单点登录系统设计方案：

系统架构图：

MySQL

后台

客户端

gRPC-Node

gRPC-C++

DJinni-Java

DJinni-C++

gRPC

整个系统分为两部分：

使用Node实现的后台svr以及使用Android实现的客户端。

前后端通信通过gRPC实现，使用RSA、md5、加盐来对数据传递进行处理；数据存储通过mysql实现。

客户端核心逻辑通过C++实现，通过DJinni提供不同平台的接口，从而实现跨平台。界面通过Android实现。

功能设计：

1. 用户注册功能。客户端使用svr的公钥加密，将密码传递给svr，svr私钥解密后存储到数据库。
2. 用户登录功能。客户端将密码与时间戳计算md5，传递给svr，svr查数据库得到密码，并通过时间戳计算md5，并与客户端传递的值进行校验。
3. 单点登录功能。用户登录后，svr在内存中保留所有用户的登录call，在用户登录时，先检查该用户是否已经登录过，如果有，则登出原有的用户，并让新的请求登录成功。
4. 按规则踢出用户。Svr端通过mysql记录规则，每分钟查询一次，将符合规则的用户踢出。

系统设计说明

1. 用户注册模块：

使用gRPC的普通的单项请求。

该模块主要实现的功能：

1. 用户信息存储。使用动态生成的盐与用户密码拼接后，进行sha256，将结果和动态的盐记录在数据库中。
2. 网络传输注册信息。注册信息比较敏感，需要在Https的基础上，增加Rsa的加密，客户端将密码以及时间戳拼接在一起后，使用服务端的公钥进行加密，之后将时间戳以及加密串发送给服务端。由于有时间戳可以保证每次加密后得到的内容不同。（RSA算法使用的psdk1，加密相同内容得到的结果每次都不同）
3. 单点登录模块。

使用gRPC的steam-to-steam模式。

主要功能有：

* 1. 用户登录功能。客户端先发起请求获得服务端返回的token和数据库中存储的盐，并通过sha256进行计算，将结果发送给服务端，服务端校验通过后则完成登录。
  2. 用户单点在线。服务端将所有登录成功的用户的连接（ServerReaderWriter）记录下来。每当新的登录请求校验成功后，遍历所有已经登录的用户，并向之前的连接发送登出命令，之后将新的连接进行记录，并返回登录成功状态。

1. 按规则踢出用户。

使用gRPC的steam-to-steam模式。

主要功能有：

1. 定期遍历。通过创建一个新线程以及slee()功能，来实现循环任务的功能。
2. 规则配置。在数据库中的user\_forbid表中配置需要踢出的用户规则（正则表达式），在执行循环任务时，svr会先查询有哪些需要用户规则，然后遍历所有已经登录的用户列表，将匹配成功的用户踢出。

**代码框架：**

系统主框架部分先编写proto文件，再通过gRPC工具生成不同平台的对应代码。这里客户端与服务端都使用CPP部分的代码。

1. 服务端。服务端实现proto中定义的方法，并通过mysql API与数据库连接。从而完成上文提到的几个模块的功能。
2. 客户端。客户端实现proto中定义的方法，实现相应的请求的封装以及服务端响应的处理。

客户端接口部分在通过djinni定义方便java和object-c调用和回调的客户端代码。本系统中通过异步的callback来实现回调。