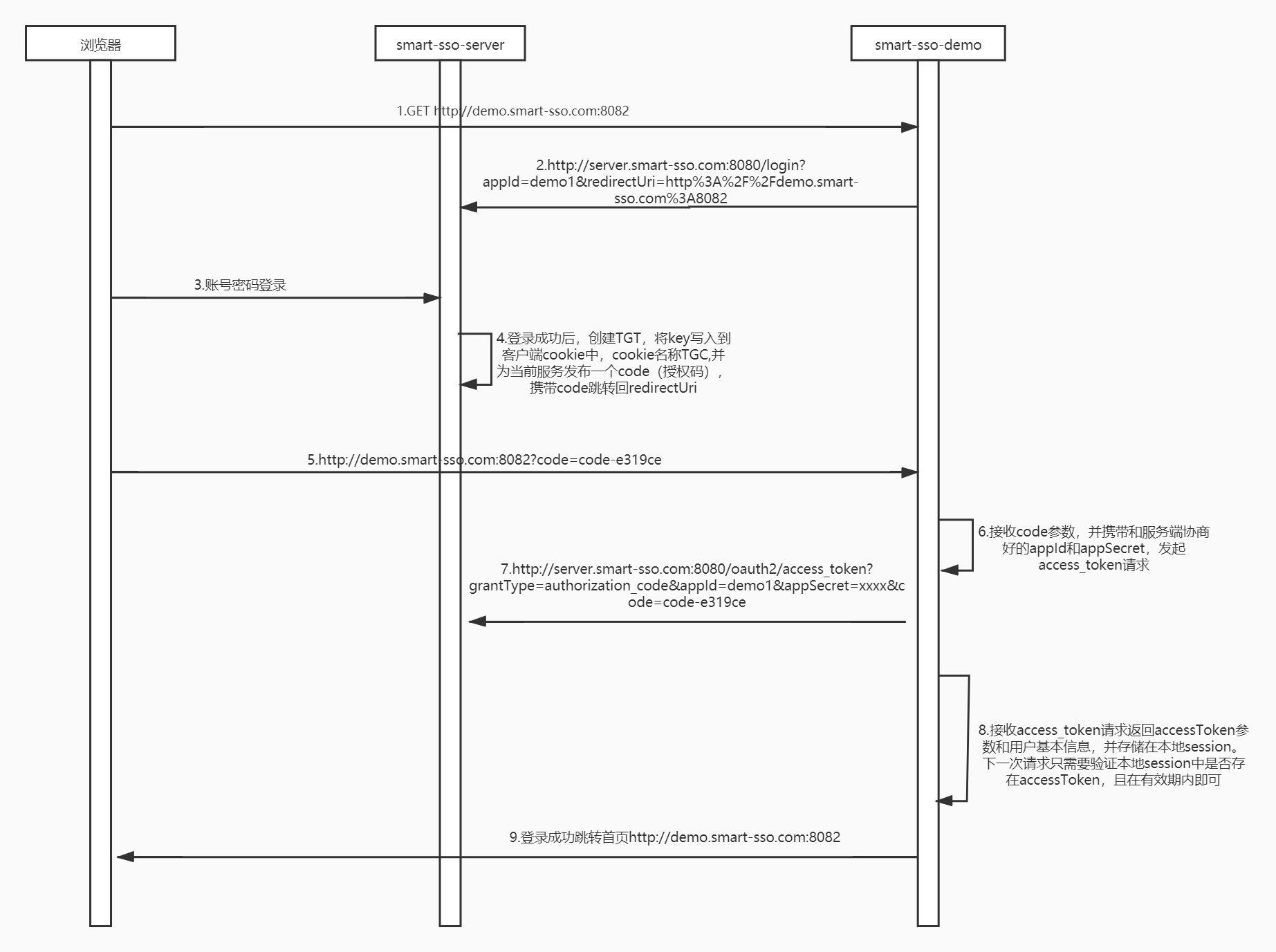
INSERT INTO ssodb.sso\_user (username,password,create\_time,update\_time) VALUES

('admin','123456',NULL,NULL);

测试脚本

见文件 SSO.postman\_collection.json

# 登录阶段



smart-sso-server 认证中心

负责管理认证用户，授权的客户服务，记录用户授权关系。负责统一清理会话状态。

smart-sso-demo 客户服务

负责本地账号与认证账号的对应关系

认证过程：

1. 浏览器尝试使用demo的服务，判断是否有登陆状态（这里用的是本地判断）
2. 没有认证信息会重定向到认证中心，认证中心判断有没有登陆信息，没有的话跳转到登录界面。重定向过程需要用的参数信息

response\_type：授权类型，必选项，此处的值固定为"code"

　　client\_id：客户端的ID，必选项

　　redirect\_uri：重定向URI，必选项

　　scope：申请的权限范围，可选项 read

state：任意值，认证服务器会原样返回,用于抵制CSRF(跨站请求伪造)攻击。 将state放在session里面

注：需要验证重定向地址是否client\_id对应的

1. 输入登录用户名、密码。验证登录。登录成功后，拿用户ID查询授权记录表，是否存在对饮client\_id的授权记录，如果没有的话需要转到授权确认界面（目前只做了过程，还需要细化）。
2. 登录成功后写入用户信息，生成授权码code，结合前面拿到的state参数，拼接重定向地址
3. 用上步的重定向地址返回到demo客户端。demo端要注意返回的state是否与发送的一致。
4. 通过授权码code交换访问令牌，所需要的参数信息

client\_id：表示客户端ID，必选项。

client\_secret：客户秘钥 用于二次验证客户

grant\_type：表示使用的授权模式，必选项，此处的值固定为"authorization\_code"。

code：表示上一步获得的授权码，必选项。

redirect\_uri：表示重定向URI，必选项，且必须与A步骤中的该参数值保持一致。

1. 认证中心接收请求验证参数，并验证内容，主要逻辑如下：

client\_id与client\_secret是否与认证中心保存的一致

code是否是由上次client\_id申请的

重定向地址是否client\_id对应的

返回访问令牌，应包括如下信息

access\_token：访问令牌，必选项。

token\_type:"bearer",

expires\_in：过期时间，单位为秒。如果省略该参数，必须其他方式设置过期时间。

refresh\_token：更新令牌，用来获取下一次的访问令牌，可选项。

scope:"read",

1. demo客户端接收访问令牌，缓存到session中，并获得在认证中心的用户信息，通过认证用户ID来查询本地绑定记录，如果有绑定就直接可以进一步获取权限。如果没有绑定本地账户，则需要指定绑定账户。 user信息为认证中心的登录用户，可用id字段与系统中用户绑定（即用这个字段查库，看有没有绑定的账户，没有的话进行绑定，绑定是还需要检查是否已经绑定过其他账号进行提醒）

{

"data": {

"accessToken": "AT-70950ad0ed7f43ba9952a675debb5a5a",

"expiresIn": 3600,

"refreshToken": "RT-ef1ec8e7b80c429092d2f0cca4baf498",

"user": {

"id": 1,

"username": "admin"

}

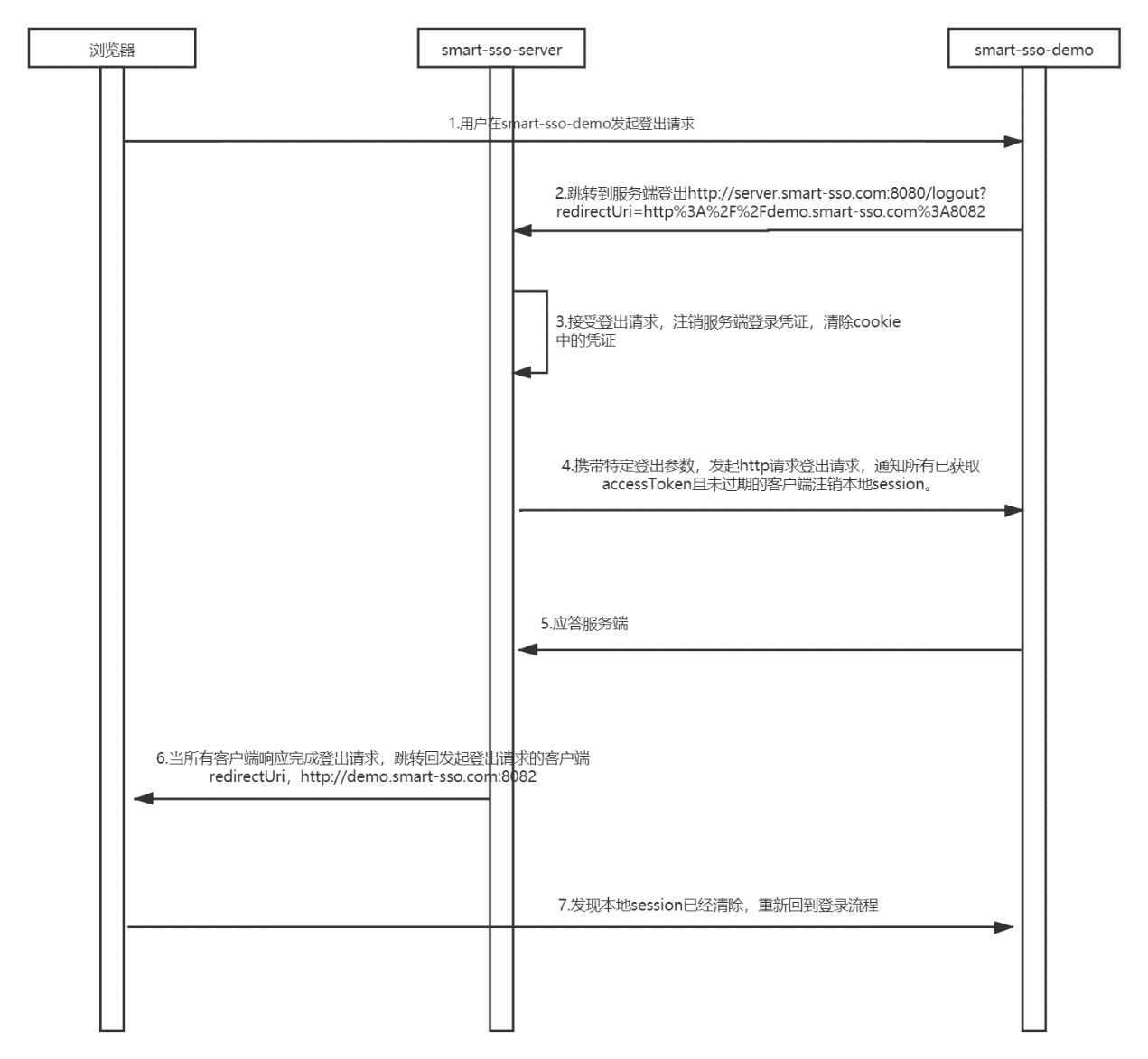
},

"code": 1,

"message": null

}

# 退出阶段



退出过程

1. 用户在demo页面调用退出按钮
2. demo服务实际上调用的是认证中心的退出功能
3. 认证中心清理凭证、缓存
4. 认证中心遍历访问令牌，向客户端注销令牌。这里用的登录时记录的回调地址。
5. demo端识别请求是否带有特殊标识来判断登出命令，并清理数据
6. 认证中心重定向到demo的回调地址
7. demo端重新发起认证流程

认证用户表 sso\_user

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 类型 | 限定条件 | 说明 |
| id | varchar | 主键 |  |
| username | varchar | 非空 | 登录用户名 |
| password | varchar | 非空 | 登录密码 |
| create\_time | datetime |  | 创建时间 |
| update\_time | datetime |  | 修改时间（上次修改密码时间） |

第三方登记表 sso\_app

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 类型 | 限定条件 | 说明 |
| id | varchar | 主键 |  |
| client\_name | varchar | 非空 | 客户名 |
| client\_id | varchar | 非空 | 客户ID |
| client\_secret | varchar | 非空 | 登录秘钥 |
| redirect\_url | varchar | 非空 | 回调地址 |
| create\_time | datetime |  | 创建时间 |

第三方授权表 auth\_history

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 类型 | 限定条件 | 说明 |
| id | varchar | 主键 |  |
| user\_id | varchar | 非空 | 用户ID 外键 |
| client\_id | varchar | 非空 | 客户ID 外键 |
| create\_time | datetime |  | 创建时间 |

历史密码表 password\_history

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 类型 | 限定条件 | 说明 |
| id | varchar | 主键 |  |
| user\_id | varchar | 非空 | 用户ID 外键 |
| password | varchar | 非空 | 登录密码 |
| create\_time | datetime |  | 创建时间 |