



# 概述

Casdoor是一个基于OAuth 2.0、OIDC、SAML 和 CAS 的，UI-first的身份和访问管理(IAM)/单点登录(SSO)平台。

You need to enable JavaScript to run this app.

Casdoor 可为网页UI和应用程序用户的登录请求提供服务。

# Casdoor features

1. 前后端分离的架构，使用 Go 语言开发，Casdoor 支持高并发，提供基于Web的用户管理 UI，并支持中、英等多种语言。
2. Casdoor支持第三方应用登录，如GitHub、谷歌、QQ、微信等，并支持通过插件扩展第三方登录。
3. 使用 [Casbin](#) 基于授权管理，Casdoor 支持 ACL, RBAC, ABAC, RESTful 等访问控制模型。
4. Casdoor可通过手机验证码、邮箱验证码登录，还有找回密码功能。
5. 审查和记录访问日志。
6. 使用阿里云、腾讯云、七牛云提供的图片CDN云存储。
7. 个性化的注册、登录和找回密码页面。
8. Casdoor 支持通过数据库同步的方法来与现有系统集成，用户可以顺利过渡到 Casdoor。
9. Casdoor 支持主流数据库: MySQL、PostgreSQL、SQL Server 等，并支持扩展插件的新数据库。

# How it works



## 步骤0（前置知识）

1. Casdoor 的授权程序建立在 OAuth 2 的基础上。因此，强烈建议简单了解 OAuth 2.0 的工作原理。详见[OAuth 2.0 简介](#)。

## Abstract Protocol Flow



### 步骤 1 (授权请求)

您的应用（或网站等）应该以 `endpoint/login/oauth/authorize?client_id=xxx&response_type=code&redirect_uri=xxx&scope=read&state=xxx` 这种格式编写 URL。在此 URL 中，使用 Casdoor 的主机 URL 替换 `endpoint`，并用您的信息替换 `xxx`。

#### ① 提示

对于 `xxx` 的部分需要写上什么？

- 对于 `client_id`: 您可以在每个应用程序里找到它
- 对于 `redirect_uri`: 您应该将此设置为自己的应用程序回调URL, 通过这个信息, Casdoor 可以知道在授权后向哪里发送信息
- 对于 `state`: 您应该用您的应用程序名称填写这个内容

应用程序对用户说：“嘿，现在我需要一些资源，我需要您的许可才能拿这些资源。您愿意跳转到这个URL，填写您的用户名和密码吗？”

使用正确的URL，您的应用程序将会让用户向此 URL 发起请求，`授权请求` 已完成。

## 步骤 2（授权认证）

此步骤比较直接：用户会被重定向到上面的URL，看到来自Casdoor的登录页面。通过在登录页面输入正确的用户名和认证信息，Casdoor现在能成功识别用户，将发送`代码`和`状态`返回第1步中设置的回调URL。

用户打开网址并向Casdoor提供凭据。Casdoor说：“好～这是我在数据库中知道的用户（授权应用获取`code`和`state`）。然后我将使用回调URL发送`code`和`state`返回应用`(redirect_uri)`”

这两个关键词发到您的应用后，应用便得到授权，至此就完成了`授权`环节。



提示

Casdoor也提供第三方登录。在这种情况下，您将不会看到输入验证信息的页面，只会看到第三方提供商列表。您可以使用这些提供商登录到您的应用，而Casdoor是中间层（中间件）。

## 步骤 3（授权认证）

在这一步，您的应用程序已经有了来自第2步的代码，它将会告诉Casdoor：“嘿，现在用户同意给我`code`，你想检查这个`code`并给我`access_token`吗？

## 步骤 4（访问令牌）

这一步骤中，Casdoor会通知应用程序：“这个`code`应该是合法的，你一定就是那个正确的应用程序。这是`access_token`。”

有了`code`，Casdoor知道它是一个已授权的应用（第二步中用户给予的授权），试图获取`access_token`（稍后将会用来获取更有用的信息）。

## 步骤 5（访问令牌）

在这个步骤中，您的应用程序说：“很好，刚刚获得了最新的`access_token`我现在可以使用它从资源服务器获得更多宝贵的东西！”

您的应用程序转向 资源服务器：“嗨朋友，想检查这个 `access_token` 吗？我从 Casdoor 得到了访问令牌，你想看看这是否与 Casdoor 的一致吗？”

## 步骤 6 (受保护资源)

Resource Server 又告知您的应用程序：“不错~ 它似乎就像我在 Casdoor 中的那个一样。Casdoor 说谁拥有这个 `access_token` 谁就可以拥有这些 受保护的资源 现在，你可以自取这些资源了！”

这就是Casdoor 如何与您的应用程序一起工作。

### ① 提示

Casdoor 可以同时运行 认证服务器 和 资源服务器 配件，也就是说：Casdoor 授权我们的应用程序从Casdoor的数据库获取资源（例如通常是当前登录用户的信息）。

## 在线演示

### Casdoor

这里是一个由Casbin部署的在线演示。

- [Casdoor官方演示](#)

全局管理员登录：

- 用户名：`admin`
- 密码：`123`

### Casbin-OA

Casbin-OA是Casbin 的web 应用程序之一。它使用 Casdoor 作为身份验证。

- [Casbin-OA](#)
- Source code: <https://github.com/casbin/casbin-oa>

## Casnnode

Casnnode 是Casbin社区开发的官方论坛。

它使用 Casdoor 作为认证平台并管理成员。

- [Casnnode](#)
- Source code: <https://github.com/casbin/casnnode>

## 结构

Casdoor包含两个部分：

名称	描述	语言	源代码
前端	Casdoor的前端 Web界面	JavaScript + React	<a href="https://github.com/casdoor/casdoor/tree/master/web">https://github.com/casdoor/casdoor/tree/master/web</a>
后端	Casdoor后端 RESTful API	Golang + Beego + SQL	<a href="https://github.com/casdoor/casdoor">https://github.com/casdoor/casdoor</a>

# 核心概念

作为Casdoor的管理员，您至少应该熟悉4个核心概念：组织(Organization)，用户(User)，应用(Application) 和 提供商(Provider)。



提示

In the following parts, we will use the demo site: <https://door.casdoor.com> as example.

## 组织

在Casdoor中，组织是用户和应用程序的容器。例如，一个公司的所有雇员或一个企业的所有客户都可以抽象成为一个组织。组织的类别定义如下：

```
type Organization struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`

    DisplayName     string `xorm:"varchar(100)" json:"displayName"`
    WebsiteUrl     string `xorm:"varchar(100)" json:"websiteUrl"`
    Favicon        string `xorm:"varchar(100)" json:"favicon"`
    PasswordType    string `xorm:"varchar(100)" json:"passwordType"`
    PasswordSalt    string `xorm:"varchar(100)" json:"passwordSalt"`
    PhonePrefix     string `xorm:"varchar(10)" json:"phonePrefix"`
    DefaultAvatar   string `xorm:"varchar(100)" json:"defaultAvatar"`
    Tags           []string `xorm:"mediumtext" json:"tags"`
    MasterPassword  string `xorm:"varchar(100)" json:"masterPassword"`
    EnableSoftDeletion bool `json:"enableSoftDeletion"`
    IsProfilePublic bool `json:"isProfilePublic"`

    AccountItems []*AccountItem `xorm:"varchar(2000)" json:"accountItems"`
}
```

## 用户

Casdoor 中的用户可以登录到一个应用程序。一个用户只能属于一个组织，但可以登录到该组织拥有的多个应用程序。Casdoor目前有两种类型的用户：

- 内置用户(built-in 组织下的所有用户)，例如 `built-in/admin`：全局管理员，在Casdoor平台上拥有完整的管理员权限。
- 其他组织下的用户，如 `my-company/Alice`：普通用户，只能注册、登录、登出、更改他/她自己的个人资料等。

在 Casdoor API 中，用户通常被定义为 `<organization_name>/<username>`，e.g., Casdoor的默认管理员被定义为 `built-in/admin`。用户名权限中有一个名为 `id` 的属性，这是一个类似 `d835a48f-2e88-4c1f-b907-60ac6b6c1b40` 的 UUID，它也可以是用户通过一个应用程序选择的 ID。



提示

如果您的应用程序仅应用于一个组织，您可以使用 `<username>` instead of `<organization_name>/<username>` 作为您整个应用程序的用户ID。

用户的类别定义如下：

```
type User struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`
    UpdatedTime string `xorm:"varchar(100)" json:"updatedTime"
```

# 应用程序

应用程序是指需要受Casdoor保护的网络服务。例如，论坛网站、OA系统、CRM系统都属于应用程序。

```
type Application struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`

    DisplayName     string      `xorm:"varchar(100)" json:"displayName"`
    Logo           string      `xorm:"varchar(100)" json:"logo"`
    HomepageUrl   string      `xorm:"varchar(100)" json:"homepageUrl"`
    Description    string      `xorm:"varchar(100)" json:"description"`
    Organization   string      `xorm:"varchar(100)" json:"organization"`
    Cert          string      `xorm:"varchar(100)" json:"cert"`
    EnablePassword bool        `json:"enablePassword"`
    EnableSignUp   bool        `json:"enableSignUp"`
    EnableSigninSession bool      `json:"enableSigninSession"`
    EnableCodeSignin  bool      `json:"enableCodeSignin"`
    Providers     []*ProviderItem `xorm:"mediumtext" json:"providers"`
    SignupItems   []*SignupItem  `xorm:"varchar(1000)" json:"signupItems"`
    OrganizationObj *Organization `xorm:"-" json:"organizationObj"`

    ClientId      string      `xorm:"varchar(100)" json:"clientId"`
    ClientSecret   string      `xorm:"varchar(100)" json:"clientSecret"`
    RedirectUris  []string    `xorm:"varchar(1000)" json:"redirectUris"`
    TokenFormat    string      `xorm:"varchar(100)" json:"tokenFormat"`
    ExpireInHours int         `json:"expireInHours"`
    RefreshExpireInHours int      `json:"refreshExpireInHours"`
    SignupUrl     string      `xorm:"varchar(200)" json:"signupUrl"`
    SigninUrl     string      `xorm:"varchar(200)" json:"signinUrl"`
    ForgetUrl     string      `xorm:"varchar(200)" json:"forgetUrl"`
    AffiliationUrl string      `xorm:"varchar(100)" json:"affiliationUrl"`
    TermsOfUse    string      `xorm:"varchar(100)" json:"termsOfUse"`
    SignupHtml    string      `xorm:"mediumtext" json:"signupHtml"`
    SigninHtml    string      `xorm:"mediumtext" json:"signinHtml"`
}
```

每个应用程序都可以有自己的自定义注册页面，登录页等。E.g., the root login page `/login` (like: <https://door.casdoor.com/login>) is the sign in page only for Casdoor's built-in application: `app-built-in`.

应用程序是用户登录到Casdoor的“入口”或“界面”。用户需要通过一个应用程序的登录页面才能登录到Casdoor。

应用程序	注册页面网址	登录页面网址
内置应用程序	<a href="https://door.casdoor.com/signup">https://door.casdoor.com/signup</a>	<a href="https://door.casdoor.com/login">https://door.casdoor.com/login</a>
casnode 论坛系统	<a href="https://door.casdoor.com/signup/app-casnode">https://door.casdoor.com/signup/app-casnode</a>	<a href="https://door.casdoor.com/login/oauth/authorize?client_id=014ae4bd048734ca2dea&amp;response_type=code&amp;redirect_uri=http://localhost:9000/callback&amp;scope=read&amp;state=casdoor">https://door.casdoor.com/login/oauth/authorize?client_id=014ae4bd048734ca2dea&amp;response_type=code&amp;redirect_uri=http://localhost:9000/callback&amp;scope=read&amp;state=casdoor</a>
casbin 的OA系统	<a href="https://door.casdoor.com/signup/app-casbin-oa">https://door.casdoor.com/signup/app-casbin-oa</a>	<a href="https://door.casdoor.com/login/oauth/authorize?client_id=0ba528121ea87b3eb54d&amp;response_type=code&amp;redirect_uri=http://localhost:9000/callback&amp;scope=read&amp;state=casdoor">https://door.casdoor.com/login/oauth/authorize?client_id=0ba528121ea87b3eb54d&amp;response_type=code&amp;redirect_uri=http://localhost:9000/callback&amp;scope=read&amp;state=casdoor</a>

## 登录 URL

It's very easy to log into Casdoor via Casdoor's built-in application, just visit Casdoor server's homepage (like: <https://door.casdoor.com> for demo site) and it will automatically redirect you to `/login`. 但如何在前端和后端代码中为其他应用程序获取这些URL? 您可以将您自己的字符串连接起来, 也可以调用 Casdoor SDK 提供的一些实用功能来获取 URL:

### 1. By concatenating string manually

- 注册页面URL
  - 注册指定的应用程序: `<your-casdoor-hostname>/signup/<your-application-name>`
  - 通过OAuth注册: `<your-casdoor-hostname>/signup/oauth/authorize?client_id=<client-id-for-your-application>&response_type=code&redirect_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor`
  - 自动注册: `<your-casdoor-hostname>/auto-signup/oauth/authorize?client_id=<client-id-for-your-application>&response_type=code&redirect_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor`
- 登录页面URL
  - 登录指定的组织: `<your-casdoor-hostname>/login/<your-organization-name>`
  - 通过OAuth登录: `<your-casdoor-hostname>/login/oauth/authorize?client_id=<client-id-for-your-application>&response_type=code&redirect_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor`

### 2. Use frontend SDK (for frontend Javascript code using React, Vue or Angular)

`getSignupUrl()` and `getSigninUrl()`: [casdoor-js-sdk](#)

### 3. Use backend SDK (for backend code using Go, Java, etc.)

`GetSignupUrl()` and `GetSigninUrl()`: [casdoor-go-sdk](#)

## 提供商

Casdoor是一个联合单点登录系统, 通过OIDC、OAuth 和SAML支持多个身份提供者。Casdoor 也可以通过电子邮件或短信(快速消息服务) 向用户发送验证码或其他通知。Casdoor 使用这个概念: `Provider` 来管理所有这些第三方连接器。

Currently, All providers supported by Casdoor can be found here: [provider/overview](#)

```
type Provider struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`

    DisplayName  string `xorm:"varchar(100)" json:"displayName"`
    Category    string `xorm:"varchar(100)" json:"category"`
    Type        string `xorm:"varchar(100)" json:"type"`
    Method      string `xorm:"varchar(100)" json:"method"`
    ClientId    string `xorm:"varchar(100)" json:"clientId"`
    ClientSecret string `xorm:"varchar(100)" json:"clientSecret"`
    ClientId2   string `xorm:"varchar(100)" json:"clientId2"`
    ClientSecret2 string `xorm:"varchar(100)" json:"clientSecret2"`

    Host     string `xorm:"varchar(100)" json:"host"`
    Port     int     `json:"port"`
    Title    string `xorm:"varchar(100)" json:"title"`
    Content  string `xorm:"varchar(1000)" json:"content"`

    RegionId   string `xorm:"varchar(100)" json:"regionId"`
    SignName    string `xorm:"varchar(100)" json:"signName"`
    TemplateCode string `xorm:"varchar(100)" json:"templateCode"`
    AppId      string `xorm:"varchar(100)" json:"appId"
```

## Casdoor是如何自我管理的?

当您首次运行Casdoor时， Casdoor将创建一些内置的对象来帮助管理员管理Casdoor本身：

- 一个内置的命名为 `built-in` 的组织。
- `built-in` 组织下用户名为 `admin` 的用户。
- 一个内置的应用程序名为 `app-built-in`，由 `built-in` 组织所拥有，代表Casdoor 本身(实际上也是一个应用程序)。

`built-in` 组织中的所有用户，包括 `admin` 默认情况下将在Casdoor平台上拥有完整的管理员权限。所以，如果你有多个管理员，可在 `built-in` 机构下创建新帐户。否则，请记住关闭 `app-built-in` 应用程序的注册功能。

### ⚠ 注意事项

内置对象已被禁止在网页UI或 RESTful API中重命名或删除。 Casdoor在许多地方硬编码了这些保留的名称。不要试图重命名或删除它们，比如修改数据库，否则整个系统可能崩溃。



# 服务器安装

## 安装要求

### 操作系统

支持所有主流的操作系统，包括Windows、Linux和macOS。

### 环境

- Go 1.17+
- Node.js LTS (18)
- Yarn 1.x

#### 信息

我们强烈建议您使用 [Yarn 1.x](#) 运行 & Casdoor 前端，使用 NPM 可能会导致 UI 风格问题。更多详细信息见：[Casdoor#294](#)

#### 注意事项

对于中国大陆用户，为了成功下载依赖关系包，您需要通过配置 GOPROXY 环境变量来使用 Go 代理。We strongly recommend: <https://goproxy.cn/>

## 数据库

Casdoor 使用 [XORM](#) 与数据库进行交互。基于 [Xorm Drivers Support](#)，当前支持的数据包括：

- MySQL
- MariaDB
- PostgreSQL
- CockroachDB
- SQL Server
- Oracle
- SQLite 3
- TiDB

## 下载

The source code of Casdoor is hosted at GitHub: <https://github.com/casdoor/casdoor>. Go 后端代码和 React 前端代码都在单个仓库中。

名称	描述	语言	源代码
前端	Casdoor的网页前端界面	JavaScript + React	<a href="https://github.com/casdoor/casdoor/tree/master/web">https://github.com/casdoor/casdoor/tree/master/web</a>
后端	Casdoor的ResTful API 后端	Golang + Beego + XORM	<a href="https://github.com/casdoor/casdoor">https://github.com/casdoor/casdoor</a>

Casdoor支持 Go Modules。要下载代码，您直接通过git克隆仓库就可以了：

```
cd /文件夹路径/  
git clone https://github.com/casdoor/casdoor
```

# 配置

## 配置数据库

Casdoor支持MySQL, msSQL, Sqlite3, PostgreSQL等数据库。默认使用MySQL。如果您想使用支持以外的数据库, 请自行修改object/adapter包

### MySQL

Casdoor将会把users, nodes和topics信息存储在一个命名为casdoor的MySQL数据库中。如果数据库不存在, 则需手动创建。The DB connection string can be specified at: <https://github.com/casdoor/casdoor/blob/master/conf/app.conf>

```
driverName = mysql
dataSourceName = root:123456@tcp(localhost:3306)-
dbName = casdoor
```

### PostgreSQL

因为在使用xorm打开Postgres时我们必须选择一个数据库, 所以您应该在运行Casdoor前手动准备一个数据库

假设您已经准备了一个名为casdoor的数据库, 您应该这样编写app.conf:

```
driverName = postgres
dataSourceName = "user=postgres password=postgres host=localhost
port=5432 sslmode=disable dbname=casdoor"
dbName =
```

### ① 信息

对于PostgreSQL，请确保 `dataSourceName` 中的 `dbName` 不是空值，并且与上面例子一样将单独 `dbName` 配置留空

## CockroachDB

你也可以通过`postgres`驱动程序访问`cockroachdb`。它的配置与 `postgreSQL` 相同。

```
driverName = postgres
dataSourceName = "user=postgres password=postgres host=localhost
port=5432 sslmode=disable dbname=casdoor
serial_normalization=virtual_sequence"
dbName =
```

### ① 信息

For CockroachDB, don't forget to add

`serial_normalization=virtual_sequence` to the `dataSourceName` like the above example. otherwise you will get error regarding existed database, whenever the service started or restarted. Notice, this must be added before the database created.

## Sqlite3

您应该像这样配置 `app.conf`:

```
driverName = sqlite
dataSourceName = "file:casdoor.db?cache=shared"
dbName = casdoor
```

## 通过 Ini 文件配置

Casdoor can be configured via a single file: [conf/app.conf](#), the content of which by default is:

```
appname = casdoor
httpport = 8000
runmode = dev
SessionOn = true
copyRequestBody = true
driverName = mysql
dataSourceName = root:123456@tcp(localhost:3306)-
dbName = casdoor
tableNamePrefix =
showSql = false
redisEndpoint =
defaultStorageProvider =
isCloudIntranet = false
authState = "casdoor"
socks5Proxy = "127.0.0.1:10808"
verificationCodeTimeout = 10
initScore = 2000
logPostOnly = true
origin = "https://door.casdoor.com"
staticBaseUrl = "https://cdn.casbin.org"
enableGzip = true
```

- `appname` 是应用程序名称，目前没有实际使用
- `httpport` 是您后端应用程序正在监听的端口
- `runmode` 是 `dev` 或 `prod`
- `SessionOn` 控制是否启用会话，默认为使用。
- `driverName`, `dataSourceName` 和 `dbName` 之前已经介绍，请参阅 [Configure Database](#).

- `verificationCodeTimeout` 设置验证码到期时间。 超时之后用户需要再次获取验证码。

Despite all the configurable fields, as a beginner, you only need to modify two items: `driverName` and `dataSourceName` based on your database. This database will be used by Casdoor to store all data, including users, organizations, applications and so on.

- `tableNamePrefix` 是使用适配器时表格的前缀。
- `showSql` : 如果日志级别大于INFO 则显示 SQL 语句或不在 Logger 上。
- `redisEndpoint` 后填写Redis地址，被Beego用于session存储 如果此参数为空，session数据将被储存在 `./tmp` 文件夹中 使用Redis作为Beego的session存储，此参数的示例为: `redis.example.com:6379`。 If Redis is deployed in the local machine, you can use `localhost:6379`. If Redis password is enabled, use `redis.example.com:6379, db, password`. See more details at: <https://github.com/beego/beedoc/blob/master/en-US/module/session.md#saving-provider-config>
- `defaultStorageProvider` 是默认的文件存储服务名称。 如果您需要使用文件存储服务，例如 `头像上传`， 您需要设置存储提供商，并在您的 `应用` 中应用它。 详情请参阅 [存储](#)
- `isCloudIntranet` 用于确定您的提供者端是否是内联网端点。
- `authstate` 是授权应用程序名称。 登录时将检查该参数。
- `socks5Proxy` 是 SOCKS 代理服务器 IP 地址。 [设置代理端口](#)， 因为我们有与谷歌相关的服务或使用 `Google` `GitHub` `Facebook` `LinkedIn` `Steam` 作为OAuth Provider，在某些领域将受到网络限制。
- `initscore` 是每个用户的最初分数。 每个用户都有一个得分属性。 得分被 [Casnode](#) 所使用。 分数无法控制 Casdoor 中的任何东西。
- `logPostOnly` 用于识别是否只使用帖子方法添加记录。
- `origin` 是origin形式的后端域名
- `staticBaseUrl` 是系统初始化数据库时静态图像的地址。

- `enableGzip` 当请求头带有 `Accept-Encoding=gzip` 标志时，将会接受并返回通过 gzip 编码的响应。

## 通过环境变量配置

All configuration items defined by Casdoor in the ini file mentioned above can be chosen to configuration via environmental variables, so can some of the beego configurations items(`httpport`,`appname`).

For example, when you try to start casdoor, you can use something like this to pass the configuration via environmental variables.

```
appname=casbin go run main.go
```

Besides, `export` derivatives are also a possible method. The names of environmental variables should be exactly the same with the names you want to use in the ini file.

注意: 环境变量中的配置会覆盖 *ini* 文件中的配置。

## 运行

当前有两种启动方法，您可以根据自己的情况选择一个。

### 开发模式

#### Backend

Casdoor's Go backend runs at port 8000 by default. You can start the Go backend with the following command:

```
go run main.go
```

After the server is successfully running, we can start the frontend part.

## Frontend

Casdoor's frontend is a very classic [Create-React-App \(CRA\)](#) project. It runs at port `7001` by default. Use the following commands to run the frontend:

```
cd web  
yarn install  
yarn start
```

Visit: <http://localhost:7001> in your browser. Log into Casdoor dashboard with the default global admin account: `built-in/admin`

```
admin  
123
```

# 生产模式

## Backend

Build Casdoor Go backend code into executable and start it.

For Linux:

```
go build  
.casdoor
```

For Windows:

```
go build  
casdoor.exe
```

## Frontend

Build Casdoor frontend code into static resources (.html, .js, .css files):

```
cd web  
yarn install  
yarn build
```

Visit: <http://localhost:8000> in your browser. Log into Casdoor dashboard with the default global admin account: `built-in/admin`

```
admin  
123
```



提示

To use another port, please edit `conf/app.conf` and modify `httpport`, then restart the Go backend.

### ❗ CASDOOR PORT DETAILS

In dev environment, the frontend is run by `yarn run` in port 7001, so if you want to go to Casdoor login page, you need set Casdoor link as <http://localhost:7001>.

In prod environment, the frontend files is first built by `yarn build` and served in port 8000, so if you want to go to Casdoor login page, you need

to set Casdoor link as <https://your-casdoor-url.com:8000> (If you are using reverse proxy, you need to set the link as your domain).

Take our official forum Casnode as an example

Casnnode uses Casdoor to handle authentication.

When we are testing Casnode in dev environment, we set the `serverUrl` as <http://localhost:7001>, so when we test signin and signup functionality using Casdoor, it will go to localhost 7001 which is the Casdoor port.

And when we put Casnode to prod environment, we set the `serverUrl` as <https://door.casdoor.com>, so users can signin or signup using Casdoor.

```
14 import * as ConfBackend from "./backend/ConfBackend.js"
15
16 export const AuthConfig = {
17   // serverUrl: "https://door.casbin.com",
18   serverUrl: "http://localhost:7001",
19   clientId: "014ae4bd048734ca2dea",
20 }
```



# (可选) 使用 Docker 运行

## 安装要求

### 硬件

如果您想要自己构建Docker镜像, 请确保您的机器至少有**2GB** 的内存。Cassdoor的前端是React的一个NPM项目。构建前端至少需要 **2GB** 的内存。低于 **2GB** 的内存可能导致前端构建失败。

如果您只需要运行预编译的镜像, 请确保您的机器至少有**100MB** 的内存。

### 操作系统

支持所有操作系统 (Linux, Windows 和 macOS)

### Docker

在Linux系统中, 您可以使用**docker** (要求**docker-engine** 版本不低于 17.05), 在 Windows和mac系统中可以使用 **Docker Desktop**。

- [Docker](#)

所有操作系统的用户必须确保 **docker-engine** 版本不低于17.05。这是因为我们在 **docker-compose.yml** 中使用多阶段构建功能, 这个功能在17.05及以上版本中得到支持。更多信息请参阅 <https://docs.docker.com/develop/develop-images/multistage-build/>

如果你使用**docker-compose**, 请确保 **docker-compose** 版本不低于2.2 对于Linux用

户，考虑到docker-compose与docker-engine相分离，你还需要确保已安装docker-compose。

## 获取镜像

我们提供了两个DockerHub 镜像：

名称	描述	建议
casdoor-all-in-one	Casdoor 和 MySQL 数据库都在镜像内	已经包含示例数据库，仅用于测试
casdoor	只有casdoor在镜像里	可以连接到您自己的数据库并用于生产

1. 在casbin/casdoor-all-in-one中，casdoor二进制文件、mysql数据库和所有必要的配置都打包在一起。这个镜像是为了让新用户能够快速试用Casdoor。通过这个镜像，您可以使用一两个命令而不是复杂的配置即可启动Casdoor。注意：我们不建议您在生产环境中使用此镜像。

### 选项 1: 使用示例数据库

使用端口 8000 运行容器。如果本地host中不存在，它将自动下载镜像。

```
docker run -p 8000:8000 casbin/casdoor-all-in-one
```



注意事项

中国等地区的一些用户通常使用 Docker 镜像服务，例如 [Alibaba Cloud Image Booster \(English\)](#) 实现比DockerHub 更高的下载速度。然而，有一个问题，那些服务提供的 `latest` 标签不是最新的。它可能会通过获取 `latest` 标签来产生一个非常旧的镜像。为了缓解这个问题，您可以使用以下命令明确指定镜像版本号：

```
docker pull casbin/casdoor-all-in-one:$(`curl -ss "https://hub.docker.com/v2/repositories/casbin/casdoor-all-in-one/tags/?page_size=1&page=2" | sed 's/,/,\\n/g' | grep '"name"' | awk -F '"' '{print $4}'`)
```

注意：上面的命令使用的 Linux 工具，例如 `curl`, `ed`, `grep`, `awk`. 如果您正在使用 Windows，请确保您在 Linux 样式外壳中运行，如 `Git Shell` 或 `Cygwin`。`CMD` 或 `PowerShell` 将无法工作。

在您的浏览器中访问：<http://localhost:8000>。 使用默认的全局管理员帐户登录 Casdoor 仪表板：`built-in/admin`

admin

123

## 选项 2: 使用 docker-compose

### ⚠ 注意事项

中国等地区的一些用户通常使用 Docker 镜像服务，例如 [Alibaba Cloud Image Booster \(English\)](#) 实现比DockerHub 更高的下载速度。然而，有一个问题，那些服务提供的 `latest` 标签不是最新的。它可能会通过获取 `latest` 标签来产生一个非常古老的图像。为了缓解这个问题，您可以使用以下命令明确指定图像

版本号:

```
docker pull casbin/casdoor:$(  
curl -sS  
"https://hub.docker.com/v2/repositories/casbin/casdoor/  
tags/?page_size=1&page=2" | sed 's/,/,\\n/g' | grep '"name"'  
| awk -F '"' '{print $4}' )
```

注意: 上面的命令使用的 Linux 工具, 例如 curl, ed, grep, awk。如果您正在使用 Windows, 请确保您在 Linux 样式外壳中运行, 如 Git Shell 或 Cygwin。CMD 或 PowerShell 将无法工作。

在与 docker-compose.yml 同级目录下创建 conf/app.conf 文件, 然后从 Casdoor 复制 app.conf。您可以从 [Via Ini file](#) 找到 app.conf 的详细信息。

通过 docker-compose 创建一个独立的数据库:

```
docker-compose up
```

这样就完成了! ✨

在您的浏览器中访问 <http://localhost:8000>。 使用默认的全局管理账户登录 Casdoor 控制面板: built-in/admin

```
admin  
123
```

注意: 如果你深入了解 docker-compose.yml, 你可能会对我们创建的称为 "RUNNING\_IN\_DOCKER" 的环境变量感到困惑。当数据库 'db' 是通过 docker-compose 创建时, 它可以在您的 pc 的本地主机上使用, 而不是连带容器的本地主机上使用。这是为了防止你因修改应用程序而引起问题。我们提供这个环境变量, 并且在 docker-

`compose.yml`中预先分配了它。当此环境变量为 `true` 时，本地主机将被替换为 `host.docker.internal`，以便您可以访问 `db`。

## 选项3 直接尝试使用标准镜像

### ⚠ 注意事项

中国等地区的一些用户通常使用 Docker 镜像服务，例如 [Alibaba Cloud Image Booster \(English\)](#) 实现比 DockerHub 更高的下载速度。然而，有一个问题，那些服务提供的 `latest` 标签不是最新的。它可能会通过获取 `latest` 标签来产生一个非常旧的镜像。为了缓解这个问题，您可以使用以下命令明确指定镜像版本号：

```
docker pull casbin/casdoor:$(  
curl -sS  
"https://hub.docker.com/v2/repositories/casbin/casdoor/  
tags/?page_size=1&page=2" | sed 's/,/,\\n/g' | grep '"name"'  
| awk -F '"' '{print $4}' )
```

注意：上面的命令使用的 Linux 工具，例如 `curl`, `sed`, `grep`, `awk`。如果您正在使用 Windows，请确保您在 Linux 样式外壳中运行，如 `Git Shell` 或 `Cygwin`。`CMD` 或 `PowerShell` 将无法工作。

### 💡 提示

如果不方便挂载配置文件，使用环境变量也是一种可能的解决方法。

#### example

```
docker run \  
-e driverName=mysql \  
...
```

创建 `conf/app.conf` 文件，您可以从 `conf/app.conf` 复制。关于 `app.conf` 的更多信息，您可以访问 [Via Ini file](#).

然后运行

```
docker run -p 8000:8000 -v /folder/of/app.conf:/conf casbin/casdoor:latest
```

将 `app.conf` 挂载到 `/conf/app.conf` 并启动它。

使用您的浏览器访问 `http://localhost:8000` 使用默认的全局管理账户登录 Casdoor 控制面板： [built-in/admin](#)

```
admin  
123
```

# Casdoor公共API

Casdoor采用前后端分离的模式开发（与JSP或PHP不同）。只有通过 RESTful API才能显示其功能。React前端代码通过调用RESTful API来渲染Web UI并执行操作。这个 RESTful API接口被称为 `Casdoor Public API`。这个API被用在以下几个地方：

- Casdoor前端页面
- Casdoor Client SDK
- 应用方自定义的任何代码

`Casdoor Public API`的完整参考文档可以在 <https://door.casdoor.com/swagger> 中查看。这个Swagger文档是由Beego的Bee工具自动生成的。

# 教程

## 产品文档

产品	技术	文档
Dashboard of PingCAP TiDB	React + Typescript + Go + Gin	<a href="#">Use Casdoor for TiDB Dashboard SSO sign-in</a> (other languages: <a href="#">Chinese</a> , <a href="#">Japanese</a> )
GitLab	Vue + Ruby + Rails	<a href="#">OpenID Connect OmniAuth provider</a>
Apache Shenyu	Java	<a href="#">Casdoor Plugin</a> (other languages: <a href="#">Chinese</a> )
Alist	Typescript + SoildJS + Go + Gin	<a href="#">Casdoor SSO</a> (other languages: <a href="#">Chinese</a> )
BookStack	jQuery + Bootstrap + Go + Beego	<a href="#">Casdoor integrates registration and login</a>

# 文章

技术	语言	标题
ASP.Net Core 6	英语	ASP.Net Core .net 6 Demo Authentication Project using local Casdoor Docker Container on Windows Subsystem for Linux
OAuth2 Proxy (Go)	中文	Use Casdoor + OAuth-Proxy to protect web applications on public networks
Casnode (Javascript + React + Go + Beego)	中文	Use Lighthouse to set up a forum like v2ex
Cloudreve (Go)	中文	Modify Cloudreve to support Casdoor
KodExplorer (PHP)	中文	Modify KodExplorer to support Casdoor



&gt;

Deployment

# Deployment

## 数据初始化

如何从文件初始化Casdoor 数据

## 将静态文件托管到CDN

在CDN中托管前端静态文件

## 将静态文件托管在局域网内

如何部署Casdoor静态资源

## 数据库迁移

处理Casdoor中的 数据库迁移

# 数据初始化

如果您将Casdoor和其他服务作为一个应用整体进行部署， 您可能想要为用户提供开箱即用的功能(用户不需要任何配置就可以直接使用应用程序)。

在这种情况下，您可以使用数据初始化功能，通过一个配置文件将您服务注册到Casdoor。此文件可以由服务所有者预定义或动态生。

## 使用方式

如果在 Casdoor 的根目录下有一个名为 `init_data.json` 的配置文件，它将被用于初始化Casdoor中的数据。您要做的只是将此文件放到运行Casdoor的根目录下。

如果您使用 Casdoor 的官方 `docker` 镜像，以下脚本可以帮助您将 `init_data.json` 挂载到容器里。

### Docker

如果您使用 `docker` 部署 Casdoor，您可以使用 `卷` 将 `init_data.json` 挂载到容器中。

```
docker run ... -v /path/to/init_data.json:/init_data.json
```

### Kubernetes

如果您使用 Kubernetes 部署了 Casdoor，您可以使用 `configmap` 来存放 `init_data.json`。

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: casdoor-init-data
data:
  init_data.json:
```

您可以通过挂载 configmap 将数据挂载到Casdoor的 pods 中。 您可以按照如下例子修改 deployment 的定义文件：

```
apiVersion: apps/v1
kind: Deployment
...
spec:
  template:
    ...
      spec:
        containers:
          ...
            volumeMounts:
              - mountPath: /init_data.json
                name: casdoor-init-data-volume
                subPath: init_data.json
        volumes:
          - configMap:
              name: casdoor-init-data
              name: casdoor-init-data-volume
```

## 文件内容

Casto版本库的根目录中已经有一个名为 init\_data.json.template 的模板文件。 您可以参考此文件来自定义您的数据初始化文件。

以下是每个对象对应的Go结构体和文档链接：

对象	Go 结构体	文档
组织机构	<a href="#">stuct</a>	<a href="#">doc</a>
应用	<a href="#">stuct</a>	<a href="#">doc</a>
用户	<a href="#">stuct</a>	<a href="#">doc</a>
提供商	<a href="#">stuct</a>	<a href="#">doc</a>
证书	<a href="#">stuct</a>	
Idaps	<a href="#">stuct</a>	<a href="#">doc</a>

如果您对填写此模板仍然感到困惑， 您可以调用 restful api或使用浏览器的调试模式来查看这些对象的 `GetXXX` 的响应。 这些响应和 `init_data.json` 的内容是相同的。

# 将静态文件托管到CDN

前端静态资源 (.js, .css 文件) 在 `web/build/static/` 目录下。如果您想要将它部署在公共云服务CDN中 Casdoor 提供了一个脚本，让您可以轻松地部署前端静态文件 请按照以下步骤操作：

## ① 备注

我们假定您已经构建过了Casdoor的前端代码。如果还没有，请参照：[文档](#)。

## 准备工作

首先，您需要在 Casdoor UI 中创建一个有效的 [存储提供商](#)。您可以参考 [示例](#)。

### ⚠ 注意事项

当您填写 `域` 字段时，请以 '/' 结束

Domain [?](#) :

<https://cdn.casbin.com/casdoor/>

## 使用说明

脚本放在文件[deployment/deploy\\_test.go](#)中。

你需要修改 `deploy_test.go` 中传入 `GetProvider()` 方法的参数 `id` 的值。The format of provider `id` is `<owner>/<name>`

```
func TestDeployStaticFiles(t *testing.T) {
    provider := object.GetProvider("admin/
provider_storage_aliyun_oss")
    deployStaticFiles(provider)
}
```

然后使用以下命令来运行脚本：

```
cd deployment
go test
```

如果执行成功，您将看到：

```
PASS
ok      github.com/casdoor/casdoor/deployment  2.951s
```

## 工作原理

脚本的功能：

- 它将会上传文件夹： `css/` and `js/` 中所有文件到指定的存储提供商的 CDN 服务上。
- 替换 `web/build/index.html` 中所有 `.css` 和 `.js` 的URL为托管CDN的URL。

您仍然需要保留 `index.html`。静态文件上传到CDN后，用户通过 Casdoor Go 后端请求 `index.html` 这些托管在CDN中的静态文件将通过 `index.html` 的URL被请求加载。

# 将静态文件托管在局域网内

如果您在 **内网上**部署了Casdoor，您可能无法直接通过互联网访问静态资源。您需要将静态资源部署在能够访问到它们的地方，然后在Casdoor中3处地方的配置。

## 部署静态资源

Casdoor的所有静态资源，包括图像、标志、css等，都存储在[casbin/static repository](#)仓库中。

克隆仓库，在网页服务器上部署静态资源。请确保您可以访问该资源。

## 配置Casdoor

您可以简单地修改配置文件，将静态资源地址设置为您部署的地址。[Go to conf/app.conf](#), set `staticBaseUrl` to your deployed address.

```
staticBaseUrl = "https://cdn.casbin.org"
```

# 数据库迁移

当数据库升级时，数据很可能崩溃，我们需要删除旧字段。幸运的是，Casdoor 使用的 [xorm](#) 能帮助我们处理许多有关数据库迁移的问题。但我们仍需自行处理一些模式和数据迁移，例如 **字段名称发生更改时**。

## ① 备注

有关Schema 操作的信息，您可以参阅[xorm docs](#)

## 工作原理

如上所述，当字段名称发生更改时，xorm将无法进行任何操作，但是它提供了一个 [迁移](#) 包来帮助我们解决这个问题。

你可以通过以下代码来处理字段重命名相关问题：

```
migrations := []*migrate.Migration{
{
    ID: "CasbinRule--fill ptype field with p",
    Migrate: func(tx *xorm.Engine) error {
        _, err :=
        tx.Cols("ptype").Update(&xormadapter.CasbinRule{
            Ptype: "p",
        })
        return err
    },
    Rollback: func(tx *xorm.Engine) error {
        return tx.DropTable(&xormadapter.CasbinRule{})
    },
}
```

我们想要实现的目标是：将 `p_type` 重命名为 `ptype` 但既然orm 不支持字段重命名我们只能使用更复杂的方式：将 `p_type` 的值分配给 `ptype` 然后删除 `p_type` 字段。

`ID` 字段特指我们进行的迁移。 `m.Migrate()` 运行后，`ID` 的值将被添加到数据库的迁移表。

当项目再次启动时 数据库将检查表中现有的 `ID` 字段，不会执行相同 `ID` 的操作。



> 如何连接到Casdoor

# 如何连接到Casdoor

## 概览

将您的应用连接到Casdoor

## 标准OIDC 客户端

使用 OIDC 发现迁移到Casdoor

## Casdoor SDKs

Using Casdoor SDKs instead of standard OIDC protocol

## 如何启用单点登录

启用单点登录



## Vue SDK

Casdoor Vue SDK



## 桌面 SDK

4 个项目



## Casdoor插件

在 Spring Boot, WordPress, Odoo 等其他框架中使用 Casdoor 插件或中间件。



## OAuth 2.0

使用AccessToken验证客户端



## CAS

使用 Casdoor 作为 CAS 服务器



## SAML

4 个项目



## WebAuthn

在 Casdoor 中使用 webauthn

# 概览

在本节中，我们将显示如何将您的应用程序连接到Casdoor。

作为服务提供商(SP)， Casdoor 支持两项认证协议：

- OAuth 2.0 (OIDC)
- SAML

作为身份提供商 (Idp)， Casdoor 支持四项认证协议：

- OAuth 2.0
- OIDC
- SAML
- CAS 1.0, 2.0, 3.0

## OAuth 2.0 (OIDC)

什么是 OAuth 2.0?

OAuth 2 是一个授权框架，允许应用程序例如Facebook, GitHub, 和Casdoor 访问HTTP服务上的用户帐户。它的工作原理是将用户身份验证委托给主机服务，并授权第三方应用程序访问该用户帐户。 OAuth 2 为网页和桌面应用程序以及移动设备提供了 授权流程。

Casdoor的授权程序基于OAuth 2.0协议。 我们建议使用 OAuth 2.0 协议的原因在于：

- 1. 该协议简单易行，能解决多种问题。
- 2. 成熟度高且社区支持广泛

因此，您的应用程序将通过 OAuth 2.0 (OIDC) 与 Casdoor 连接。具体而言，有三种方式连接Casdoor：

## 标准 OIDC 客户端

**标准OIDC 客户端:** 标准OIDC 客户端在各类编程语言或框架都广泛应用。

什么是OIDC?

OpenID Connect (OIDC) 是一个在OAuth 2.0 框架顶端运行的开放身份验证协议。针对消费者，OIDC允许个人通过单点登录(SSO)访问使用OpenID提供商(OPs)的依赖方站点，如电子邮件提供商或社交网络平台，以验证其身份。它向应用程序或服务提供用户信息、认证背景，并允许访问用户个人资料。

Casdoor 完全执行了OIDC协议。如果您的应用程序已经运行了另一个 OAuth 2，那么(OIDC) 身份提供商一般会通过标准的 OIDC 客户端库提供服务，如果您想迁移到 Casdoor，可使用OIDC discovery帮您轻松切换到Casdoor。

## Casdoor SDKs

**Casdoor SDKs:** For most programming languages, Casdoor will provide easy-to-use SDK library on top of OIDC, with supporting extended functionality which are only available in Casdoor.

与标准的 OIDC 协议相比，Casdoor 在 SDK 中提供了更多的功能，如用户管理、资源上传等。通过 Casdoor SDK 连接到Casdoor 的成本比使用 OIDC 标准客户端库要高，但前者能提供最佳灵活性和最强API。

## Casdoor插件

**Casdoor 插件:** 如果您的应用建立在一个流行的平台上(如Spring Boot, WordPress等), 并且Casdoor(或第三方) 已经为它提供了一个插件或中间件, 那么就可以直接使用。 使用插件比手动使用 Casdoor SDK 更容易, 因为前者是专门为平台制作的。

**插件:**

- [Jenkins 插件](#)
- [APISIX 插件](#)

**中间件:**

- [Spring Boot插件](#)
- [Django 插件](#)

## SAML

什么是SAML?

安全鉴别标记语言(SAML)是一种开放标准, 允许身份提供者(IDP)将授权证书传给服务提供者。 该术语意味着您可以使用一组凭据登录多个不同的网站。 管理一个用户的单次登录要比管理分别登录到电子邮件、客户关系管理(CRM) 软件、Active Directory等简单得多。

SAML交易使用可扩展标记语言(XML) 在标识提供者和服务提供者之间进行标准化通信。 SAML 是用户身份验证和使用服务授权之间的链接。

Casdoor可以使用 SAML IdP。 目前Casdoor支持SAML2.0 的主要功能。 More details



see [SAML](#).

示例:

[Casdoor 作为一个 SAML IdP in Keycloak](#)

建议:

1. 该协议十分 **强大** 且适用于多种情景，可以说它是最全面的 SSO 协议之一。
2. 该协议 **涵盖范围过大**，有很多可选参数，因此，在实操中很难100%涵盖所有应用场景。
3. 如果应用程序是 **新开发的**，那么**不是很推荐 SAML**，因为其技术过于复杂。

## CAS

什么是CAS?

The Central Authentication Service (CAS) is a single sign-on protocol for the web. Its purpose is to permit a user to access multiple applications while providing their credentials (such as user ID and password) only once. It also allows web applications to authenticate users without gaining access to a user's security credentials, such as a password.

Casdoor已实现 CAS 1.0, 2.0, 3.0 功能。 详情见 [CAS](#)

建议:

1. 该协议本身较为轻量，易于执行，但它适用的场景较为单一。
2. CAS 客户端和CAS服务器之间的相互信任是通过在没有任何加密或签名机制的情况下使用接口建立的，目的是确保进一步的安全。
3. CAS协议与其他协议相比没有优势。

# 集成表

一些应用程序已有连接到Casdoor的示例。 您可以按照文档指示操作， 快速连接到 Casdoor。 您可以在 [集成表](#) 中看到所有应用程序。

# 标准OIDC 客户端

## OIDC discovery

Casdoor 完全实现了OIDC协议。如果您的应用程序已经运行了另一个 OAuth 2，那么(OIDC) 身份提供商一般会通过标准的 OIDC 客户端库提供服务，如果您想要迁移到 Casdoor，使用 OIDC discovery 会帮助您非常容易地切换到Casdoor。Cassdoor's OIDC discovery URL 是：

```
<your-casdoor-backend-host>/.well-known/openid-configuration
```

E.g., the OIDC discovery URL for the demo site is: <https://door.casdoor.com/.well-known/openid-configuration> , with the following content:

```
{
  "issuer": "https://door.casdoor.com",
  "authorization_endpoint": "https://door.casdoor.com/login/oauth/authorize",
  "token_endpoint": "https://door.casdoor.com/api/login/oauth/access_token",
  "userinfo_endpoint": "https://door.casdoor.com/api/userinfo",
  "jwks_uri": "https://door.casdoor.com/.well-known/jwks",
  "introspection_endpoint": "https://door.casdoor.com/api/login/oauth/introspect",
  "response_types_supported": [
    "code",
    "token",
    "id_token",
    "code token",
    "code id_token",
  ]}
```

# OIDC 客户端库列表

这里我们列出了一些OIDC 客户端库，如Go 和 Java 等语言：

OIDC 客户端库	语言	链接
go-oidc	Go	<a href="https://github.com/coreos/go-oidc">https://github.com/coreos/go-oidc</a>
pac4j-oidc	Java	<a href="https://www.pac4j.org/docs/clients/openid-connect.html">https://www.pac4j.org/docs/clients/openid-connect.html</a>

上表远远没有完成。 OIDC 客户端库的完整列表请查看更多详情：

1. <https://oauth.net/code/>
2. <https://openid.net/certified-open-id-developer-tools/>

## OIDC UserInfo fields

The following table shows how OIDC UserInfo fields (via `/api/userinfo` API) are mapped from properties of Casdoor's User table:

Casdoor User Field	OIDC UserInfo Field
Id	sub
originBackend	iss

Casdoor User Field	OIDC UserInfo Field
Aud	aud
Name	preferred_username
DisplayName	name
Email	email
Avatar	picture
Location	address
Phone	phone

See UserInfo's definition here: <https://github.com/casdoor/casdoor/blob/95ab2472ce84c479be43d6fc4db6533fc738b259/object/user.go#L175-L185>



# Casdoor SDKs

## 简介

与标准的 OIDC 协议相比，Casdoor 在 SDK 中提供了更多的功能，如用户管理、资源上传等。通过 Casdoor SDK 连接到 Casdoor 的成本比使用 OIDC 标准客户端库更低，并将提供灵活性最佳和最强大的 API。

Casdoor SDK 可分为两类：

1. 前端 SDK：用于网站的 Javascript SDK 和 Vue SDK，用于应用的 Android 或 iOS SDK。Casdoor 支持为网站和移动应用程序提供身份验证。
2. 后端 SDK：Go, Java, Node.js, Python, PHP 等后端语言的 SDK。



如果您的网站是采用后端分离的方式开发，您可以使用 Javascript SDK：[casdoor-js-sdk](#) 或 Vue SDK：[casdoor-vue-sdk](#) 将 Casdoor 整合到前端。如果您的网页应用程序是由 JSP 或 PHP 开发的传统网站，那您就只能使用后端 SDK。示例：[casdoor-Python-vue-sdk示例](#)

Desktop SDK	描述	SDK 代码库	示例
Android SDK	For Android apps	<a href="#">casdoor-android-sdk</a>	<a href="#">casdoor-android-example</a>
iOS SDK	For iOS apps	<a href="#">casdoor-ios-sdk</a>	<a href="#">casdoor-ios-example</a>
Electron SDK	For Electron apps	<a href="#">casdoor-js-sdk</a>	<a href="#">casdoor-electron-example</a>
.NET Desktop SDK	For .NET desktop apps	<a href="#">casdoor-dotnet-sdk</a>	WPF: <a href="#">casdoor-dotnet-desktop-example</a> WinForms: <a href="#">casdoor-dotnet-winform-example</a> Avalonia UI: <a href="#">casdoor-dotnet-avalonia-example</a>
Unity Games SDK	For Unity 2D/3D PC/Mobile games	<a href="#">casdoor-dotnet-sdk</a>	<a href="#">casdoor-unity-example</a>
Flutter SDK	For Flutter apps	<a href="#">casdoor-flutter-sdk</a>	<a href="#">casdoor-flutter-example</a>
uni-app SDK	For uni-app apps	<a href="#">casdoor-uniapp-sdk</a>	<a href="#">casdoor-uniapp-example</a>

Frontend SDK	描述	SDK code	示例
Javascript SDK	For traditional non-SPA websites	<a href="#">casdoor-js-sdk</a>	Small example: <a href="#">casdoor-raw-js-example</a> Large examples: <a href="#">Casnode</a> , <a href="#">Casbin-OA</a> , <a href="#">Confita</a>
React SDK	For SPA React websites	<a href="#">casdoor-react-sdk</a>	Nodejs backend: <a href="#">casdoor-nodejs-react-example</a> Java backend: <a href="#">casdoor-spring-security-react-example</a>
Vue SDK	For SPA Vue websites	<a href="#">casdoor-vue-sdk</a>	<a href="#">casdoor-python-vue-sdk-example</a>
Angular SDK	For SPA Angular 1.x, 2.x websites	<a href="#">casdoor-angular-sdk</a>	<a href="#">casdoor-nodejs-angular-example</a>
Flutter SDK	For Flutter websites	<a href="#">casdoor-flutter-sdk</a>	<a href="#">casdoor-flutter-example</a>

接下来，根据您后端的语言，可以选择使用下面的后端 SDK 之一：

Backend SDK	Description	Sdk code	Example code
Go SDK	For Go backends	<a href="#">casdoor-go-sdk</a>	<a href="#">Casnode</a> , <a href="#">Casbin-OA</a> , <a href="#">Confita</a>
Java SDK	For Java backends	<a href="#">casdoor-java-sdk</a>	<a href="#">casdoor-spring-boot-starter</a> , <a href="#">casdoor-spring-boot-example</a> , <a href="#">casdoor-spring-security-react-example</a>
Node.js SDK	For Node.js backends	<a href="#">casdoor-nodejs-sdk</a>	<a href="#">casdoor-nodejs-react-example</a>
Python SDK	For Python backends	<a href="#">casdoor-python-sdk</a>	Flask: <a href="#">casdoor-python-vue-sdk-example</a> FastAPI: <a href="#">casdoor-fastapi-js-sdk-example</a>
PHP SDK	For PHP backends	<a href="#">casdoor-php-sdk</a>	<a href="#">wordpress-casdoor-plugin</a>
.NET SDK	For ASP.NET backends	<a href="#">casdoor-dotnet-sdk</a>	<a href="#">casdoor-dotnet-sdk-example</a>
Rust SDK	For Rust backends	<a href="#">casdoor-rust-sdk</a>	<a href="#">casdoor-rust-example</a>
C/C++ SDK	For C/C++ backends	<a href="#">casdoor-cpp-sdk</a>	<a href="#">casdoor-cpp-qt-example</a>
Dart SDK	For Dart backends	<a href="#">casdoor-dart-sdk</a>	
Ruby SDK	For Ruby backends	<a href="#">casdoor-ruby-sdk</a>	

For a full list of the official Casdoor SDKs, please see: <https://github.com/orgs/casdoor/repositories?q=sdk&type=all&language=&sort=>

## 如何使用 Casdoor SDK ?

### 1. 后端 SDK 配置

当您的应用程序启动时，您需要调用 `InitConfig()` 函数来初始化Casdoor SDK 配置。Take casdoor-go-sdk as example: <https://github.com/casbin/casnode/blob/6d4c55f5c9a3c4bd8c85f2493abad3553b9c7ac0/controllers/account.go#L51-L64>

```
var CasdoorEndpoint = "https://door.casdoor.com"
var ClientId = "541738959670d221d59d"
var ClientSecret = "66863369a64a5863827cf949bab70ed560ba24bf"
var CasdoorOrganization = "casbin"
var CasdoorApplication = "app-casnode"

//go:embed token_jwt_key.pem
var JwtPublicKey string

func init() {
    auth.InitConfig(CasdoorEndpoint, ClientId, ClientSecret, JwtPublicKey, CasdoorOrganization, CasdoorApplication)
}
```

`InitConfig()` 的所有参数解释为：

Parameter	Must	Description
endpoint	Yes	Casdoor Server URL, like <code>https://door.casdoor.com</code> or <code>http://localhost:8000</code>
clientId	Yes	Client ID for the Casdoor application
clientSecret	Yes	Client secret for the Casdoor application
jwtPublicKey	Yes	The public key for the Casdoor application's cert
organizationName	Yes	The name for the Casdoor organization
applicationName	No	The name for the Casdoor application

### 提示

`jwtPublicKey` 可以在 `Certs` 页面中进行管理。

Certs <a href="#">Add</a>								
Name	Created time	Display name	Scope	Type	Crypto algorithm	Bit size	Expire in years	
cert_rjeegc	2022-02-16 11:04:10	New Cert - rjeegc		JWT	x509	RSA	4096	20
cert-built-in	2022-02-15 12:31:46	Built-in Cert		JWT	x509	RSA	4096	20

2 in total < 1 > 10 / page ✓

您可以在证书编辑页面中找到公钥，复制或下载它以供 sdk 使用。

Public key [②](#)

[Copy public key](#) [Download public key](#)

```
-----BEGIN CERTIFICATE-----
MIIEtTCCAuGgAwIBAgDAlEAMA0GCSwGQGSIb3DQEBCwUAMDYxhTAbBgNVBAoTFENh
c2Rvb3IgT3JyNWPsem0aW9uMRUwEwDVQDExwDYXNkb29yEnIcnQvWhcNMjEx
MDE1MDgxMTUyWhcNNDExMDExMDExMDgxMTUyWjA2MR0wGwDVQQExRDYXNkb29yIE9y
Z2FuaxphdGlbjEVMBMGa1UEAxMMQ2FzG9vcibDZXJ0MIIcIjAnBgkqhkiG9w0B
AQEEACAgAaMIICCGkCAgEA5npbSE1ym0f1RfSDSSE8IR7y+lw+RJj74e5ejrq4b8zMY
k7HeHcyZrhnNewEVnhXu1P0mbeQ5ppp/QGo8vgEmjAEfNTmzkl1NjOQCjCyWra
sO/fMn1C0j13x6mV1khZj5prksMhYY1vaxTEP3+VB8Hjg3MhFvrb07
CjCwUrsoAf/Mn1C0j13x6mV1khZj5prksMhYY1vaxTEP3+VB8Hjg3MhFvrb07
uvFMcLe5W8+0rKzCkCTR8+9V8janeBz//zQePFVh79rbZate/hUPkOgo9P1g
OrwloC1A3saarHTPA0m/LDR0rhq2FybdySpw/WAQvhNaPEf7mTsRSBb/vtJNCUBD
PTSLVjC4WILf6Nkh027Kmvp1Sj+btvcsqRAfGtdsB9h62Kpjjs1Yn/GAuo
l3qf1zokb1URyxQjKlwvQsEftUkSev5zuSiDRUoLByGTlbxoQjLAFNFW3g/
pz5Dgd/60d6t1mtvbZn45mjdyfXCDb1Kn7N+xtojfa!kweP2REv+RMcOfx4Gu
hRsN.smkmUDeyPz7B0w4ZvOfj/1VLrOfjtPbLif0bfh/AeZMhpIKoxvfz4
IDp5262hod45RSvbz7B0w4ZvOfj/1VLrOfjtPbLif0bfh/AeZMhpIKoxvfz4
yF+hg268wdfDVR9x9y/RbsAT73230s/njyjEgnURohnRgCpjik/Mz2kt84kb0
wn8CAwEaaMQMA4DADYVRDTAQh/BaWaDABNgkjhG9w0BAQsFAAACAgEAn2f
DKLX+F1vKR0/Sgl+Plr8P5NuKuQkmwH97bCS2g1S1phDyNgk4/Lsdzu4Aw6ve
C06lvdWSts8UPuPdjm72uMPSNjwLxG3QsrinMURNwfLTRen/Hele02gun91IM
8haawdSjjH2RgmFoDeEr2vNRfhroRknCo1ddJkUs1N0/rh21W4jt4rxzCvI
2nR42fjyap3D/g2jXMHNR0wZmNjggsF7XVENCsuFO1jywLaguxCg54l7XVLG
omKNNNcc8h1FCekj/nmbGMhodnFWKDTsJkImCPNH6oixzqmly-Hqc+mWYy7maAG
Jtevs3ggM2F8P9Qz9rHpcUc63R2YWWYDv/xyPisukHoP2gHt979X4ndf0WnP0bLqL
2D1zaBnnjjeGolv7XNVKctJfXXy8w5ZTZ5h9cl4e+6bnyWqJlhw+AtiJFvE
XzG70B4IALX6au1kLepV901GERzYRz5P9Nuna7koOSAVMp9w0DDTkt+LbXnZE
HHnWky8xhQKZ9s7RyBPLG/c66vn5u150gj/8dtRZ/veyFGo2yZs+hKVUS
nCjHBCayFrnn1hdvdwEdi33jDBnCiozJzr/3VmlSalDosHAgMWlxuWP+h
8XKxmzkuHbTMQY1ZDgspSSak+54Q9wb8RRAYO=
-----END CERTIFICATE-----
```

Private key [②](#)

[Copy private key](#) [Download private key](#)

```
-----BEGIN PRIVATE KEY-----
MIUKQBAAKCgAsInpbE1ym0f1RfSDSSE8IR7y+lw+RJj74e5ejrq4b8zMY
k7HeHcyZrhnNewEVnhXu1P0mbeQ5ppp/QGo8vgEmjAEfNTmzkl1NjOQCjCyWra
sO/fMn1C0j13x6mV1khZj5prksMhYY1vaxTEP3+VB8Hjg3MhFvrb07
CbiUrYxQjKlwvQsEftUkSev5zuSiDRUoLByGTlbxoQjLAFNFW3g/pz5Djd9
60d6H1mvbZn45mjdyfXCDb1Kn7N+xtojfa!kweP2REv+RMcOfx4GuhrSn.lsmk
mUDey29aLs9j1YEQfMj2Eq+RVtUx+wB4y8K/D1bcv+fnfG5r8pw!DpS262b
oq4S5v62Z7RbW4ZvOfj/1VLrOfjtPbLif0bfh/AeZMhpIKoxvfz4
8wfD09r9yCjRbsAT73230s/njyjEgnURohnRgCpjik/Mz2kt84kb0vn8CawEA
AQKCAgAhTzrVJINVRQyDfZf1P1Yd+IMlMjmpQH9woRiB6640Upxelplpzb1CpOYu
npF7x9jUtzTzcdU6FLDqL82ktx607TnkFvymhNy4wkn75gTgwMrQzTrbwxb
Aftxp4ZVM8l/15W7zMVbHuabHAAu50s0RbvBN+v3nTa7/JvdMs5wX6ah3uFLQW
aYEfQVn3Ww/IpZaBwfD94HKAav/TgSKU4EcplAcI6CC1fnnnSb6/pBBG
khaTdAkoCgWVx3Em1dkR2uuav4Bgs7dzJkoAv/JBwVt-/f5JrFpHmy5AKYLa
buMf6rdHhEixRbmIdoAwTfFms08bzU26caGhufo4YMI+B4b6EqsNmNsNR9
Msau8qkSlpr6oMrj1Q7y1q3rShf8SzBa3xk0hBy8z91jkJ+Dtbg2zakQWzDG
JLEttbGgdeYUMS2yc/CfUYVN/YPCdri769kw!mOr2K156wpbWvYwR9jYgnQzb3jd
4AOGrsg3AavDwX0v1o71c8e334wusvxNCjRuZBY/DK0/W7jyjxdvKevHxGrh
1Gc+FkkbeFnf9qDzdlkx66N80nyZuLyymRiavn9bVcarb5xHS5ICEHOHwogH
5GdesqMugTSOev1DrUnfc1CWWMvPeEuShW9jbl3BBh4wfLsQKCAoEay4*x+c+F8
icbaKfssrnPMYJiWhee39p0vHDtm/3sx0ysefUxLsjagQz7n0ijf7Xf10h+
vcG66A0jwA6+jQdxEf8er04t56uMac3fWPVAJ000/HClvrgA9gC2H0c940/Yn
66gWYqz2AxU156RA3P/TetgsuVCFI62PxPyfbqzB71Y9h9sC22zG3K8+PvZ
dp+DFjyHb6lRuWvDxxKuQjUx7Ff7zFqvKhm52qvKXfQfWpx4H71kQAU7Q
cJ8lygquFlouIopOD6DB10P/1Btg59a+jsnxccpCj9yXchGsj1d7s0mQd
har/kYNa4dhNy2QKCAQEa3gekrROPdglocoamednlwlplrryK5MrhEAg1Be.idp
98HhNc08wra5xLMs62C7R0KOkiKsSQLpAgIT22W59lq/nPQ7PN5Pdn2Rz0lrmHc
```

[Save](#)

[Save & Exit](#)

之后，您可以在应用编辑页面选择证书。

## 2. 前端配置

首先，通过 NPM 或 Yarn 安装 `casdoor-js-sdk`

```
npm install casdoor-js-sdk
```

或者：

```
yarn add casdoor-js-sdk
```

然后定义以下实用功能(在全局JS文件中更好，比如 `Setting.js`)：

```
import Sdk from "casdoor-js-sdk";

export function initCasdoorSdk(config) {
  CasdoorSdk = new Sdk(config);
}

export function getSignupUrl() {
  return CasdoorSdk.getSignupUrl();
}

export function getSigninUrl() {
  return CasdoorSdk.getSigninUrl();
}

export function getUserProfileUrl(userName, account) {
  return CasdoorSdk.getUserProfileUrl(userName, account);
}

export function getMyProfileUrl(account) {
  return CasdoorSdk.getMyProfileUrl(account);
}

export function getMyResourcesUrl(account) {
  return CasdoorSdk.getMyProfileUrl(account).replace("/account?", "/resources?");
}
```

在您前端代码的入口文件(如 `index.js` 或 `app.js` 在React中), 您需要通过调用 `InitConfig()` 函数来初始化 `casdoor-js-sdk` 前4个参数应该使用与 Casdoor 后端SDK 相同的值。最后一个参数 `重定向路径` 是从Casdoor的登录页面返回的重定向URL的相对路径。

```
const config = {
  serverUrl: "https://door.casdoor.com",
  clientId: "014ae4bd048734ca2dea",
  organizationName: "casbin",
  appName: "app-casnode",
  redirectPath: "/callback",
};

xxx.initCasdoorSdk(config);
```

(可选) 因为我们正在使用React作为示例, 我们的 `/callback` 路径正在撞击React路由。我们使用以下React组件接收 `/回调` 调用并发送到后端。如果您直接重定向到后端(如JSP 或 PHP), 您可以忽略此步骤。

```
import React from "react";
import {Button, Result, Spin} from "antd";
import {withRouter} from "react-router-dom";
import * as Setting from "./Setting";

class AuthCallback extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      classes: props,
      msg: null,
    };
  }

  componentWillMount() {
    this.login();
  }

  login() {
    Setting.signin().then((res) => {
      if (res.status === "ok") {
        Setting.showMessage("success", `Logged in successfully`);
        Setting.goToLink("/");
      } else {
        this.setState({
          msg: res.msg,
        });
      }
    });
  }

  render() {
    return (
      <div style={{textAlign: "center"}}>
        {this.state.msg === null ? (
          <Spin
            size="large"
            tip="Signing in..."
            style={{paddingTop: "10%"}}
          />
        ) : (
          <div style={{display: "inline"}}>
            <Result
              status="error"
              title="Login Error"
              subTitle={this.state.msg}
              extra={[
                <Button type="primary" key="details">
                  Details
                </Button>
              ]}
            />
        )}
      </div>
    );
  }
}
```

### 3. 获取登录 URL

接下来，您可以显示“注册”和“登录”按钮或链接到您的用户。可以在前端或后端检索URL。详细信息见：[/docs/basic/core-concepts#login-urls](#)

### 4. 获取并验证token

步骤如下：

1. 用户点击登录URL并重定向到Casdoor的登录页面，如 `https://door.casbin.com/login/oauth/authorize?client_id=014ae4bd048734ca2dea&response_type=code&redirect_uri=https%3A%2F%2Fforum.casbin.com%2Fcallback&scope=read&state=app-casnode`
2. 用户输入用户名和密码，并点击登录（或者选择第三方登录，例如通过GitHub进行登录）
3. 该用户被重定向到您的应用，使用Casto发行的授权码(例如： `https://forum.casbin.com?code=xxx&state=yyy`)，您的应用程序的后端需要将授权码与访问令牌交换，并验证访问令牌是否有效和由Casdoor签发。函数`GetOAuthToken()` 和`ParseJwtToken()` 由Casdoor后端SDK提供。

以下代码显示如何获取并验证访问令牌。For a real example of Casnode (a forum website written in Go), see: <https://github.com/casbin/casnode/blob/6d4c55f5c9a3c4bd8c85f2493abad3553b9c7ac0/controllers/account.go#L51-L64>

```
// 从重定向 URL 的 GET 参数中获取代码和状态
code := c.Input().Get("code")
state := c.Input().Get("state")

// 用代码和状态交换token
token, err := auth.GetOAuthToken(code, state)
if err != nil {
    panic(err)
}

// 验证访问令牌
claims, err := auth.ParseJwtToken(token.AccessToken)
if err != nil {
    panic(err)
}
```

如果`ParseJwtToken()`结束时没有错误，那么用户已成功登录到应用程序。返回的`claims`可以稍后用来识别用户。

### 4. 用token识别用户



信息

这一部分实际上是您的应用程序本身的业务逻辑，而不是OIDC、OAuth 或Casodo的一部分。我们只是提供正确做法，因为许多人不知道该怎么

在Casodo中，访问令牌通常与ID令牌相同。他们是一样的。因此，访问令牌包含登录用户的所有信息。

由`ParseJwtToken()`返回的变量`claims`被定义为：

```
Type Claims struct {
    User
    AccessToken string `json:"accessToken"`
    jwt.RegisteredClaims
}
```

1. `User`: User 对象，包含登录用户的所有信息，请参见定义：[/docs/basic/core-concepts#user](#)
2. `AccessToken`: token信息
3. `jwt.RegisteredClaim`: JWT需要一些其他值。

这时，应用程序通常有两种方法记住用户会话： `session` and `JWT`。

## Session

设置Session的方法因语言和框架而大不相同。例如，Casnode 使用 [Beego web 框架](#) 并通过调用设置会话：`c.SetSessionUser()`。

```
token, err := auth.GetOAuthToken(code, state)
if err != nil {
    panic(err)
}

claims, err := auth.ParseJwtToken(token.AccessToken)
if err != nil {
    panic(err)
}

claims.AccessToken = token.AccessToken
c.SessionUser(claims) // 设置会话
```

## JWT

从 Casdoor 返回的 `accessToken` 实际上是一个 JWT。因此，如果您的应用程序使用 JWT 来保持用户 `session`，只需直接为它使用访问令牌：

1. 将访问令牌发送到前端，在本地存储浏览器等地方保存。
2. 让浏览器为每一个请求发送访问令牌到后端。
3. 调用 `ParseJwtToken()` 或使用您自己的函数来验证 `token`，通过后端提供的已登录用户信息。

## 5. (可选) 与用户表的互动



信息

这一部分是由 [Castor Public API](#) 提供的，而不是OIDC 或 OAuth 的一部分。

Casdoor Backend SDK 提供了许多辅助功能，不仅限于：

- `GetUser(name string)`: 通过用户名获取用户。
- `GetUsers()`: 获取所有用户。
- `AddUser()`: 添加一个用户。
- `UpdateUser()`: 更新一个用户。
- `DeleteUser()`: 删除一个用户。
- `CheckUserPassword(auth.User)`: 检查用户的密码。

这些函数是通过对 [Castor Public API](#) 调用 RESTful API 实现的。如果 Casdoor Backend SDK 中没有提供功能，您可以自己调用 RESTful API。

# 如何启用单点登录

## 简介

您已连接了 Casdoor，并在组织中配置了多个应用程序。 You want users to sign in once to any app in the organization, and then be able to sign in when they go to another app, without any extra clicks.

我们提供这个单点登录，您只需要：

- 启用自动登录按钮。
- 填写首页的 URL。
- 在应用程序主页中添加 Silent Signin 函数。

### 备注

The basic sign in process provided by Casdoor allows users to log in to other applications in the organization by selecting the user who is currently logged in or using another account.

启用自动登录后，选择框将不显示，登录用户将直接登录。

## 配置

1. 填充字段 首页。 它可以是应用程序的主页或登录页面。

**Casdoor** Home Organizations Users Roles Permissions Models Adapters Providers Applications

Edit Application

Name [?](#) : app-casbin-oa

Display name [?](#) : Casbin OA

Logo [?](#) : URL [https://cdn.casbin.org/img/casbin\\_logo\\_1024x256.png](https://cdn.casbin.org/img/casbin_logo_1024x256.png)

Preview: 

Home [?](#) : <https://oa.casbin.com>

Description [?](#) : OA system for Casbin

2. 启用自动登录按钮。

Password ON [?](#) :

Enable signup [?](#) :

Signin session [?](#) :

Auto signin [?](#) :

Enable code signin [?](#) :

Enable WebAuthn signin [?](#) :

# 添加静默登录

事实上，我们通过在URL上传参数来实现自动登录。所以您的应用程序需要有一种方法来在跳转到URL后触发 登录。我们提供了 [casdoor-react-sdk](#) 以便您快速实现此功能。详细信息请参阅[use-in-react](#).

## ① 信息

工作原理

1. 在应用程序主页的 URL 中，我们将携带 `silentSignin` 参数。
2. 在您的主页中确定您是否需要通过参数 `静音签名` 来静音登录。如果 `silentSignin === 1`，函数返回 `SilentSignin` 组件，这将帮助您启动登录请求。既然您已启用自动登录，用户将自动登录无需点击。

# Add Popup Signin

`popup signin` will pop up a small window. After logging in to Casdoor in the child window, it will send authentication information to the main window and then close automatically. We implement it by carrying parameters on the URL.

## ① 信息

How to use

Use the method `popupSignin()` in sdk [casdoor-js-sdk](#) to quickly implement the feature. You can see a demo in [casdoor-nodejs-react-example](#).

## How it works

1. In the URL to the application home page, we will carry the `popup` parameter.
2. When `popup=1` in login params, Casdoor will send `code` and `state` as a message to main window and finish get `token` in main window by SDK.

# 使用 SSO

配置已完成，下面将展示如何使用自动登录。

### ① 信息

请确保您的应用程序可以重定向到用户的个人资料页面。The API [`getMyProfileUrl\(account, returnUrl\)`](#) is provided in our SDK for each language.

Open the profile page and go to the "Home" page (`/` URL path). You will see the application list provided by the organization. It's notable that only users in organizations other than `built-in` can see the application list in the "Home" page. All the global administrators (aka in the `built-in` organization) cannot see it.



Click on a tile in the application list, it will jump to the homepage URL of that application with GET parameter: `?silentSignin=1` and automatically log into the application if the application has integrated with Casdoor SSO (so it will recognize the `?silentSignin=1` parameter and perform silent login in the background).

# Vue SDK

Casdoor Vue SDK 是专为Vue2 和 Vue3 设计的，使用非常方便。

Vue SDK 基于 `casdoor-js-sdk`, 您也可以直接使用 `casdoor-javascript-sdk`，这将更易定制使用。

这个插件还在开发中！如果您有任何问题和建议，请联系我们 [issue](#)

我们将向您展示以下步骤。

如果在阅读README后您仍然不知道如何使用它, 您可以访问这个链接: [casdoor-python-vue-sdk-example](#)来获取更多信息。

The example' front-end is built with casdoor-vue-sdk, and the back-end is built with casdoor-python-sdk, you can see the source code in the example.

## 安装

```
# NPM  
npm i casdoor-vue-sdk  
  
# Yarn  
yarn add casdoor-vue-sdk
```

# Init SDK

初始化需要 5 个参数，它们都是字符串类型：

名称(按顺序排列)	必选项	描述信息
服务器Url	是	您的Casdoor服务器URL
客户端 Id:	是	您Casdoor应用程序的客户端 ID
应用程序名称	是	您的Casdoor应用程序的名称
组织名称	是	与您的Casdoor应用程序相关联的Casdoor组织名称
重定向路径	否	您的 Casdoor 应用程序的重定向URL 路径将是 <code>/callback</code>

install:

对于Vue3:

```
// in main.js
import Casdoor from 'casdoor-vue-sdk'
const config = {
  serverUrl: "http://localhost:8000",
  clientId: "4262bea2b293539fe45e",
  organizationName: "casbin",
  appName: "app-casnnode",
  redirectPath: "/callback",
```

对于Vue2:

```
// in main.js
import Casdoor from 'casdoor-vue-sdk'
import VueCompositionAPI from '@vue/composition-api'
const config = {
  serverUrl: "http://localhost:8000",
  clientId: "4262bea2b293539fe45e",
  organizationName: "casbin",
  appName: "app-casnnode",
  redirectPath: "/callback",
};
Vue.use(VueCompositionAPI)
Vue.use(Casdoor,config)
new Vue({
  render: h => h(App),
}).$mount('#app')
```

## 示例：

```
// in app.vue
<script>
export default {
  name: 'App',
  methods: {
    login() {
      window.location.href = this.getSigninUrl();
    },
    signup() {
      window.location.href = this.getSignupUrl();
    }
  }
}</script>
```

## 自动修复

如果 没有触发 `postinstall` hook，或者您更新了 Vue 版本，尝试运行以下命令解决重定向问题。

```
npx vue-demi-fix
```

关于Vue 版本的更多信息：请参见 [vue-demi 文档](#)

# 桌面 SDK

## Electron 应用

一个 Electron 桌面应用程序用于Casdoor的示例

## Dotnet桌面应用程序

一个 Dotnet 桌面应用程序用于Casdoor的示例

## Mobile SDKs .NET MAUI App

An .NET MAUI App example for Casdoor

## Qt 桌面应用程序

一个 Qt 桌面应用程序用于Casdoor的示例

# Electron 应用

一个为 Casdoor 的 Electron 桌面应用程序示例。

## 如何运行示例

### 初始化

您需要初始化6个参数，它们都是字符串类型：

名称	描述	路径
serverUrl	您的Casdoor服务器URL	src/App.js
clientId	您Casdoor应用程序的客户端 ID	src/App.js
appName	您的Casdoor应用程序的名称	src/App.js
redirectPath	您的 Casdoor 应用程序的重定向URL 路径将是 /callback 如果没有提供	src/App.js
clientSecret	您的Casdoor应用程序的客户端密钥	src/App.js
casdoorServiceDomain	您的Casdoor服务器URL	public/electron.js

如果您没有设置这些参数，此项目将使用 [Casdoor 在线演示](#) 作为默认的Casdoor 服务器，并使用 [Casnode](#) 作为默认的 Casdoor 应用程序。

### 可用命令

在项目目录中，您可以运行：

`npm run dev` 或者 `yarn dev`

构建electron应用并运行此应用。

`npm run make` 或者 `yarn make`

打包和分发您的应用程序。它将创建 `out` 文件夹，您的软件包将被定位：

```
// macOS下的out/示例
├── out/make/zip/darwin/x64/casdoor-electron-example-darwin-x64-1.0.0.zip
├── ...
└── out/casdoor-electron-example-darwin-x64/casdoor-electron-example.app/Contents/MacOS/casdoor-electron-example
```

### 预览

在您运行此electron应用程序后，将在您的桌面上显示一个新的窗口。



如果您点击 [使用 Casdoor 登陆] 按钮，您的默认浏览器将被自动打开并显示登录页面。



Made with ❤ by Casdoor

登录成功后，将会打开您的electron应用程序，您的用户名将会显示在您的应用程序中。



您可以通过下面的Gif图像预览整个过程。



## 如何整合?

### 设置自定义协议

首先，您需要设置自定义协议名为 `casdoor`。

```
const protocol = "casdoor";

if (process.defaultApp) {
  if (process.argv.length >= 2) {
    app.setAsDefaultProtocolClient(protocol, process.execPath, [
      path.resolve(process.argv[1]),
    ]);
  }
} else {
  app.setAsDefaultProtocolClient(protocol);
}
```

这将帮助浏览器打开您的electron应用程序并将登录信息发送到electron应用程序中。

### 通过浏览器打开登录网址

```
const serverUrl = "https://door.casdoor.com";
const appName = "app-casnode";
const redirectPath = "/callback";
const clientId = "014ae4bd048734ca2dea";
const clientSecret = "f26a4115725867b7bb7b668c81e1f8f7fae1544d";

const redirectUrl = "casdoor://localhost:3000" + redirectPath;
```

您可以更改前5个参数。

## 监听开启的应用程序事件

在您在浏览器成功登录后，浏览器将打开您的electron应用程序。您需要监听打开的应用程序事件。

```
const gotTheLock = app.requestSingleInstanceLock();
const ProtocolRegExp = new RegExp(`^${protocol}://`);

if (!gotTheLock) {
    app.quit();
} else {
    app.on("second-instance", (event, commandLine, workingDirectory) => {
        if (mainWindow) {
            if (mainWindow.isMinimized()) mainWindow.restore();
            mainWindow.focus();
            commandLine.forEach((str) => {
                if (ProtocolRegExp.test(str)) {
                    const params = url.parse(str, true).query;
                    if (params && params.code) {
                        store.set("casdoor_code", params.code);
                        mainWindow.webContents.send("receiveCode", params.code);
                    }
                }
            });
        }
    });
    app.whenReady().then(createWindow);

    app.on("open-url", (event, openUrl) => {
        const isProtocol = ProtocolRegExp.test(openUrl);
        if (isProtocol) {
            const params = url.parse(openUrl, true).query;
            if (params && params.code) {
                store.set("casdoor_code", params.code);
                mainWindow.webContents.send("receiveCode", params.code);
            }
        }
    });
}
}
```

您可以从broswer获取代码，即`casdoor_code`或`params.code`。

## 解析代码并获取用户信息

```
async function getUserInfo(clientId, clientSecret, code) {
    const { data } = await axios({
        method: "post",
        url: authCodeUrl,
        headers: {
            "content-type": "application/json",
        },
        data: JSON.stringify({
            grant_type: "authorization_code",
            client_id: clientId,
            client_secret: clientSecret,
            code: code,
        }),
    });
    const resp = await axios({
        method: "get",
        url: `${userInfoUrl}?accessToken=${data.access_token}`,
    });
    return resp.data;
}
```

最后，您可以解析代码，并按照[OAuth文档](#)页面获取用户信息。

# Dotnet桌面应用程序

一个为 Casdoor 的 [Dotnet 桌面应用程序示例](#)。

## How to run example

### Prerequisites

[dotnet6 sdk](#)

[webview2 runtime](#) (已经在您的窗口中预装了)

### Initialization

初始化需要 5 个参数，它们都是字符串类型：

名称	描述	文件
Domain	您的 Casdoor 服务器主机/域	<a href="#">CasdoorVariables.cs</a>
Clientid	您的 Casdoor 应用程序的客户端 ID	<a href="#">CasdoorVariables.cs</a>
AppName	您的 Casdoor 应用程序的名称	<a href="#">CasdoorVariables.cs</a>
CallbackURL	您的 Casdoor 应用程序的回调 URL 路径 将是 <code>casdoor://callback</code> 如果没有提供	<a href="#">CasdoorVariables.cs</a>

名称	描述	文件
ClientSecret	您的Casdoor应用程序的客户端密钥	CasdoorVariables.cs

如果您没有设置这些参数，此项目将使用 [Casdoor 在线演示](#) 作为默认的Casdoor 服务器，并使用 [Casnode](#) 作为默认的 Casdoor 应用程序。

## Running

### Visual Studio

1. 打开casdoor-dotnet-desktop-example.sln
2. 按 Ctrl + F5 开始

### Command line

1. cd src/DesktopApp
2. dotnet run

## Preview

在运行此dotnet桌面应用程序后，您的桌面将显示一个新窗口。



如果您点击 `Casdoor Login` 按钮，您的桌面将显示登录窗口。



成功登录后，桌面上将显示一个用户配置文件窗口。它显示您的用户名。



您可以通过下面的 gif 图像预览整个过程。



# How to integrate

## Open the login window

```
var login = new Login();
// 登录成功时触发，您将在事件处理程序中收到身份验证代码
login.CodeReceived += Login_CodeReceived;
login.ShowDialog();
```

## Use auth code to get the user info

```
public async Task<string?> RequestToken(string clientId, string
clientSecret, string code)
{
    var body = new
    {
        grant_type = "authorization_code",
        client_id = clientId,
        client_secret = clientSecret,
        code
    };

    var req = new RestRequest(_requestTokenUrl).AddJsonBody(body);
    var token = await _client.PostAsync<TokenDto>(req);

    return token?.AccessToken;
}

public async Task<UserDto?> GetUserInfo(string token)
{
    var req = new
    RestRequest(_getUserInfoUrl).AddQueryParameter("accessToken",
token);
```





> 如何连接到Casdoor > 桌面 SDK > Mobile SDKs .NET MAUI App

# Mobile SDKs .NET MAUI App

The repository contains .NET MAUI app and .NET MAUI library for demonstration Casdoor authentication by Open ID Connect.

# Demonstration

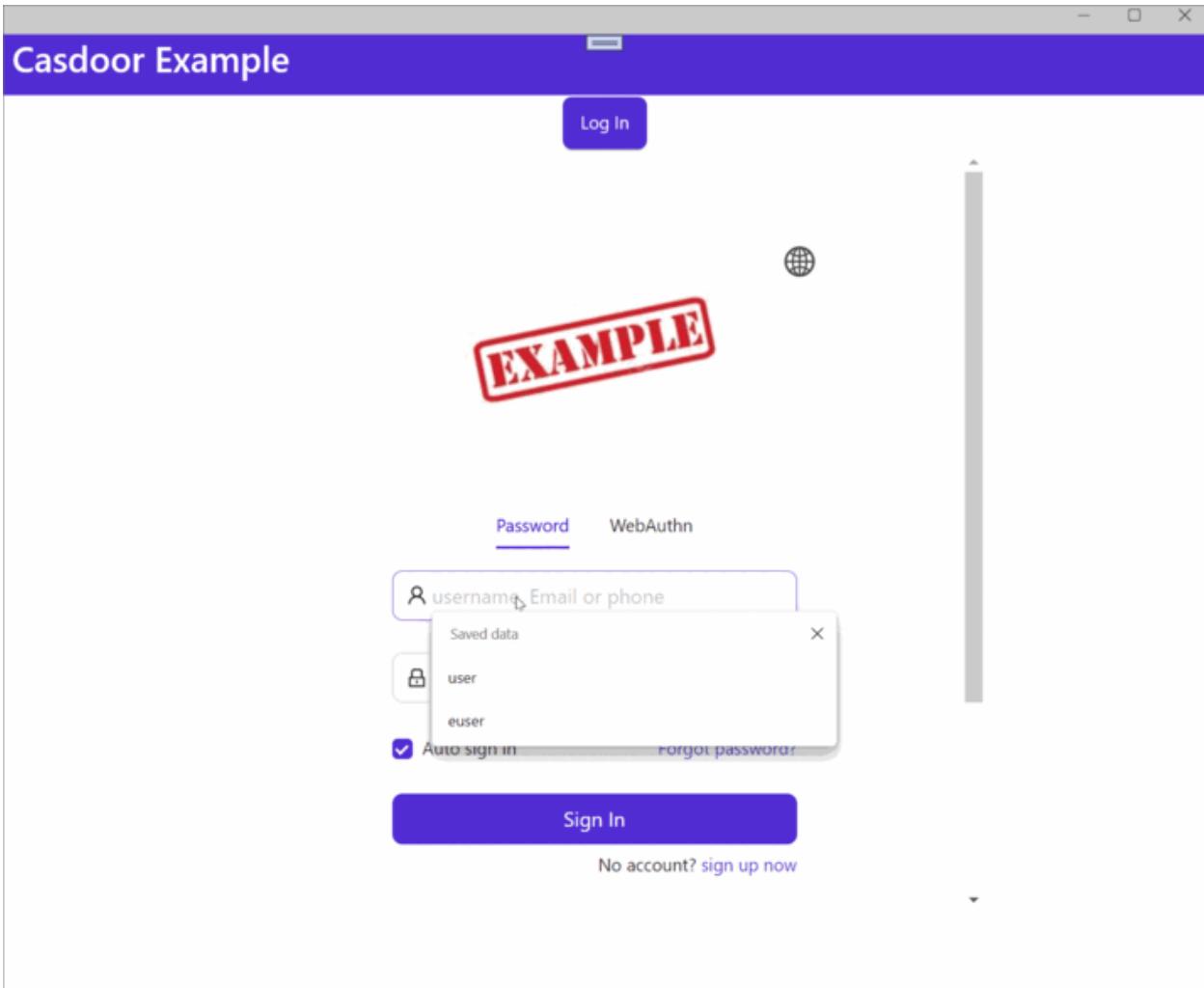
Android

21:06



.NET

# Windows



# Requirements

- [.NET 7 SDK](#) installed on your machine
- The required assets needed for your target(s) platform(s) as described [here](#)
- Visual Studio 2022 for Windows 17.3 or Visual Studio 2022 for Mac 17.4 (optional)

# Getting started

## Step 1: Create MAUI Application

Create your [MAUI Application](#).

## Step 2: Add reference

Add a reference to the `Casdoor.MauiOidcClient` in your project.

## Step 3: Add Casdoor client

Add `CasdoorClient` as singleton in the services.

```
builder.Services.AddSingleton(new CasdoorClient(new()
{
    Domain = "<your domain>",
    ClientId = "<your client>",
    Scope = "openid profile email",

#if WINDOWS
    RedirectUri = "http://localhost/callback"
#else
    RedirectUri = "casdoor://callback"
#endif
}));
```

## Step 4: Design UI

Add code to `MainPage` file.

## MainPage.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Casdoor.MauiOidcClient.Example.MainPage">

    <ScrollView>
        <VerticalStackLayout>

            <StackLayout
                x:Name="LoginView">
                <Button
                    x:Name="LoginBtn"
                    Text="Log In"
                    SemanticProperties.Hint="Click to log in"
                    Clicked="OnLoginClicked"
                    HorizontalOptions="Center" />

                <WebView x:Name="WebViewInstance" />
            </StackLayout>

            <StackLayout
                x:Name="HomeView"
                IsVisible="false">

                <Label
                    Text="Welcome to .NET Multi-platform App UI"
                    SemanticProperties.HeadingLevel="Level2"
                    SemanticProperties.Description="Welcome to dot net
Multi platform App U I"
                    FontSize="18"
                    HorizontalOptions="Center" />

                <Button
                    x:Name="CounterBtn"
                    Text="Click me"
```

## MainPage.cs

```
namespace Casdoor.MauiOidcClient.Example
{
    public partial class MainPage : ContentPage
    {
        int count = 0;
        private readonly CasdoorClient client;
        private string acsessToken;
        public MainPage(CasdoorClient client)
        {
            InitializeComponent();
            this.client = client;

#if WINDOWS
            client.Browser = new
WebViewBrowserAuthenticator(WebViewInstance);
#endif
        }

        private void OnCounterClicked(object sender, EventArgs e)
        {
            count++;

            if (count == 1)
                CounterBtn.Text = $"Clicked {count} time";
            else
                CounterBtn.Text = $"Clicked {count} times";

            SemanticScreenReader.Announce(CounterBtn.Text);
        }

        private async void OnLoginClicked(object sender, EventArgs e)
        {
            var loginResult = await client.LoginAsync();
            acsessToken = loginResult.AccessToken;
```

## Step 5: Support Android platform

Modify `AndroidManifest.xml` file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/
    android">
    <application android:allowBackup="true" android:icon="@mipmap/
        appicon" android:roundIcon="@mipmap/appicon_round"
        android:supportsRtl="true"></application>
    <uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <queries>
        <intent>
            <action
                android:name="android.support.customtabs.action.CustomTabsService"
            />
            </intent>
        </queries>
    </manifest>
```

## Step 6: Launch application

Visual Studio: Press Ctrl + F5 to start

# Qt 桌面应用程序

一个为 Casdoor 的 [Qt 桌面应用程序示例](#)。

## 如何运行示例

### 前置要求

[Qt6 sdk](#)

[OpenSSL 工具包](#)

### 初始化

您需要初始化7个参数，它们都是字符串类型：

名称	描述	文件
endpoint	您的 Casdoor 服务器主机/域	<a href="#">mainwindow.h</a>
client_id	您的 Casdoor 应用程序的客户端 ID	<a href="#">mainwindow.h</a>
client_secret	您的 Casdoor 应用程序的客户端密钥	<a href="#">mainwindow.h</a>
certificate	Casdoor 应用程序证书的公钥	<a href="#">mainwindow.h</a>
org_name	您的 Casdoor 应用程序的名称	<a href="#">mainwindow.h</a>

名称	描述	文件
app_name	您的Casdoor应用程序的名称	mainwindow.h
redirect_url	您的Casdoor 应用程序的回调URL路径将是 casdoor://callback 如果没有提供	mainwindow.h

如果您没有设置参数 端点，此项目将使用 <http://localhost:8000> 作为默认Casdoor服务器。

## 运行

### Qt Creator

1. 打开casdoor-cpp-qt-example.pro
2. 在casdoor-cpp-qt-example.pro中设置 OpenSSL 的 INCLUDEPATH
3. 按 Ctrl + R 开始

## 效果预览

在运行此Qt桌面应用程序后，您的桌面将显示一个新窗口。



如果您点击 `Sign In` 按钮，您的桌面将显示登录窗口。



成功登录后，桌面上将显示一个用户配置文件窗口，它会显示您的信息。



您可以通过下面的 gif 图像预览整个过程。



## 如何整合?

### 打开登录窗口

```
// Load and display the login page of Casdoor
m_webview->page()->load(*m_signin_url);
m_webview->show();
```

## 监听开启的应用程序事件

```
// Initialize the TcpServer object and listen on port 8080
m_tcpserver = new QTcpServer(this);
if(!m_tcpserver->listen(QHostAddress::LocalHost, 8080)) {
    qDebug() << m_tcpserver->errorString();
    close();
}
connect(m_tcpserver, SIGNAL(newConnection()), this,
SLOT(on_tcp_connected()));
```

## 使用认证码获取用户信息

```
// Get token and parse it with the JWT library
std::string token = m_casdoor->GetOAuthToken(code.toStdString());
auto decoded = m_casdoor->ParseJwtToken(token);
```



# Casdoor插件

Casdoor还为一些热门平台提供插件或中间件，例如Java的SpringBoot、PHP的WordPress、Python的Odoo等。

Casdoor 插件	语言	源代码
Spring Boot插件	Java	<a href="https://github.com/casdoor/casdoor-spring-boot-starter">https://github.com/casdoor/casdoor-spring-boot-starter</a>
Spring Boot示例	Java	<a href="https://github.com/casdoor/casdoor-spring-boot-example">https://github.com/casdoor/casdoor-spring-boot-example</a>
WordPress 插件	PHP	<a href="https://github.com/casdoor/wordpress-casdoor-plugin">https://github.com/casdoor/wordpress-casdoor-plugin</a>
Odoo 插件	Python	<a href="https://github.com/casdoor/odoo-casdoor-oauth">https://github.com/casdoor/odoo-casdoor-oauth</a>
Django 插件	Python	<a href="https://github.com/casdoor/django-casdoor-auth">https://github.com/casdoor/django-casdoor-auth</a>

完整的Casdoor官方插件列表请查看：[Casdoor repositories](#)。

# OAuth 2.0

## 介绍

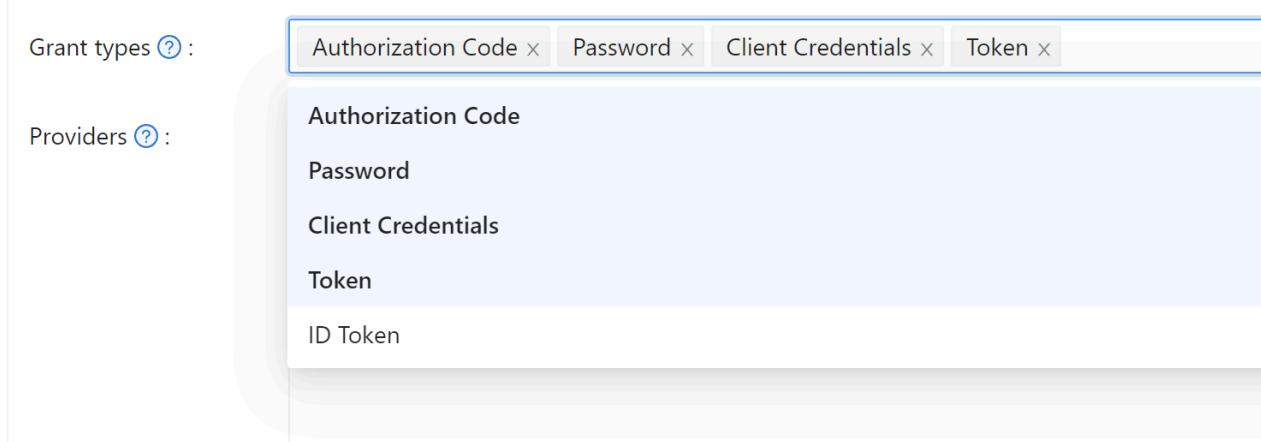
Casdoor 支持AccessToken验证客户端。在本节中，我们将向您展示如何获取AccessToken，如何验证AccessToken，以及如何使用AccessToken。

## 如何获取AccessToken

You have two ways to get the AccessToken: you can use the [Casdoor SDKs](#), for details please refer to the SDK documentation, here we will mainly show you how to use the API to get the AccessToken.

Casdoor支持四种OAuth 授予类型: [Authorization Code Grant](#), [Implicit Grant](#), [Resource Owner Password Credentials Grant](#), 和 [Client Credentials Grant](#).

出于安全考虑，Casto应用默认已开启授权码模式。如果您需要使用其他模式，请前往相应的应用程序来设置它。



## 获取授权码

首先重定向您的用户请求到:

```
https://<CASDOOR_HOST>/login/oauth/authorize?  
client_id=CLIENT_ID&  
redirect_uri=REDIRECT_URI&  
response_type=code&  
scope=openid&  
state=STATE
```

## 可用的作用域 (scope)

名称	描述
openid (no scope)	sub (用户ID), iss (发行人) 和 aud (受众)
profile	用户资料信息，包括名称、显示名称、头像
email	用户的电子邮件地址
address	用户地址
phone	用户的电话号码

### ① 信息

您的 OAuth 应用程序可以在首次重定向时带上请求的作用域。您可以指定多个作用域并使用空格（转义后为%20）分隔：

```
https://<CASDOOR_HOST>/login/oauth/authorize?  
client_id=...&  
scope=openid%20email
```

更多详情，请参阅 [OIDC 标准](#)

当您的用户通过casdoor身份验证后，他的请求会被casdoor重定向到：

```
https://REDIRECT_URI?code=CODE&state=STATE
```

现在您已经获得授权码，在你的后端应用发送 POST 请求：

```
https://<CASDOOR_HOST>/api/login/oauth/access_token
```

在你的后端应用

```
{  
  "grant_type": "authorization_code",  
  "client_id": ClientId,  
  "client_secret": ClientSecret,  
  "code": Code,  
}
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb...","  
    "id_token": "eyJhb...","  
    "refresh_token": "eyJhb...","  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

#### ⓘ 备注

Casdoor也支持 [PKCE](#) 功能。当发送获取授权码请求时，您可以通过添加两个参数来启用 PKCE。

```
&code_challenge_method=S256&code_challenge=YOUR_CHALLENGE
```

获取令牌时，您需要通过 `code_verifier` 参数来验证 PKCE。值得一提的是，启用PKCE 后，`Client_secret`并不是必需的，但如果您要发送这个参数，它的值就必须是正确的。

## 隐式授权

如果您的应用程序没有后端，您需要使用隐式授权。首先，您需要确保您启用了隐式授权，然后将您的用户请求重定向到：

```
https://<CASDOOR_HOST>/login/oauth/  
authorize?client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&response_type=token&scope=openid&state=STATE
```

当您的用户通过casdoor身份验证后，他的请求会被casdoor重定向到：

```
https://REDIRECT_URI/#access_token=ACCESS_TOKEN
```

Casdoor还支持 `id_token` 作为参数 `response_type` 的值，这是OpenID的一个功能。

## 使用资源拥有者的密码凭据授权

如果您的应用程序没有前端来重定向用户到Casdoor，那么您可能需要这个功能。

首先，您需要确保您已启用密码凭证授权，并发送一个 POST 请求：

```
https://<CASDOOR_HOST>/api/login/oauth/access_token
```

```
{  
    "grant_type": "password",  
    "client_id": ClientId,  
    "client_secret": ClientSecret,  
    "username": Username,  
    "password": Password,
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

## 使用客户端凭据授权

当应用程序没有前端时，您也可以使用客户端凭据授权。

首先，您需要确保您已启用客户端凭据授权，并发送一个 POST 请求到 [https://<CASDOOR\\_HOST>/api/login/oauth/access\\_token](https://<CASDOOR_HOST>/api/login/oauth/access_token)：

```
{  
    "grant_type": "client_credentials",  
    "client_id": ClientId,  
    "client_secret": ClientSecret,  
}
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

必须指出，以这种方式获得的AccessToken 不同于前三个，因为它与应用程序相对应，而不是与用户相对应。

## 更新访问令牌

如果您想要更新访问令牌，您可以使用上面的 `refreshToken`。

首先您需要在应用程序中设置refreshToken的到期时间(默认为0小时)，发送一个 POST 请求到 [https://<CASDOOR\\_HOST>/api/login/oauth/refresh\\_token](https://<CASDOOR_HOST>/api/login/oauth/refresh_token)

```
{  
    "grant_type": "refresh_token",  
    "refresh_token": REFRESH_TOKEN,  
    "scope": SCOPE,
```

您将得到响应:

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

## 如何验证访问令牌

目前Casdoor有支持 [token 校验](#) 的API。 Currently the endpoint is protected by Basic Authoritarian(ClientId:ClientSecret):

```
POST /api/login/oauth/introspect HTTP/1.1  
Host: CASDOOR_HOST  
Accept: application/json  
Content-Type: application/x-www-form-urlencoded  
Authorization: Basic Y2xpZW50X2lkOmNsawVudF9zZWNyZXQ=  
  
token=ACCESS_TOKEN&token_type_hint=access_token
```

您将得到以下响应:

```
{  
    "active": true,  
    "client_id": "c58c... ",  
    "username": "admin",  
    "token_type": "Bearer",  
    "exp": 1647138242,  
    "iat": 1646533442,  
    "nbf": 1646533442,  
    "sub": "7a6b4a8a-b731-48da-bc44-36ae27338817",  
    "aud": [  
        "c58c... "  
    ],  
    "iss": "http://localhost:8000"  
}
```

## 如何使用访问令牌

您可以使用AccessToken访问需要认证的 Casdoor API。

例如，请求 `/api/userinfo` 的两种不同方法。

方法 1 查询参数:

```
https://<CASDOOR\_HOST>/api/userinfo?accessToken=<your\_access\_token>
```

#### 方法 2 HTTP Bearer token

```
https://<CASDOOR\_HOST>/api/userinfo with the header: "Authorization: Bearer <your_access_token>"
```

Casdoor 将解析 access\_token，根据 scope 作用域返回对应的用户信息。您将得到响应：

```
{  
  "sub": "7a6b4a8a-b731-48da-bc44-36ae27338817",  
  "iss": "http://localhost:8000",  
  "aud": "c58c..."  
}
```

如果您需要更多用户信息，在申请访问令牌[获取授权码](#)这一步添加更多 scope。

## userinfo 和 get-account API 之间的差异

- `/api/userinfo`: 返回用户信息是OIDC 协议的一部分。Less information is returned, including only the basic information in OIDC standards. 请查看 Casdoor支持的可用的作用域（scope）。
- `/api/get-account`: 获取当前登录帐户的用户对象。这是一个只适用于Casdoor 的API，用于获取Casdoor中的 用户 的所有信息。

# CAS

## 使用 Casdoor 作为 CAS 服务器

Cassdoor 现在可以用作CAS 服务器。 目前Casdoor支持CAS3.0 的功能。

### 简介

Casdoor中CAS终端的前缀是 `<Endpoint of casdoor>/cas/<organization name>/<application name>`, 这意味着

假设Cassdoor 的端点是 `https://door.cassdoor.com`, 其中包含一个名为 `cas-java-app` 属于一个名为 `casbin`的组织, 如果我们试图让用户通过CAS登录, 那么

- `/login` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/login`
- `/logout` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/logout`
- `/serviceValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/serviceValidate`
- `/proxyValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/proxyValidate`
- `/proxy` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/proxy`
- `/validate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/validate`
- `/p3/serviceValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/p3/serviceValidate`
- `/p3/proxyValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/p3/proxyValidate`
- `/samlValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/samlValidate`

See <https://apereo.github.io/cas/6.6.x/protocol/CAS-Protocol-Specification.html> for more information about CAS and its different versions, as well as parameters for these endpoints.

### 一个示例

这里是一个官方示例 <https://github.com/aperece/cas-sample-java-webapp>, 其中包含一个使用正式的CAS java 客户端的网络应用实例 <https://github.com/aperece/java-cas-client> 我们将通过这个例子来说明如何通过CAS 连接到Casdoor。

 备注

注意：Cassdoor 目前仅支持所有三个版本的 CAS 1.0 & 2.0 & CAS 3.0

Cas 配置位于 `src/main/webapp/WEB-INF/web.yml`。

默认情况下，此应用使用 CAS 3.0，由以下配置指定。

```
<filter-name>CAS Validation Filter</filter-name>
<filter-
class>org.jasig.cas.client.validation.Cas30ProxyReceivingTicketValidationFilter</filter-
class>
```

假定您想要通过 CAS 2.0 保护此网络应用，您应该将 CAS 验证过滤器更改为以下内容。

```
<filter-name>CAS Validation Filter</filter-name>
<filter-
class>org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter</filter-
class>
```

如果您想要使用 CAS 1.0，请使用

```
<filter-name>CAS Validation Filter</filter-name>
<filter-class>org.jasig.cas.client.validation.Cas10TicketValidationFilter</filter-class>
```

对于参数“casServerUrlPrefix”，将其更改为

```
<param-name>casServerUrlPrefix</param-name>
<param-value>http://door.casdoor.com/cas/casbin/cas-java-app</param-value>
```

对于参数“casServerLoginUrl”，将其更改为

```
<param-name>casServerLoginUrl</param-name>
<param-value>http://door.casdoor.com/cas/casbin/cas-java-app/login</param-value>
```

如果您需要自定义更多的配置，请参阅 <https://github.com/apetrue/java-cas-client> 获取详细信息。

# SAML

## Overview

Using Casdoor as SAML IdP

## Keycloak

Using Casdoor as SAML IdP

## Google Workspcae

Using Casdoor as SAML IdP

## Appgate (POST)

How to use Casdoor as SAML IdP for Appgate

# Overview

Casdoor now can be used as SAML IdP. Up to now the Casdoor has supported the main feature of SAML 2.0 .

## Configuration in SP

Generally, SP needs the Single Sign-On, Issuer and Public Certificate three required fields. Most of the SP can get these fields by uploading the XML Metadata file or the XML Metadata URL to autocomplete.

The metadata of SAML endpoint in Casdoor is <Endpoint of casdoor>/api/saml/metadata?application=admin/<application name>. Suppose the endpoint of Casdoor is https://door.casdoor.com, which contains an application called app-built-in. The XML Metadata endpoint will be:

```
https://door.casdoor.com/api/saml/metadata?application=admin/app-built-in
```

And you can also find the metadata in the application edit page. Click the button to copy the URL and paste it in browser to download the XML Metadata.

```
SAML metadata ⓘ
<EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" entityID="https://door.com">
    <IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
        <KeyDescriptor use="signing">
            <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
                    <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAUgIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDIxNrb2...
                </X509Data>
            </KeyInfo>
        </KeyDescriptor>
        <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
        <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
        <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
    <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0-bindings:HTTP-Redirect" Location="https://door.casdoor.com/login/saml/authorize/admin/app-built-in"></SingleSignOnService>

```

[Copy SAML metadata URL](#)

# Configuration in Casdoor IdP

Casdoor supports both GET and POST `SAMLResponse`. Casdoor need to know what types of request the SP supports, when Casdoor sends the `SAMLResponse` to SP. You need to configure the application in casdoor based on the `SAMLResponse` type supported by your SP.

## ⓘ 信息

If you fill the `Reply URL`, Casdoor will send the `SAMLResponse` by POST Request. If the Reply URL is empty, Casdoor will use GET request. You might wonder how casdoor knows the `Reply URL` of the SP, if the `Reply URL` is empty. Actually, Casdoor can get the URL called `AssertionConsumerServiceURL` by parsing the `SAMLRequest`. And send the request with `SAMLResponse` to `AssertionConsumerServiceURL`.

The `Reply URL` will overwrites the `AssertionConsumerServiceURL` in `SAMLRequest`.

- Reply URL Type in the URL of the ACS verifying the SAML response.

The screenshot shows the Casdoor configuration interface for a specific application. The 'Grant types' section includes 'Authorization Code' and 'Password'. The 'SAML Reply URL' field is set to `https://mycontroller.mycompany.com/admin/saml`, which is highlighted with a red box. The 'Enable SAML compress' toggle switch is turned on. The URL for the screenshot is <https://casdoor.netlify.app/#/applications/1/edit>.

Grant types [?](#) : Authorization Code × Password ×

SAML Reply URL [?](#) : `https://mycontroller.mycompany.com/admin/saml`

Enable SAML compress [?](#) :

- Redirect URL Type in a unique name. This may be called `Audience` or `Entity`

ID in your SP. Make sure you fill the same Redirect URL here as in your SP.

Redirect URLs :

Redirect URLs	Add
Redirect URL	
<a href="#">appgate</a>	
<a href="https://git.casbin.com/user/oauth2/casdoor/callback">https://git.casbin.com/user/oauth2/casdoor/callback</a>	
<a href="http://localhost:3000/callback">http://localhost:3000/callback</a>	

## User profile

After logged in successfully, the user profile in the SAMLResponse Casdoor returned has three fields. The attributes in the xml and the attributes of the user in casdoor are mapped as follows:

XML Attribute Name	User field
Email	email
DisplayName	displayName
Name	name

See [https://en.wikipedia.org/wiki/SAML\\_2.0](https://en.wikipedia.org/wiki/SAML_2.0) for more information about SAML and its different versions.

## An example

The [gosaml2](#) is a SAML 2.0 implementation for Service Providers based on etree

and goxmldsig, a pure Go implementation of XML digital signatures. And we use this library to test the SAML 2.0 in Casdoor as below.

Suppose you can access Casdoor through `http://localhost:7001/`, and your Casdoor contains an application called `app-built-in` which belongs to an organization called `built-in`. The URLs, `http://localhost:6900/acs/example` and `http://localhost:6900/saml/acs/example`, should be added to the Redirect URLs in `app-built-in`.

```
import (
    "crypto/x509"
    "fmt"
    "net/http"

    "io/ioutil"

    "encoding/base64"
    "encoding/xml"

    saml2 "github.com/russellhaering/gosaml2"
    "github.com/russellhaering/gosaml2/types"
    dsig "github.com/russellhaering/goxmldsig"
)

func main() {
    res, err := http.Get("http://localhost:7001/api/saml/
metadata?application=admin/app-built-in")
    if err != nil {
        panic(err)
    }

    rawMetadata, err := ioutil.ReadAll(res.Body)
    if err != nil {
        panic(err)
    }
}
```

Run the above codes and the console will display the following message.

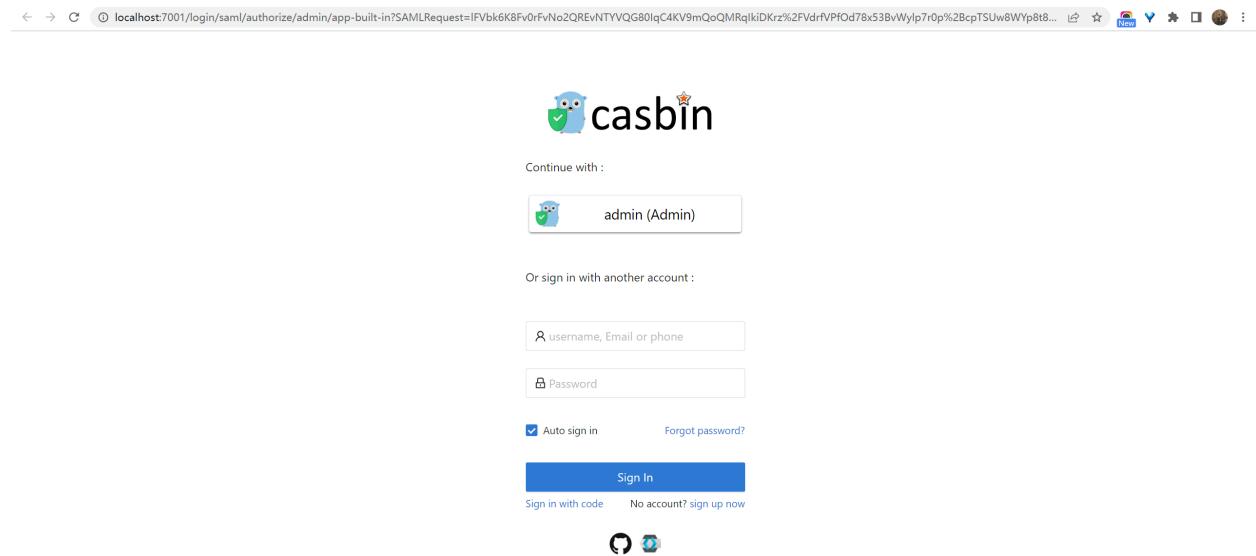
Visit this URL To Authenticate:

<http://localhost:7001/login/saml/authorize/admin/app-built-in?SAMLRequest=lFVbk6K8Fv0rFvNo2QR...>

Supply:

SP ACS URL : [http://localhost:6900/v1/\\_saml\\_callback](http://localhost:6900/v1/_saml_callback)

Click the URL to authenticate, the login page of Casdoor will display.



You will get the response messages as below after authenticating.









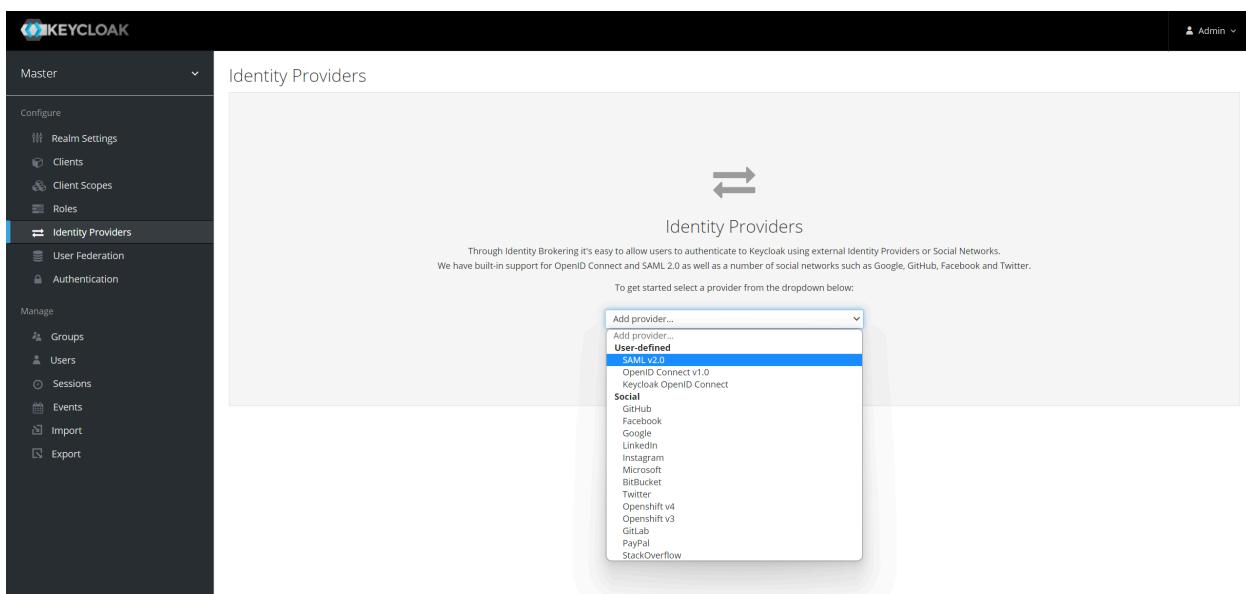
# Keycloak

## Casdoor as a SAML IdP in keycloak

This guide will show you how to configure Casdoor and Keycloak to add Casdoor as a SAML IdP in Keycloak.

### Add SAML IdP in Keycloak

Open Keycloak admin page, click Identity Providers and select SAML v2.0 from the list of providers.



#### ⓘ 信息

You can visit Keycloak SAML Identity Providers [documentation](#) to get more detailed information.

Enter the Alias and the Import from URL in Keycloak IdP edit page. The content of Import from URL can be found in the Casdoor application edit page. Click Import and the SAML config will be filled automatically.

Import External IDP Config

Import from URL: http://localhost:7001/api/saml/metadata?application=admin/app-built-in

Import

Import from file Select file

Save Cancel

You should remember the Service Provider Entity ID and then save the configuration.

## Configure SAML application in Casdoor

In the application edit page, add a redirect URL which the content of it is Service Provider Entity ID in Keycloak. And you should enable SAML compress for Keycloak.

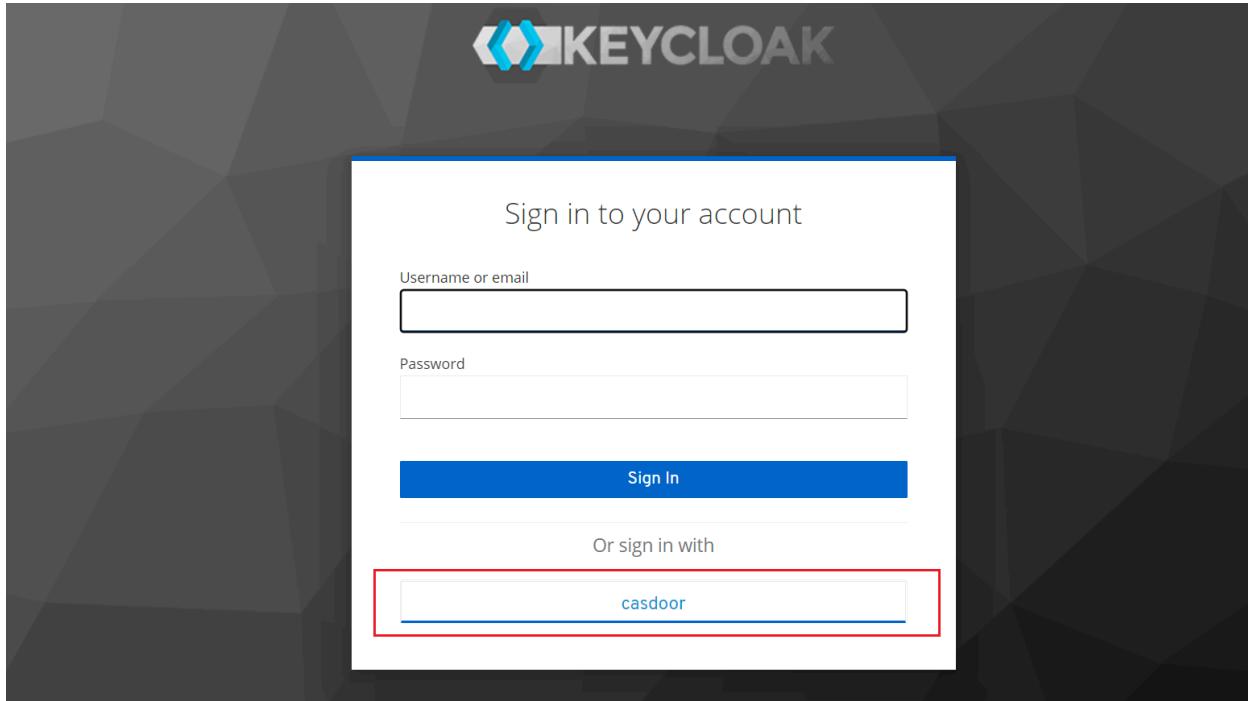
Enable SAML compress:

SAML metadata: <EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" entityID="http://localhost:8000"><IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><KeyDescriptor use="signing"><KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#"><X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAwGAWIBAgIDAeJAMAOGCCqGSIB3DQEBCwIAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29jIElcnQWhicNMjExMDE1MDgxM...</X509Data></KeyInfo></KeyDescriptor><NameIDFormat:urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat><NameIDFormat:urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat><NameIDFormat:urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat><SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0-bindings:HTTP-Redirect" Location="http://localhost:7001/login/saml/authorize/admin/app-built-in"></SingleSignOnService>

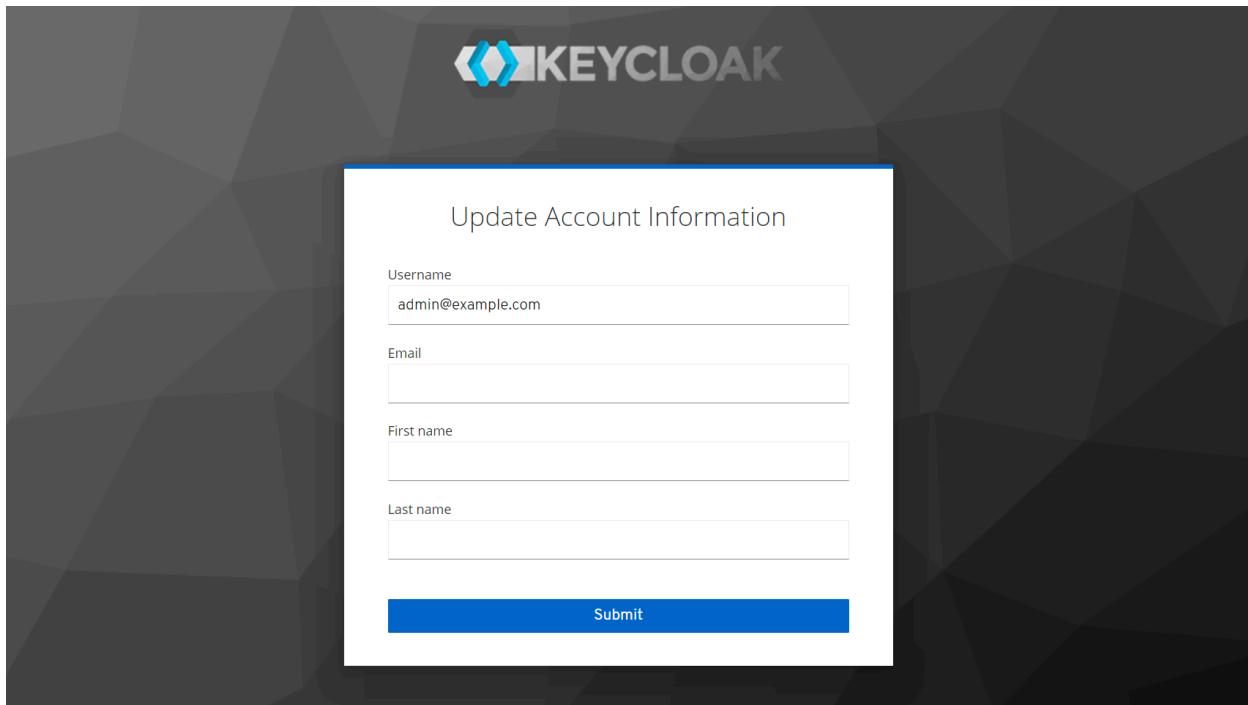
Copy SAML metadata URL

## Login using Casdoor SAML

Open the Keycloak login page and you can find the additional button that allows you to login to Keycloak using the Casdoor SAML provider.



Click on the button and you will be redirected to the Casdoor SAML provider for the authentication. After the successful authentication, you will be redirected back to Keycloak. Then you need to assign users to the application.



We also provide a demo video to demonstrate the entire process, which we hope will be helpful to you.

# Google Workspcae

## Casdoor as a SAML IdP in Google Workspace

This guide will show you how to configure Casdoor and Google Workspace to add Casdoor as a SAML IdP in Google Workspace

### Add Certificate

In Casdoor, add a certificate of type X.509 with RSA crypto algorithm and download it.

Edit Cert Save Save & Exit

Organization : admin (Shared)

Name : cert-built-in

Display name : Built-in Cert

Scope : JWT

Type : x509

Crypto algorithm : RS256

Bit size : 4096

Expire in years : 20

Certificate : Copy certificate Download certificate

Private key : Copy private key Download private key

```
-----BEGIN CERTIFICATE-----  
MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgN  
VBAoTFENh  
-----BEGIN PRIVATE KEY-----  
MIJKQIBAAKCAgEAslnpb5E1ym0fRfSDSSE8R7y+lw+RJl74e5ejrq4b8zM  
Y
```

### Configure SAML Application

In the application edit page, select the certificate you just created. Add the

domain name of the Google application you will use in the Redirect URLs, such as google.com.

Cert ⓘ: cert-built-in

Redirect URLs ⓘ:  
: Redirect URLs Add  
Redirect URL  
🔗 google.com  
🔗 gmail.com

Action
↑ ↓ 🗑️
↑ ↓ 🗑️

In the SAML reply URL field, enter `https://www.google.com/a/<your domain>/acs`, which is the ACS URL. You can find relevant information about ACS URL here: [SSO assertion requirements](#)

SAML reply URL ⓘ `https://www.google.com/a/casbin.com/acs`

Enable SAML compression ⓘ:

```
SAML metadata: <EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:idp="urn:oasis:names:tc:SAML:2.0:protocol"><IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><KeyDescriptor use="signing"><KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#><X509Data xmlns="http://www.w3.org/2000/09/xmldsig#><X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW!</X509Data></X509Certificate></KeyInfo></KeyDescriptor><NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat><NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat><NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat><SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="http://localhost:7001/login/saml/authorize/admin/wor...</SingleSignOnService><SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="http://localhost:7001/logout/saml/logout/admin/wor...</SingleLogoutService><AttributeStatement><Attribute Name="urn:oid:0.9.2342.192.12.54.1" Type="xs:string" Value="Casbin"/>
```

Copy the signin page URL. This will be used in the next step.

Providers [?](#)

Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
								No data

Preview [?](#)

[Copy signup page URL](#)



The screenshot shows a Casdoor sign-up form. It has fields for Username, Display name, Password, and Confirm. There are also links for Auto sign in and Forgot password?.

[Copy signin page URL](#)



The screenshot shows a Casdoor sign-in form. It has fields for username, Password, and a Sign In button. There are also links for Auto sign in and Forgot password?.

## Add Third-Party SAML IdP for Google Workspace

In the Google Workspace Admin console, navigate to **Security** and then **Overview**. Look for the **SSO with third-party IdP** section. Click on **Add SSO profile** to access the editing page. Check the **Set up SSO with third-party identity provider** checkbox. Paste the copied sign-in page URL into the **Sign-in page URL** and **Sign-out page URL** fields. Upload the certificate downloaded in the previous step. Click **Save** to save the changes.

The screenshot shows the Google Workspace Admin console interface. On the left, there's a navigation sidebar with various options like Home, Dashboard, Directory, Devices, Apps, Security, Overview (which is selected), Alert centre, Authentication, 2-step verification, Account recovery, Advanced Protection Programme, Login challenges, Passwordless (BETA), and Password management. The main content area has a header "Search for users, groups or settings" and a breadcrumb path "Security > SSO with third-party IDPs > Third-party SSO profile for your organisation". The main content is titled "Single Sign-On (SSO) with third-party Identity Providers (IDPs)". It includes a section for "Third-party identity provider" with a checked checkbox for "Set up SSO with third-party identity provider". Below this, there are fields for "Sign-in page URL" containing "https://localhost/login/oauth/authorize?client\_id=12" and "Sign-out page URL" containing "https://localhost/login/oauth/authorize?client\_id=12". A "Verification certificate" section shows a message "A certificate file has been uploaded" and a "REPLACE CERTIFICATE" button. A note at the bottom says "The certificate file must contain the public key for Google to verify sign-in requests." with a "Learn more" link.

## Add Users for Testing

In Google Workspace, create a user with the username "test" (you can customize the username, this is just an example).

Your new user can start using Google Workspace within 24 hours. In most cases, it should just take a few minutes.



test test

Username: test@casbin.com

Password: ● ○

[COPY PASSWORD](#) [PRINT](#)

#### Send sign-in instructions

The email will provide a link to set the password and sign in to Google Workspace

[PREVIEW AND SEND](#)



The user will be assigned licences based on your current subscriptions. [View billing](#)

[ADD ANOTHER USER](#)

[DONE](#)

In Casdoor, add a user with the same username as set in Google Workspace. Make sure to select the appropriate organization and enter the user's email address.

Organization [?](#) : built-in

ID [?](#) : 4899cef3-8eeb-485a-8f6d-12b41df0d8d2

Name [?](#) : test

Display name [?](#) : test

Avatar [?](#) :

Preview:



Upload a photo...

User type [?](#) : normal-user

Password [?](#) : [Modify password...](#)

Email [?](#) : test@casbin.com

Phone [?](#) : +1 34086653696

As an example using "google.com," follow these steps:

1. Click on the login button on the Google.com page. Enter the user's email address to initiate the login process.
2. You will be redirected to the Casdoor page. On the Casdoor page, enter the corresponding email address and password.
3. If the login is successful, you will be redirected back to google.com

Gmail Images ☰ Sign in



Google Search I'm Feeling Lucky

Google offered in: 日本語

Japan

About Advertising Business How Search works

Privacy Terms Settings

# Appgate (POST)

## Casdoor as a SAML IdP in Appgate

Appgate accepts the `SAMLResponse` sent by POST Request. If you use other SP that also supports POST request, you can refer to this document.

### Casdoor configuration

Go to your Casdoor and add a new application.

Enter basic SAML configuration in the application:

- Redirect URLs – Type in a unique name. This may be called `Audience` or `Entity ID` in your SP. See the table below.

Redirect URLs [?](#) :

Redirect URLs	Add
Redirect URL	
<a href="#">appgate</a>	
<a href="https://git.casbin.com/user/oauth2/casdoor/callback">https://git.casbin.com/user/oauth2/casdoor/callback</a>	
<a href="http://localhost:3000/callback">http://localhost:3000/callback</a>	

- Reply URL – type in the URL of the ACS verifying the SAML response, refer to the table below

Grant types [?](#) :

SAML Reply URL [?](#) :

Enable SAML compress [?](#) :

Administrator Authentication	User Authentication
Redirect URL = "AppGate"	Redirect URL = "AppGate Client"
SAML Reply URL = <a href="https://mycontroller.your-site-url.com/admin/saml">https://mycontroller.your-site-url.com/admin/saml</a>	SAML Reply URL = <a href="https://redirectserver.your-site-url.com/saml">https://redirectserver.your-site-url.com/saml</a>

## Download the XML metadata file

You can copy the URL of metadata and download the file from your browser.

Enable SAML compress [?](#) :

SAML metadata [?](#) :

```
<KeyDescriptor use="signing">
<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#>
<X509Data xmlns="http://www.w3.org/2000/09/xmldsig#>
<X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxSYXnkba...</X509Data>
</X509Data>
</KeyInfo>
</KeyDescriptor>
<NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://door.casdoor.com/login/saml/authorize/admin/app-gitea"><SingleSignOnService>
<Attribute Name="Email" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" FriendlyName="E-Mail" xmlns="urn:oasis:names:tc:SAML:2.0:assertion"></Attribute>
<Attribute Name="DisplayName" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" FriendlyName="displayName" xmlns="urn:oasis:names:tc:SAML:2.0:assertion"></Attribute>
```

# Add SAML IdP in Appgate

In your AppGate SDP console:

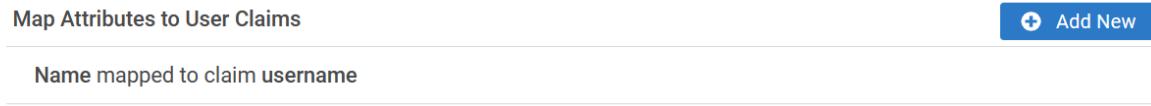
- Select System > Identity Providers
- Create a new Identity Provider
- Choose the type SAML
- Start configuring your identity provider following the details in the tables below

Administrator Authentication	
Name	Enter a unique name eg: "Casdoor SAML Admin"
Single Sign-on URL	See below
Issuer	See below
Audience	Type in the Redirect URL from the Casdoor application
Public Certificate	See below

- Upload the XML Metadata file to autocomplete Single Sign-On, Issuer and Public Certificate fields
- Click Choose a file and select the metadata file that you created previously - this will autocomplete the relevant fields

## Map Attributes

Mapping the **Name** to **username**, your completed form should look something like this:



The screenshot shows a user interface titled "Map Attributes to User Claims". At the top right is a blue button labeled "Add New" with a plus sign icon. Below the title, there is a single row containing the text "Name mapped to claim username".

## Test Integration

On your AppGate SDP Controller console:

- Log out of the admin UI
- Log in using the following information:
  - Identity Provider – choose your Azure IdP from the drop down list
  - Click **Sign in with browser** to connect to your authenticator
  - You may see the following message: "You don't have any administration rights" – this confirms that the test user credentials have been successfully authenticated by your Identity Provider.

## Access Policy

Your need to modify the access policy that the administrator can log in the Appgate by the SAML idp. Enter Builtin Administrator Policy:

Your completed form should look something like this:

## Editing Policy - Admin

- Enabled  
 Disabled

Assignment - Active when custom logic is met ▾

 Add New

Custom Logic (1 OR 3) AND 2

- 1 Identity Provider is local
- 2 user.username is admin
- 3 Identity Provider is Casdoor SAML Admin



# WebAuthn

## 概述

我们非常高兴地通知您，Casdoor现在支持通过 Webauthn 登录，这意味着：您能通过生物识别登录，例如指纹或面部识别登录，甚至是通过U盘登录，但前提是您的设备支持这些授权方法和 WebAuthn。

## 什么是 WebAuthn？

The Web Authentication API (also known as WebAuthn: <https://webauthn.io/>) is a specification written by the W3C and FIDO, with the participation of Google, Mozilla, Microsoft, Yubico, and others. API允许服务器使用公用钥匙加密而非密码来注册和认证用户。它允许服务器与强大的身份验证器集成，现已编入设备，例如Windows Hello 或 Apple's Touch ID。

简而言之，Webauthn要求用户生成公钥——私钥对，并将公钥移交给网站。当用户想登录到网站时，Web生成随机的数字，请用户用私钥加密，并将结果发送回来。收到结果后，网站将尝试使用公钥解密，并且如果解密后的数字与之前生成的随机数字相同。该用户将被视为合法用户，允许登录。我们调用 Webauthn credential的公钥与必要的信息(例如用户名或滥用信息的用户授权者) 这正是网站存储的内容。

公钥-私钥对与这三种信息（用户名、用户授权以及网站URL）截然不同。这意味着，如果(用户名、用户授权者和网站的URL) 是相同的，那么密钥对应该是相同的，反之亦然。

关于WebAuthn 技术的更多详细信息，可参阅 <https://webauthn.guide/>。

# 如何在Casdoor中使用 Webauthn ?

在登录页面中，您肯定已看到使用 WebAuthn 登录的选择。但考虑到您尚未获得 Webauthn凭据(也就是webauthn密码，这样说您可能更好地理解)，在这个教程中，我们将向您展示如何创建和管理凭据，如何再创作和管理以及如何使用凭据登录。

## 第 0 步: 修改配置并打开webauthn身份验证

您可以在conf/app.conf 中查看

```
origin = "http://localhost:8000"
```

请确保此配置是您网站的 URL。

## 只有https 支持webauthn，除非您正在使用本地主机

然后作为管理员登录，然后转到您的应用程序的编辑页面。打开开关"启用WebAuthn signin"。默认情况下，此功能未启用。

## 第 1 步：转到“我的帐户”页面

第 1 步：转到账户页面。在这个页面，您可以看到"添加 WebAuthn Credential" 按钮和一个显示您以前注册过的所有Webauthn凭据的列表。

The screenshot shows the Casdoor application configuration interface. At the top, there's a field for 'Signup application' with a placeholder '(empty)'. Below it, a section for '3rd-party logins' shows a GitHub integration with a 'Link' button. Under 'WebAuthn credentials', there's a table with one row containing a credential ID and a 'Delete' button. The 'Permissions' section includes toggles for 'Is admin', 'Is global admin', 'Is forbidden', and 'Is deleted'. At the bottom are 'Save' and 'Save & Exit' buttons.

Signup application (empty)

3rd-party logins GitHub: Link

WebAuthn credentials	Action
WebAuthn credentials Add 0AUzbyDy1SCxNyW3vkNJP1feXhwm/pHBDmMOszRRNvg=	Delete

Roles (empty)

Permissions:

- Is admin:
- Is global admin:
- Is forbidden:
- Is deleted:

Save Save & Exit

按下按钮，然后按照您设备的指示注册新凭据进入casdoor。

您可以通过列表中的“删除”按钮删除凭据。

## 第 2 步：通过webauthn登录

在这个步骤开始之前，请确保您已经退出casdoor的登录。

转到页面登录，选择webauthn登录方式，输入您的用户名并按登录按钮，然后按照您设备的说明操作。

(例如，如果您使用指纹和 Windows Hello，您就会看到类似的东西)



Windows Security

Making sure it's you

Please sign in as admin to .casdoor.com.

This request comes from Chrome, published by Google LLC.

Auto sign in      [Forgot password?](#)

[Sign in with WebAuthn](#)

[I forgot my PIN](#)

[Cancel](#)



然后您将看到您已经登录。



&gt;

开发指南

# 开发指南

## 前端

Casdoor 前端发展指南

## 生成 Swagger 文件

生成 Swagger 文件

# 前端

The source code for Casdoor's frontend is inside the `/web` folder:

<https://github.com/casdoor/casdoor/tree/master/web>

这是一个 [Create-React-App \(CRA\)](#) 项目，其经典的 CRA 文件夹结构如下：

文件/目录	描述
public	React的 HTML 根文件
src	源代码
craco.config.js	Craco 配置文件可以在此更改主题颜色 (默认情况下为蓝色)
crowdin.yml	Crowdin i18n 配置文件
package.json	NPM/Yarn 依赖文件
yarn.lock	Yarn lockfile

在 `/src` 中，有以下几个重要文件或文件夹：

文件/目录	描述
account	已登录用户的“个人资料”页面
auth	所有与身份验证相关的代码，如OAuth、SAML、注册

文件/目录	描述
	页面、登录页面、忘记密码页等。
backend	调用 Go 后端 API 的 SDK 包含所有 <code>fetch()</code> 调用
basic	Casdoor的主页(控制面板页面) 包含几个卡片小部件
common	共享界面小部件
locales	i18n translation files in JSON, synced with our Crowdin project: <a href="https://crowdin.com/project/casdoor-site">https://crowdin.com/project/casdoor-site</a>
App.js	导入JS文件，包含所有路由
Setting.js	其他代码使用的实用函数
OrganizationListPage.js	组织列表的页面，类似于所有其他“XXXListPage.js”格式的页面
OrganizationEditPage.js	组织编辑的页面，类似于所有其他“XXXEditePage.js”格式的页面

# 生成 Swagger 文件

## 概述

我们知道，beego框架为生成交换文件提供支持，以便通过称为“bee”的命令行工具清除api。Casdoor也以beego为基础。但我们发现bee生成的swagger文件未能将api分类为“@Tag”标签，我们修改了原bee以执行功能。

## 如何写comment

大多数规则与原bee comment格式完全相同，唯一的差异是api必须按照“@Tag”标签分成不同的组别，因此，开发者有义务确保正确添加此标签。下面是一个示例：

```
// @Title Login
// @Tag Login API
// @Description login
// @Param oAuthParams     query    string  true      "oAuth
parameters"
// @Param body    body    RequestForm  true      "Login
information"
// @Success 200 {object} controllers.api_controller.Response The
Response object
// @router /login [post]
func (c *ApiController) Login() {
```

具有相同“@Tag”标签的api将会被放入同一个组。

# 如何生成swagger文件

0. 以正确的格式写入 api
1. 获取资源库 <https://github.com/casbin/bee>
2. build the modified bee, for example, in the root directory of casbin/bee, run

```
go build -o mybee .
```

3. copy mybee to the base directory of casdoor
4. in that directory, run

```
mybee generate docs
```

之后你会发现生成新的swagger文件。

# 组织

## 概述

Casdoor的基本单位 — 组织

## Organization Tree

the user's groups

## Password Complexity

Support different password complexity options.

## 账户自定义

自定义用户帐户项目

 **Customize theme**

Customize themes for organizations and applications under the organization

 **Manage Multi-Factor authentication items**

Config Muti-Factor authentication items in organization

# 概述

组织是Casdoor的基本单位，它管理用户和应用。如果用户已登录过一个组织，那么此后他可以访问所有归属于该组织的应用，无需登录。

在[应用程序](#)和[提供商](#)中，选择一个组织很重要，它决定一个用户是否能够使用特定的提供商访问该应用程序。

我们还可以在Casdoor建立LDAP。欲了解更多详情，请参阅[文档](#)。

Casdoor提供了多种密码存储算法，可在组织编辑页面中选择。

名称	算法	描述	场景
plain	-	密码将以明文形式存储。(默认)	-
salt	SHA256	SHA-256 是一个获得专利的加密哈希函数，输出256位长的值。	-
md5-salt	MD5	MD5 message-digest algorithm是一种加密中断但仍广泛使用的哈希函数，可产生128位哈希值。	Discuz!
bcrypt	bcrypt	bcrypt 是一个密码哈希函数，用于安全地对密码进行哈希和加密。	Spring Boot, WordPress
pbkdf2-salt	SHA256	PBKDF2 是一个简单的加密密钥函数，能	Keycloak

名称	算法	描述	场景
	和 PBKDF2	够抗拒 dictionary attacks 和 rainbow table attacks。它最初在 Casdoor 里用于Keycloak 语法。如果您通过Keycloak 同步方式导入用户，请选择此选项。	

### 提示

Besides logging into Casdoor via an application (which redirects to Casdoor for SSO), a Casdoor user can also choose to directly log into Casdoor via the organization's login page: `/login/<organization_name>`, e.g., <https://door.casdoor.com/login/casbin> in the demo site.

# Organization Tree

Groups is a collection of users under an organization. A user can be in multiple groups.

## Group properties

- `Owner` Owner organization of the group
- `Name` Group name (unique)
- `displayName`
- `CreatedTime`
- `UpdatedTime`
- `Type` Groups have two types: `Physical` and `Virtual`, a user can only be in one `Physical` group, but can be in multiple `Virtual` groups.
- `ParentGroup` Parent group of the group (The parent group of top groups in the organization is the organization itself)

## Manage groups

There are two ways to manage groups:

1. In the groups list pages, you can see all the groups in organizations.

The screenshot shows the Casdoor Groups page. At the top, there is a navigation bar with links for Home, Organizations, Groups (which is highlighted in blue), Users, Roles, Permissions, Models, Adapters, Applications, Providers, Chats, Messages, and Admin. Below the navigation bar is a search bar with placeholder text "Search for organizations or users". The main content area is a table titled "Groups" with a "Add" button. The table has columns for Name, Organization, Created time, Updated time, Display name, Type, Parent group, and Action. The "Action" column contains "Edit" and "Delete" buttons. A red vertical bar on the right side of the table indicates a feedback feature. At the bottom of the table, there is a pagination bar showing "9 in total" and "10 / page".

Name	Organization	Created time	Updated time	Display name	Type	Parent group	Action
casdoor_virtual	built-in	2023-06-12 12:37:44	2023-06-12 12:37:51	Casdoor Project Virtual Team	Virtual		<button>Edit</button> <button>Delete</button>
casbin_virtual	built-in	2023-06-12 12:37:18	2023-06-12 12:37:36	Casbin Project Virtual Team	Virtual		<button>Edit</button> <button>Delete</button>
dev_frontend	built-in	2023-06-12 09:43:18	2023-06-12 12:35:51	Dev (Frontend)	Physical		<button>Edit</button> <button>Delete</button>
dev_backend	built-in	2023-06-12 09:20:28	2023-06-12 12:35:58	Dev (Backend)	Physical		<button>Edit</button> <button>Delete</button>
dev	built-in	2023-06-09 18:19:06	2023-06-12 12:36:08	R & D	Physical		<button>Edit</button> <button>Delete</button>
sales	built-in	2023-06-09 01:27:19	2023-06-12 12:36:27	Sales	Physical		<button>Edit</button> <button>Delete</button>
marketing	built-in	2023-06-09 01:26:16	2023-06-12 12:36:32	Marketing	Physical		<button>Edit</button> <button>Delete</button>
hr	built-in	2023-06-09 01:25:46	2023-06-12 12:36:43	HR	Physical		<button>Edit</button> <button>Delete</button>
sales_and_marketing	built-in	2023-06-09 01:23:35	2023-06-12 12:36:57	Sales & Marketing	Physical		<button>Edit</button> <button>Delete</button>

## 2. Click the Groups button in organization list page

The screenshot shows the Casdoor Organizations page. At the top, there is a navigation bar with links for Home, Organizations (which is highlighted in blue), Groups, Users, Roles, Permissions, Models, Adapters, Applications, Providers, Chats, Messages, and Admin. Below the navigation bar is a search bar with placeholder text "Search for organizations or users". The main content area is a table titled "Organizations" with a "Add" button. The table has columns for Name, Created time, Display name, Favicon, Website URL, Password type, Password salt, Default, and Action. The "Action" column contains "Groups" and "Users" buttons, each with "Edit" and "Delete" buttons. A red vertical bar on the right side of the table indicates a feedback feature. At the bottom of the table, there is a pagination bar showing "4 in total" and "10 / page".

Name	Created time	Display name	Favicon	Website URL	Password type	Password salt	Default	Action
saas	2023-05-31 00:05:42	SaaS Users		<a href="https://saas.casbin.com">https://saas.casbin.com</a>	plain			<button>Groups</button> <button>Users</button> <button>Edit</button> <button>Delete</button>
gsoc	2021-02-11 23:26:20	GSoC Community		<a href="https://gsoc.com.cn">https://gsoc.com.cn</a>	plain			<button>Groups</button> <button>Users</button> <button>Edit</button> <button>Delete</button>
casbin	2021-02-11 23:26:20	Casbin Organization		<a href="https://forum.casbin.com">https://forum.casbin.com</a>	plain			<button>Groups</button> <button>Users</button> <button>Edit</button> <button>Delete</button>
built-in	2021-02-10 00:37:06	Built-in Organization		<a href="https://door.casdoor.com">https://door.casdoor.com</a>	plain			<button>Groups</button> <button>Users</button> <button>Edit</button> <button>Delete</button>

Then you can see the tree structure of the groups in the organization.

The screenshot shows the Casdoor interface for managing organizations. On the left, there is a sidebar with a tree view of groups:

- Root node: Casdoor Project Virtual Team
- Child node: Casbin Project Virtual Team
- Child node: R & D
  - Child node: Dev (Frontend)
  - Child node: Dev (Backend)
- Child node: HR
- Child node: Sales & Marketing
  - Child node: Sales
  - Child node: Marketing

To the right of the tree view is a table listing users:

Organization	Application	Name	Created time	Display name
built-in	app-built-in	牛头	2023-06-16 16:16:35	牛头
built-in	app-built-in	喝咖啡就大蒜	2023-06-15 10:58:48	喝咖啡就大蒜
built-in	app-built-in	danceshow	2023-06-15 10:53:48	街舞show
built-in	app-built-in	TT珍惜	2023-06-15 10:36:46	TT珍惜
built-in	app-built-in	hashjoin	2023-06-15 01:01:17	hashjoin
built-in	app-built-in	zhangyaphet@gmail.com	2023-06-14 15:50:23	pengwei zhang

Here is a video show you how to manage groups:

The screenshot shows the Casdoor interface for managing groups. The table lists the following groups:

Name	Organization	Created time	Updated time	Display name	Type	Parent group	Action
group_smf8bi	built-in	2023-06-13 23:25:31	2023-06-13 23:25:36	总部	Virtual	built-in	<button>Edit</button> <button>Delete</button>
group_zxak7d	built-in	2023-06-11 23:28:45	2023-06-13 23:24:36	New Group - zxak7d	Virtual	New Group - nahuap	<button>Edit</button> <button>Delete</button>
group_1t8lrr	built-in	2023-06-09 09:39:44	2023-06-13 23:24:46	美工	Virtual	研发子部门	<button>Edit</button> <button>Delete</button>
group_nahuap	built-in	2023-06-09 09:27:47	2023-06-12 09:36:45	New Group - nahuap	Virtual		<button>Edit</button> <button>Delete</button>
group_38ii7o	built-in	2023-06-07 21:48:49	2023-06-11 09:49:13	研发子部门	Virtual		<button>Edit</button> <button>Delete</button>
group_gnrtip9	built-in	2023-06-07 20:59:10	2023-06-13 23:24:51	实体组3	Physical	built-in	<button>Edit</button> <button>Delete</button>
group_Sacaox	forum	2023-06-06 08:24:33	2023-06-13 23:25:12	实体组2	Physical	forum	<button>Edit</button> <button>Delete</button>
group_3tt9wf	forum	2023-06-06 08:20:14	2023-06-13 23:25:06	顶级2	Virtual	forum	<button>Edit</button> <button>Delete</button>
group_azpdif	forum	2023-06-06 08:19:32	2023-06-09 10:50:25	顶级1	Virtual		<button>Edit</button> <button>Delete</button>
group_r89hga	built-in	2023-06-05 14:41:41	2023-06-12 09:38:28	研发部1	Virtual		<button>Edit</button> <button>Delete</button>

Groups can be also edit in user profile.

Title <a href="#">?</a> :	1122
Homepage <a href="#">?</a> :	
Bio <a href="#">?</a> :	
Tag <a href="#">?</a> :	222
Karma <a href="#">?</a> :	333
Signup application <a href="#">?</a> :	app-built-in
Groups <a href="#">?</a> :	 Dev (Frontend)   Casdoor Project Virtual Team 
Roles <a href="#">?</a> :	
Permissions <a href="#">?</a> :	

# Password Complexity

Casdoor supports customizing password complexity options for user password in each organization.

## Supported Complexity Options

We currently support 5 options:

- `AtLeast6`: The password must have at least 6 characters
- `AtLeast8`: The password must have at least 8 characters
- `Aa123`: The password must contain at least one uppercase letter, one lowercase letter and one digit
- `SpecialChar`: The password must contain at least one special character
- `NoRepeat`: The password must not contain any repeated characters

If you want to use multiple options, you can select them on the organization edit page:

1. Click the Edit button in organization list page

Name	Created time	Display name	Favicon	Website URL	Action
organization_pquaah	2023-06-07 18:03:53	New Organization - pquaah		<a href="https://door.casdoor.com">https://door.casdoor.com</a>	<a href="#">Groups</a> <a href="#">Users</a> <a href="#">Edit</a> <a href="#">Delete</a>
built-in	2023-05-22 23:52:27	Built-in Organization		<a href="https://example.com">https://example.com</a>	<a href="#">Groups</a> <a href="#">Users</a> <a href="#">Edit</a> <a href="#">Delete</a>

2. Then select the option you need in the **Password complexity options** column.

The screenshot shows the Casdoor web interface for managing organizations. The top navigation bar includes links for Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Applications, Providers, and Admin. The 'Organizations' tab is selected. Below the navigation is a form titled 'Edit Organization' with fields for Name (built-in), Display name (Built-in Organization), Favicon (https://cdn.casbin.org/img/casbin/favicon.ico), URL (https://example.com), Preview (an owl icon), Password type (plain), Password salt, and Password complexity options. The 'Password complexity options' section contains several validation rules: 'The password must have at least 8 characters', 'The password must contain at least one special character', 'The password must not contain any repeated characters', 'The password must contain at least one uppercase letter, one lowercase letter and one digit'. To the right of these rules is a dropdown menu for supported country codes, listing Germany +49, United Kingdom +44, India +91, and another entry with a delete icon. The bottom of the form has 'Edit Organization' and 'Save' buttons, and a 'Save & Exit' button highlighted in blue.

# Password Complexity Validation

We support password complexity validation on the following pages:

1. Sign up page

The screenshot shows the Casdoor sign-up interface. At the top, there are three tabs: "Overview | Casdoor - An Open", "Built-in Organization", and "Built-in Organization". The URL in the address bar is "localhost:7001/signup". The main form has fields for "Username" (filled with "eee"), "Display name" (filled with "222"), "Password", "Confirm", and "Email". Below these are two sets of fields for "Email code" and "Phone code", each with an "Enter your code" input field and a "Send Code" button. There is also a "Phone" field set to "+1". A checkbox for "Accept Terms of Use" is checked, and a "Sign Up" button is at the bottom right, with a link "Have account? sign in now" nearby. The footer says "Powered by Casdoor".

## 2. Forget password page

The screenshot shows the Casdoor retrieve password page. The title is "Casdoor" and the subtitle is "Retrieve password". Below the title are three tabs: "Account" (selected), "Verify", and "Reset". The "Account" tab contains fields for "Password" and "Confirm". A "Change Password" button is at the bottom. The footer says "Powered by Casdoor".

## 3. User edit page



# 账户自定义

## 介绍

在组织中，您可以自定义用户的 账户项。 这包括每个项目是否是 可见。 If visible, its view rule and modify rule.

When you customize account items in an organization, this configuration takes effect on the home page of all members of that organization.

## 如何定制？

帐户项目有四个属性：

属性	可选值	描述
Name	-	账户名称。
Visible	<input checked="" type="radio"/> 是 / <input type="radio"/> 否	选择此账户项是否在用户主页上可见。
ViewRule	<input type="radio"/> 规则项	选择用于查看帐户项目的规则。
ModifyRule	<input type="radio"/> 规则项	选择用于修改帐户项的规则。

输入组织编辑页面，您可以找到以下内容：

Name	visible	viewRule	modifyRule	Action
Organization	<input checked="" type="checkbox"/>	Public	Admin	  
ID	<input checked="" type="checkbox"/>	Public	Immutable	  
Name	<input checked="" type="checkbox"/>	Public	Admin	  
Display name	<input checked="" type="checkbox"/>	Public	Self	  
Avatar	<input checked="" type="checkbox"/>	Public	Self	  
User type	<input checked="" type="checkbox"/>	Public	Admin	  
Password	<input checked="" type="checkbox"/>	Self	Self	  
Email	<input checked="" type="checkbox"/>	Public	Self	  
Phone	<input checked="" type="checkbox"/>	Public	Self	  
Country/Region	<input checked="" type="checkbox"/>	Public	Self	  
Location	<input checked="" type="checkbox"/>	Public	Self	  
Affiliation	<input checked="" type="checkbox"/>	Public	Self	  
Title	<input checked="" type="checkbox"/>	Public	Self	  
Homepage	<input checked="" type="checkbox"/>	Public	Self	  
Bio	<input checked="" type="checkbox"/>	Public	Self	  
Tag	<input checked="" type="checkbox"/>	Public	Admin	  
Signup application	<input checked="" type="checkbox"/>	Public	Admin	  
3rd-party logins	<input checked="" type="checkbox"/>	Self	Self	  

Casdoor 提供非常简单的操作来配置：

- 将项目设置为可见或不可见

Name	visible	viewRule	modifyRule
Organization	<input checked="" type="checkbox"/>	Public	Admin
ID	<input type="checkbox"/>		
Name	<input checked="" type="checkbox"/>	Public	Admin
Display name	<input checked="" type="checkbox"/>	Public	Self
Avatar	<input checked="" type="checkbox"/>	Public	Self
User type	<input checked="" type="checkbox"/>	Public	Admin

- 设置查看和修改规则

visible	viewRule	modifyRule	Action
<input checked="" type="checkbox"/>	Public	Admin	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input type="checkbox"/>	Public		<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input checked="" type="checkbox"/>	Self	Admin	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input checked="" type="checkbox"/>	Admin	Self	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>

有 3 条规则：

- 公开：每个人都有权限
- 自我：用户有自己的权限
- 管理员：管理员拥有权限

## 账户列表

以下是帐户项目中的所有字段。 关于描述详情，您可以看 [用户](#)。

- 组织
- ID
- 姓名
- 显示名称
- 头像
- 用户类型
- 密码
- 邮件
- 手机号
- 国家/地区

- 城市
- 附属机构
- 职务
- 个人主页
- 自我介绍
- 标签
- 注册应用
- 第三方登录
- 属性
- 是否为管理员
- 是否为全局管理员
- 是否被禁用
- 是否已删除

# Customize theme

Casdoor allows you to customize theme to satisfy UI diversity from business or brand requirements, including primary color, border radius.

In Casdoor, the scope of theme includes global, organization, and application.

1. Global scope: this is the default theme of Casdoor and is applied to any organization that chooses to follow the global theme. It can only be modified in Casdoor source code, there is no way to modify it in web UI.
2. Organization scope: the theme for an organization can be customized in the organization edit page. The theme takes effect in all the Casdoor after-login pages for the users in the organization and the entry pages (signup, signin, forget password, etc.) of the applications that follow the organization theme.
3. Application scope: the theme for an application can be customized in the application edit page. The theme takes effect in the the entry pages (signup, signin, forget password, etc.) of the application.

## Customize organization theme

We provide a demo to demonstrate how to config theme in organization:

The screenshot shows the Casdoor web application interface. At the top, there is a theme editor grid with two columns of 12 items each. The first column includes 'Roles', 'Permissions', '3rd-party logins', 'Properties', 'Is admin', 'Is global admin', 'Is forbidden', 'Is deleted', 'WebAuthn credentials', and 'Managed accounts'. The second column contains 'Public', 'Immutable', and various user roles like 'Admin', 'Self', and 'Public' with edit icons. Below the theme editor is a 'Theme' section with 'Follow global theme' and 'Customize theme' buttons. Underneath is an 'LDAPs' section with a table showing one entry: 'BuildIn LDAP Server' with 'example.com:389' as the server and 'ou=BuildIn,dc=example,dc=com' as the base DN. The table has columns for 'Server Name', 'Server', 'Base DN', 'Auto Sync', 'Last Sync', and 'Action' (with 'Sync', 'Edit', and 'Delete' buttons). At the bottom are 'Save' and 'Save & Exit' buttons, and a footer note 'Powered by Casdoor'.

### ① 信息

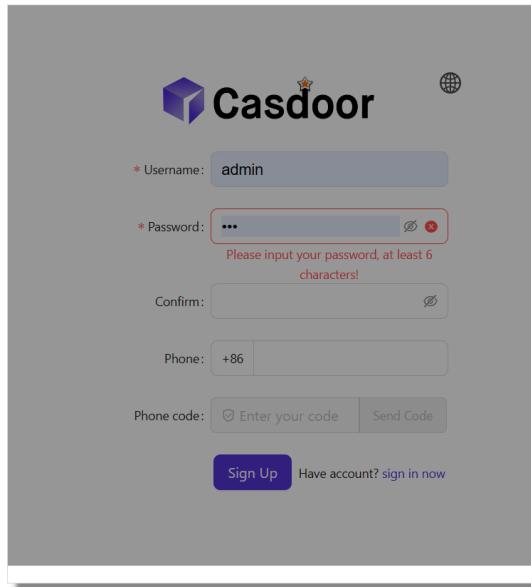
If your account organization is same as the organization you are editing, after you save the configuration, it will take effect immediately as the video above show. But if they are different, you need to log in the organization to see the effect.

## Customize application theme

Applications customize theme use the same theme editor as the organization. But even more conveniently, you can preview the theme in the preview panel.

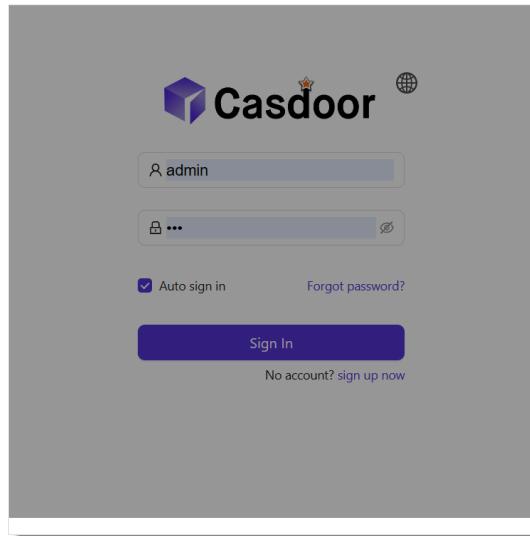
Preview ⓘ

 Copy signup page URL



The screenshot shows the Casdoor Signup page. It features a header with the Casdoor logo and a globe icon. Below the header are several input fields: a "Username" field containing "admin", a "Password" field with placeholder text "Please input your password, at least 6 characters!", a "Confirm" field, a "Phone" field with "+86", and a "Phone code" field with "Enter your code" and a "Send Code" button. At the bottom are "Sign Up" and "Have account? sign in now" buttons.

 Copy signin page URL



The screenshot shows the Casdoor Signin page. It has a header with the Casdoor logo and a globe icon. The "Username" field contains "admin". Below it is a "Password" field with a lock icon and a visibility toggle. To the right of the password field are "Auto sign in" and "Forgot password?" checkboxes. A large "Sign In" button is at the bottom, with a "No account? sign up now" link below it.

Background URL

# Manage Multi-Factor authentication items

## Add Multi-Factor authentication item in organization

In organization, admin can add the Multi-factor authentication item in account items so that user can config the Multi-factor authentication in their owner profile page.

The screenshot shows the Casdoor web interface for managing organization settings. At the top, there are fields for 'Master password' (set to \*\*\*), 'Languages' (English, 中文, Español, Français, Deutsch, Indonesia, 日本語, 한국어, Русский, Tiếng Việt), 'Init score' (0), 'Soft deletion' (disabled), and 'Is profile public' (disabled). Below these, the 'Account items' section is displayed, listing various account-related fields: Organization, ID, Name, Display name, Avatar, User type, Password, Multi-factor authentication, Email, and Phone. Each field has a 'Visible' switch, a 'View rule' dropdown (Public or Admin), a 'Modify rule' dropdown (Admin, Immutable, Self), and a set of 'Action' buttons (Edit, Delete). The 'Multi-factor authentication' row is highlighted with a red border.

Name	Visible	View rule	Modify rule	Action
Organization	<input checked="" type="checkbox"/>	Public	Admin	<span>Edit</span> <span>Delete</span>
ID	<input checked="" type="checkbox"/>	Public	Immutable	<span>Edit</span> <span>Delete</span>
Name	<input checked="" type="checkbox"/>	Public	Admin	<span>Edit</span> <span>Delete</span>
Display name	<input checked="" type="checkbox"/>	Public	Self	<span>Edit</span> <span>Delete</span>
Avatar	<input checked="" type="checkbox"/>	Public	Self	<span>Edit</span> <span>Delete</span>
User type	<input checked="" type="checkbox"/>	Public	Admin	<span>Edit</span> <span>Delete</span>
Password	<input checked="" type="checkbox"/>	Self	Self	<span>Edit</span> <span>Delete</span>
Multi-factor authentication	<input checked="" type="checkbox"/>	Self	Self	<span>Edit</span> <span>Delete</span>
Email	<input checked="" type="checkbox"/>	Public	Self	<span>Edit</span> <span>Delete</span>
Phone	<input checked="" type="checkbox"/>	Public	Self	<span>Edit</span> <span>Delete</span>

# Manage Multi-Factor authentication items

You can manage Multi-Factor authentication to determine which Multi-Factor authentication method are available to users.

There are two rules for managing Multi-Factor authentication items:

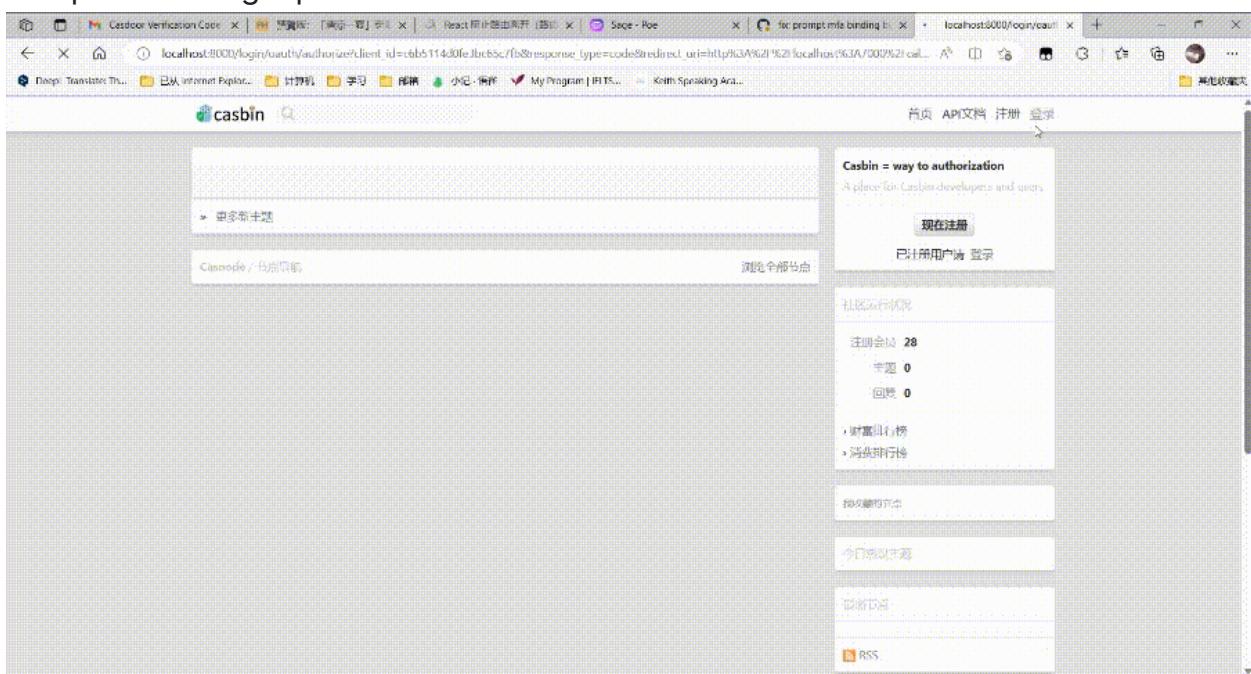
- optional: Users can choose whether to enable this type Multi-Factor authentication.
- prompt: If the user does not enable this Multi-Factor authentication mode, the user will be prompted to enable it after logging in to Casdoor.
- required: Users must enable this Multi-Factor authentication method.

MFA items		Add			
Name		Rule	Action		
Phone	▼	Prompt ▼	▲ ▼	trash	
Email	▼	Optional ▼	▲ ▼	trash	
App	▼	Required ▼	▲ ▼	trash	

The image of the notification that prompts users to enable Multi-Factor authentication



This video shows that after the Multi-Factor authentication method is set to required, the user needs to enable Multi-Factor authentication before they can complete the login process.





&gt;

应用

# 应用

## 概览

Casdoor应用概述

## 术语参考

术语参考

## 应用程序配置

配置应用程序身份验证

## 注册项目表

配置注册项表以创建自定义注册页面

## 自定义登录界面

自定义您的应用程序登录页面

## 指定登录组织

指定登录页面中的登录组织

## Tags

Configure your application tags

# 概览

Casdoor的每个应用程序都是一个application，它们之间没有关联并且不会相互影响，这意味着，你可以根据需要单独部署或删掉其中任何一个应用程序。

如果您想要使用 Casdoor 为您的页面应用提供登录服务，您可以将其添加为Casdoor 应用程序。

这样用户在访问组织中的所有应用程序时就无需重复登录。

应用程序配置非常灵活和简单。 您可以设置是否允许密码登录或第三方登录，配置您想要用户用以登录的第三方应用程序，您甚至可以自定义应用程序的注册条目等。

在本章中，您将学到如何从零开始将Casdoor配置进您自己的程序。

让我们一起探索吧！

# 术语参考

- `Name` 创建应用的名称
- `CreatedTime` 应用创建时间
- `DisplayName` 应用向公众展示的名称
- `Logo` 将显示在登录和注册页面上的应用图标
- `HomepageUrl` 应用程序主页的 url
- `Description` 应用程序的介绍
- `Tags` Only users with tag listed in the application tags can login
- `Organization` The organization that the APP belongs to
- `EnablePassword` If users can login via password
- `EnableSignUp` If users can sign up. If not, accounts of the application
- `SignupItems` fields that need to be filled in when users register
- `Providers` Provide all kinds of services for the applications (such as OAuth, Email, SMS service)
- `ClientId` OAuth client id
- `ClientSecret` OAuth client secret
- `RedirectUris` Casdoor will navigate to one of the uris if user logged in successfully
- `TokenFormat`: The format of the generated token. It can be either `JWT` (containing all `User` fields) or `JWT-Empty` containing all non-empty values
- `ExpireInHours` Login will expire after hours
- `SigninUrl`
- `SignupUrl` If you provide a sign up service independently out of Casdoor, please fill the url here
- `ForgotUrl` Same as `SignupUrl`

- `AffiliationUrl`

# 应用程序配置

将 Casdoor 部署到服务器并设置组织后，您可以使用 Casdoor 配置应用程序身份验证并部署应用程序！

让我们看看如何使用 Casdoor 配置您的应用程序的身份认证系统吧！

## ① 备注

例如，让我们看看如何为一个论坛使用 [Casnode](#) 设置身份验证。

首先，创建您的应用程序并填写必要的配置。

然后，选择您创建的组织，以便组织中的用户也可以使用该应用程序。



The screenshot shows the Casbin web application's organization management page. At the top, there is a navigation bar with links for Home, Organizations (which is underlined), Users, Providers, Applications, Resources, Tokens, Records, and Swagger. On the far right, there is a user profile icon labeled "Admin". Below the navigation bar, a modal window titled "Edit Organization" is open. It contains two input fields: "Name" with the value "my\_organization" and "Display name" with the value "My Organization". A blue "Save" button is located at the top right of the modal.

由于组织的名称为 `my_organization`，请从下拉菜单中选择它。

Edit Application Save

Name ? : my\_forum

Display name ? : My Forum

Logo ? : URL: [https://cdn.casbin.com/logo/logo\\_1024x256.png](https://cdn.casbin.com/logo/logo_1024x256.png)

Preview:



Home ? :

Description ? :

Organization ? : built-in

Client ID ? : my\_organization  
built-in

Then I want my users can use Casdoor to complete authentication when they are signing up, so I fill the redirect url here as <https://your-site-url.com/callback>

:::注意事项

所以，我们需要记住提供商应用程序中的 `callback URL` 是 Casdoor的回调url，而 Casdoor中的 `Redirect URL` 是 您网站的callback url。

进一步解释

为使身份验证过程正常工作，详细步骤如下：

Casdoor 使用 `Client ID` 和 `Client Secret` 从 GitHub、Google 或其他提供程序获

取身份验证。

If the authentication success, GitHub callback to Casdoor to tell Casdoor authentication success, so the GitHub authorization callback URL should be my Casdoor callback URL which is <http://your-casdoor-url.com/callback>, then Casdoor tells the application authentication success which means the Casdoor callback URL should be my application callback URL, that is <http://your-site-url.com/callback>.

...

最后，通过添加提供程序并设置其属性来添加可以注册的第三方应用程序。

Name	canSignUp	canSignIn	canUnlink	prompted	Action
provider_casbin_email					
provider_casbin_sms					
provider_storage_aliyun_oss					
provider_casdoor_github_localhost	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_github	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_google	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_qq	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_wechat	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_facebook	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_gitee	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
provider_casdoor_gitlab	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

You need to enable JavaScript to run this app.

 提示

注意，如果您不希望用户使用**用户名/密码**访问您的应用， 您可以关闭 **Password On** 按钮，这样用户就只能使用第三方服务访问应用：

Token expire [?](#) :  Hours

Password ON [?](#) :

Enable signup [?](#) :

# 注册项目表

在应用程序配置页面上，我们可以配置注册项表来创建一个自定义注册页面。我们可以在此注册项表上添加或删除任何注册项。

Signup items :

Name	Visible	Required	Prompted	Rule	Action		
ID				Random			
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
Display name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		None			
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
Confirm	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
Email	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Normal			
Phone	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
Agreement	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		None			

关于每个注册项的详细说明，请见下表。

参数	可选值	描述
Name	-	注册项名称
visible	True / False	选择此注册项是否在注册页面上可见
required	True / False	选择此注册项是否必须要填写
prompted	True / False	选择当用户忘记填写此注册项时是否给出提示
rule	Rule	规则。 规则可以为这个注册项添加一些自定义要求。 详

参数	可选值	描述
	Items	细规则见下表。
Action	-	用户可以将这个注册项向上移动、向下移动，或删除这个注册项。

So far, the signup items that support configuration rules include ID, Display name, Email and Agreement.

规则	可选规则	描述
ID	Random / Incremental	选择用户ID是随机生成还是递增生成。
Display name	None / Real name / First, last	选择名称的展示方式。选择 None 将展示 Display name，选择 Real name 将展示 Real name，选择 First, last 将展示 First name 和 last name。
Email	Normal / No verification	选择是否验证邮箱验证码。选择 Normal 将验证邮箱验证码。选择 No verification 则不验证邮箱验证码。
Agreement	None / Signin / Signin (Default True)	Select whether the user needs to confirm terms of use when logging in. Choose None to not display terms of use, and users can log in directly. Choose Signin to require users to confirm the terms before logging in. Choose

规则	可选规则	描述
		Signin (Default True) to set the terms confirmed by default, and users can log in directly.

### ① 备注

例如，我想设置需要注册邮箱的注册页面，但不需要邮件验证码来验证此邮箱。

首先，我添加了一些注册所需的注册项目，例如ID、用户名、密码、电子邮件。

Name	visible	required	prompted	rule	Action
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Incremental	<input checked="" type="checkbox"/> No verification
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Email	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			

然后设置电子邮件的验证规则为 No verification，然后生成的注册页面就可以实现该效果。



\* Username:

\* Password:

\* Email:

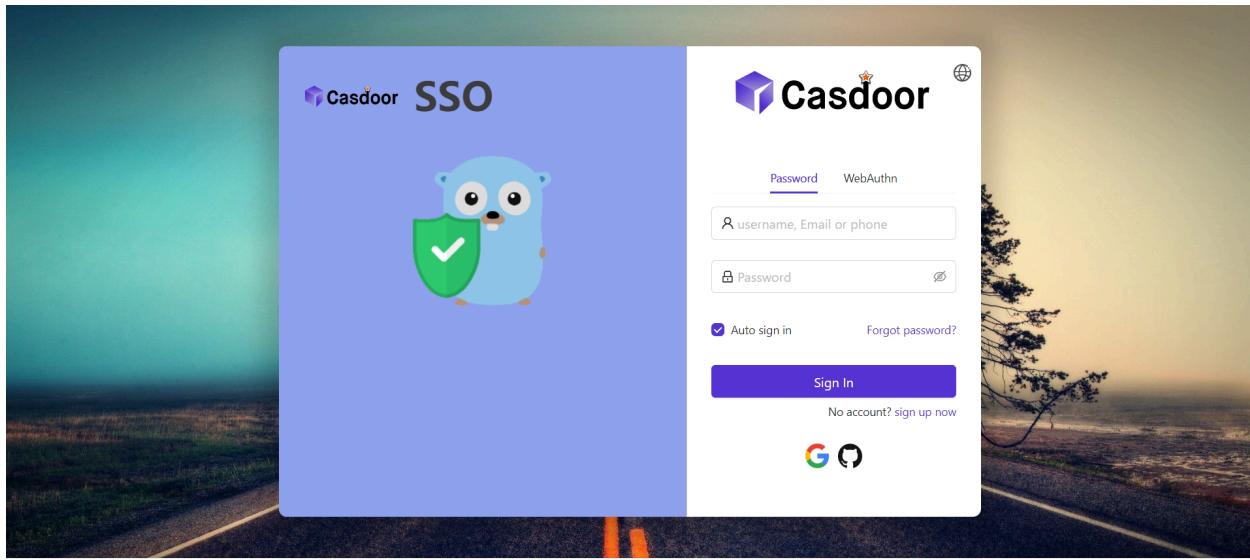
Sign Up

Have account? [sign in](#)

now

# 自定义登录界面

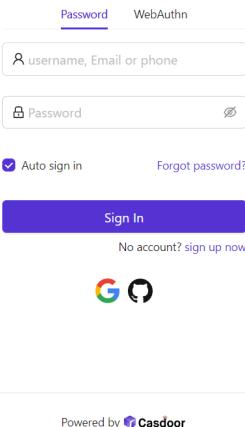
您已创建应用程序。下面将向您展示如何自定义应用程序的登录界面。在本指南中，我们将创建以下应用程序登录页面：



让我们开始吧！

## 1、添加背景图像

首先，我们要添加背景图像。默认背景是白色的，看起来很简约。



- **背景URL** 背景图像URL。

选择您喜欢的背景图像并填写**背景 URL**。如果您填写了有效的URL，预览区域将会显示您所选择的图像。

Background URL ②:	URL ②: <input type="text" value=""/>
Preview:	
Form CSS ②: <input type="text"/>	
Form position ②: <input type="button" value="Left"/> <input type="button" value="Center"/> <input type="button" value="Right"/> <input type="button" value="Enable side panel"/>	

## 2、自定义登录面板

现在我们在第一步操作结束的页面：



Powered by Casdoor

现在您需要添加一些css使面板看起来更美观。 您可以复制下面的代码并粘贴到字段  
**Form CSS** 中。

```
<style>
.login-panel{
    padding: 40px 30px 0 30px;
    border-radius: 10px;
    background-color: #ffffff;
    box-shadow: 0 0 30px 20px rgba(0, 0, 0, 0.20);
}
</style>
```

Background URL  
URL : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:



Form CSS :

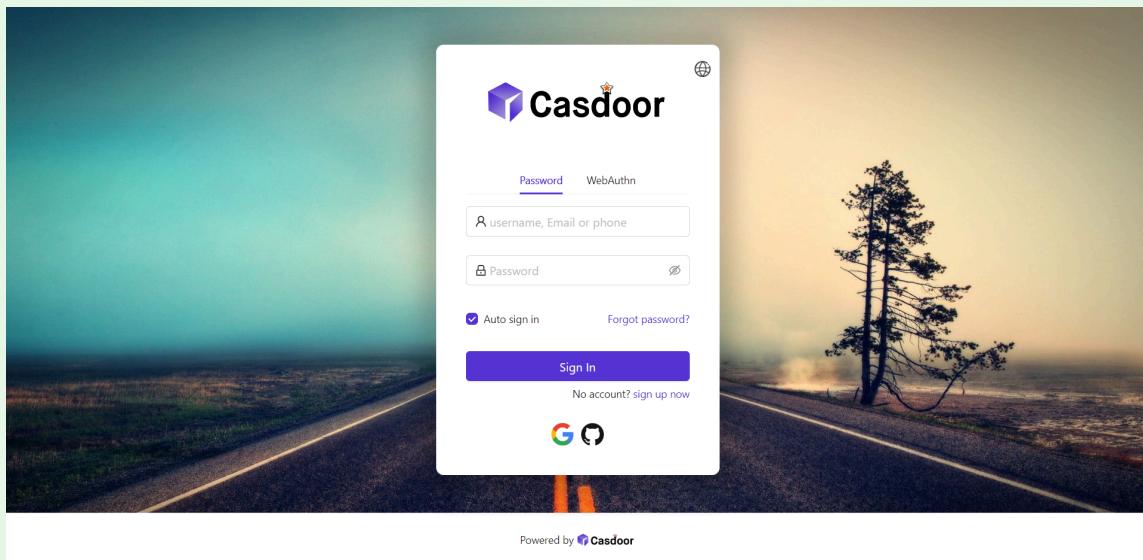
Form position :

### 💡 提示

当您编辑 `form CSS` 时，如果值为空，编辑器将显示默认值。但并不是填在字段中。您需要复制内容并粘贴。

...

填写 `表单 CSS` 后，不要忘记在底部保存配置。好，让我们看看效果。



### 3、选择面板位置

现在登录页面就比刚开始的页面更为美观。 我们还为您提供了三个按钮，可以决定面板的位置。

Background URL  
URL : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:

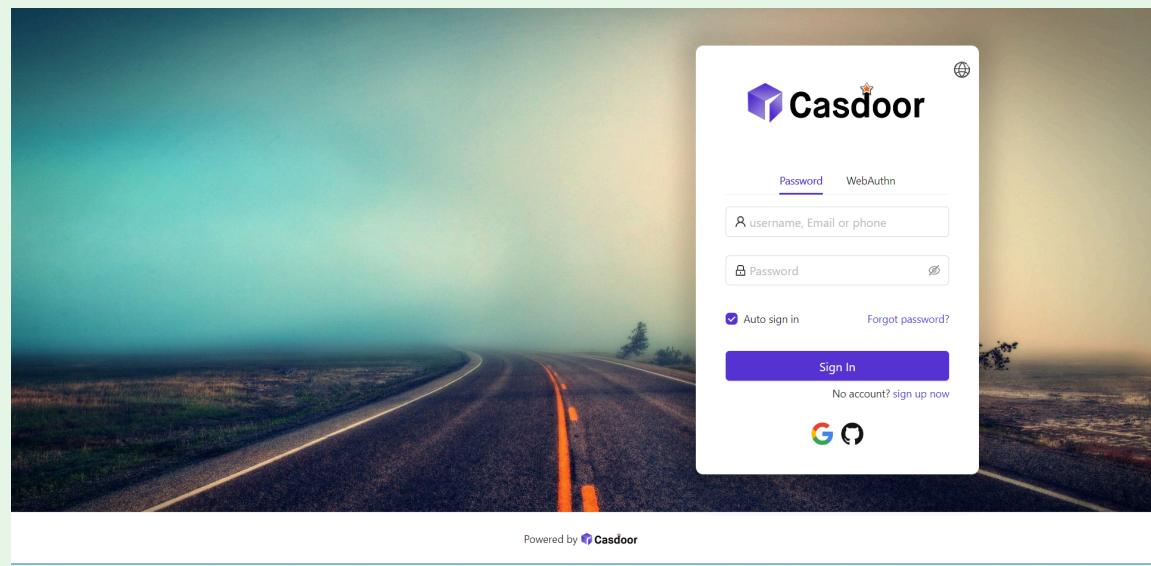


Form CSS :

```
<style>.login-panel{ padding: 40px 30px 0 30px; border-radius: 10px; background-color: #fff; }</style>
```

Form position :

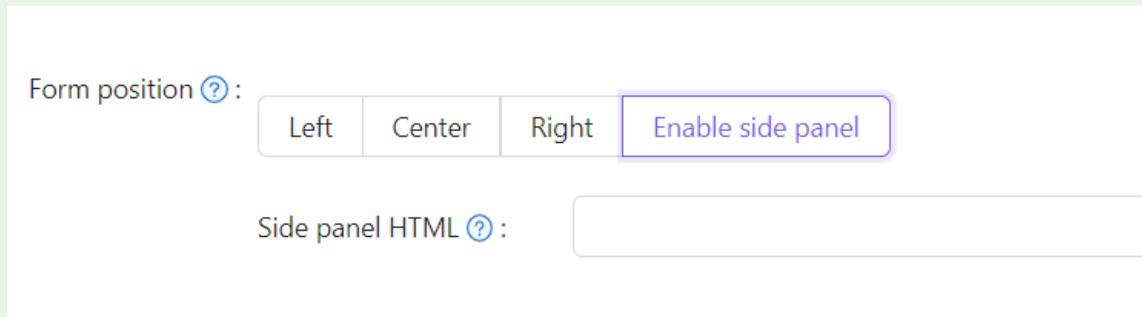
例如，选择 Right 按钮：



## 4、启用侧面板

您现在将看到如何启用侧面板并自定义样式。

首先选择按钮。在 **启用侧面板** 模式时，面板在中间位置。



然后编辑 **侧面板HTML**，它决定了侧面板中显示的内容。与 **Form CSS** 相同，我们提供了一个默认模板。复制粘贴即可。

```
<style>
  .left-model{
    text-align: center;
    padding: 30px;
    background-color: #8ca0ed;
    position: absolute;
    transform: none;
    width: 100%;
    height: 100%;
  }
  .side-logo{
    display: flex;
    align-items: center;
  }
  .side-logo span {
    font-family: Montserrat, sans-serif;
    font-weight: 900;
```

好，让我们看看效果。带有徽标和图像的侧面板已显示出来，但整体效果看起来差强人意。



您需要修改 `Form CSS` 中的一些css。

Background URL  
② : URL ② : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:

Form CSS ② : `<style> .login-panel{ padding: 40px 30px 0 30px; border-radius: 10px; background-color: #ffffff; box-shadow: 0 0 30px 20px rgba(0,0,0,0.1); }</style>`

Form position ② :

Side panel HTML ② : `<style> .left-model{ text-align: center; padding: 30px; background-color: #8ca0ed; position: absolute; left: -300px; top: 0; width: 300px; height: 100%; }</style>`

Signup items ② :

最后代码如下。

```
<style>
```

## ① 信息

.login-panel, .login-form 是div的类名。它们对应于页面的不同区域。欲了解更多详情，您可以通过开发者工具搜索。在确认类名称后，您可以在这里填写 CSS 以更灵活的方式自定义登录页面。

最后，我们便设置好了一个美观的登录页面。



## 总结

我们来总结一下：我们已添加背景图像，自定义了登录面板的风格，并且启用了侧边板。

更多关于Casdoor应用程序的介绍：

- [Customize theme](#) Customize the theme, including primary color, border radius.
- [注册项目表](#)
- [应用程序配置](#)

感谢您的阅读！



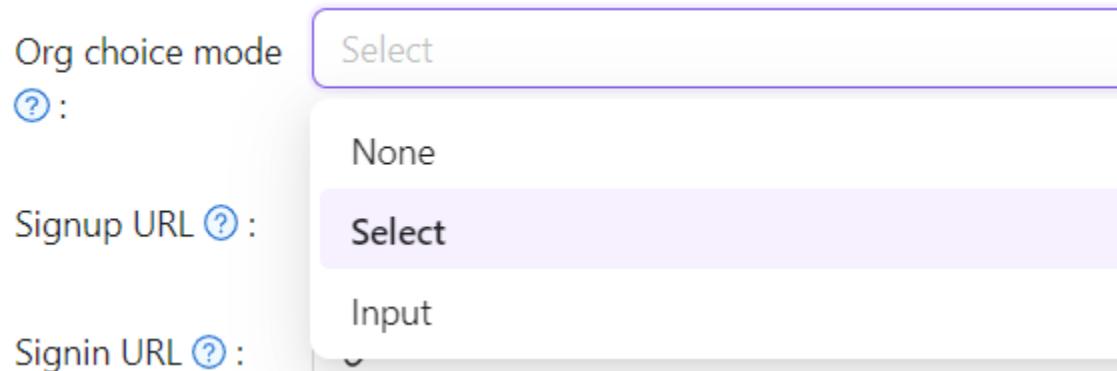
# 指定登录组织

Here will show you how to enable specify login organization page for the application.

例如，端点 `/login` 是默认登录到 `built-in` 组织。 You can enable the specify login organization page in `app-built-in` application that belong to `built-in` organization. So that the user can select an organization to login. After the user selects the organization, it will redirect to `/login/<organization>`.

## 配置

In the application edit page, you can see the `Org choice mode` config. You can select the mode in the dropdown list.



- None: Don't show the organization select page.
- Input: The user can input the organization name in the input box.
- Select: The user can select the organization in the dropdown list.



Please type an organization to sign in

built-in

Confirm



Please select an organization to sign  
in



built-in

forum

test

Star

### ① 信息

The organization select page only shows when the route is `/login`, `<organization>/login`. That means the application should be set as default application in the organization or the app-built-in.



# Tags

The application tags are used to restrict whether users can login the application. Specifically, only users with tags listed in the application tags are allowed to login. For example, application `dev_app` have tags `dev, prd`, only users with tag `dev` or `prd` can login `dev_app`. Note that admin and global admin are not affected by application tags.

In the application edit page, you can see the `Tags` config and add tags here.

The screenshot shows the Casdoor application edit page for an application named "only\_tag23". The "Applications" tab is selected. The "Tags" field contains "tag2" and "tag3". Other fields include "Name" (only\_tag23), "Display name" (New Application - 907akg), "Logo" (URL: https://cdn.casbin.org/img/casdoor-logo\_1185x256.png), "Preview" (showing the Casdoor logo), "Home" (empty), "Description" (empty), "Organization" (built-in), and "Cert" (cert-built-in). The "Tags" field is highlighted with a red border.

Here is a video show you how application tags work (download to see it):





&gt;

权限

# 权限

## 概述

使用 Casbin 管理用户在组织中的访问权限

## 权限配置

使用 Casbin 管理用户在组织中的访问权限

## 开放的 Casbin API

使用 Casbin 管理用户在组织中的访问权限

## Adapter

Config adapter and basic CRUD to policy

# 概述

## Introduction

Casdoor 中每个组织的应用程序都共享该组织中的所有用户，因此这些用户都可以访问这些应用程序。有时您可能想限制用户访问某些应用程序或某个应用程序中的某些资源。在这种情况下，你可以使用由 [Casbin](#) 实现的 [Permission](#) 功能。

在继续深入前，你应该对 Casbin 的工作原理及其相关概念有所了解，例如 Model、Policy、Adapter。简而言之，Model 定义了你的权限策略结构，且请求应当如何匹配这些权限策略以及其所产生的作用。Policy 则是你具体权限规则描述。Casbin 在获得了 Model 和 Policy 信息后便可对到来的请求强制执行权限控制。Adapter 作为抽象层为 Casbin 的执行器屏蔽了 Policy 的来源，使得 Policy 可以存放于各处，例如文件或数据库。

回到 Casdoor 的权限配置话题。在 Casdoor Web UI 中，你可以在 [Model](#) 配置项中为你的组织添加 Model，并在 [Permission](#) 配置项中为你的组织添加 Policy。With [Casbin Online Editor](#), you can get Model and Policy files suitable for your usage scenarios. 你可以很容易地将 Model 文件通过 Casdoor Web UI 导入至 Casdoor 中供内置 Casbin 使用。但对于 Policy 而言（即 Casdoor Web UI 中的 [Permission](#) 配置项），则需要在此进行一些额外的说明。让我们在后文继续提及。

正如你的应用需要通过 Casdoor 内置的 Casbin 来强制实施权限控制一样，作为一个内置应用，Casdoor 内部也通过 Casbin 使用自己的 Model 和 Policy 来控制 API 接口的调用权限。然而，Casdoor 可以通过内部代码来调用 Casbin，而外部应用却无法做到。因此，Casdoor 为向外部应用暴露了调用内置 Casbin 的 API 接口。我们将在后文向你展示这些 API 接口的定义及其使用方式。

在本章的末尾，我们将使用一个实际例子来向您展示外部应用程序如何与 Casdoor 协作

进行权限控制。

让我们开始吧！

# 权限配置

让我们依次解释 Permission 配置页面中的每一项。

- **Organization**: 该策略所属的组织名，一个组织可以拥有多个权限策略文件。
- **Name**: 权限策略名称，在组织中全局唯一。用于标识该权限策略文件。
- **Display name**: 没什么重要的。
- **Model**: 描述权限策略结构及其匹配模式的模型文件名称。
- **Adapter**: 注意！在当前版本中该字段描述的是存放权限策略的数据库表名，而非在 Casdoor Web UI 中的 Adapter 菜单项所配置的适配器名称。Casdoor 会使用其自身的数据库来存放正在配置的权限策略。如果该字段为空，权限策略将被存储至 `permission_rule` 表中，否则将存储在指定的数据库表中。如果指定的表名在 Casdoor 所使用的数据库中不存在，将会自动创建。我们强烈建议**为不同模型指定不同的适配器**，因为将所有的策略保存在同一个表格中可能导致冲突。
- **Sub users**: 哪些用户将被应用该权限策略。
- **Sub roles**: 如果使用了RBAC模型，哪些角色将被应用该权限策略。这将为该角色中的每一个用户添加诸如 `g user role` 的权限策略。
- **Sub domains**: 哪些域将被应用该权限策略。
- **Resource type**: 事实上，在当前版本中，对于外部希望鉴权的应用来说，Casdoor 并没有使用该字段。你可以暂且忽略它。
- **Resources**: 这个字段描述了你希望强制实施权限控制的资源。但请注意，这里的资源并非在 Casdoor Web UI 的 Resources 菜单项中所配置的资源。你可以在此处添加任何你所希望的字符串，例如一个 URL 或者一个文件名。
- **Actions**: 这个字段描述了操作资源的动作。与资源相同，它可以是任何你所希望的字符串，例如 Http Method 或其他自然语言。但是请注意，Casdoor 会将这些字符串全部转化成为小写再存储。另外，Casdoor 将会为每一个资源应用所有的动作。你无法指定某个动作只对某些资源生效。

- **Effect**: 这个选项对于 Casdoor 自身控制应用访问权限生效。如果你希望外部应用使用 Casdoor 所暴露的接口来强制实施权限控制，它不会起到任何作用。你应该在 Model 文件中描述模式匹配后的作用。

你可以看到，这个配置页面几乎就是为 **(sub, obj, act)** 模型量身定制的。

# 开放的 Casbin API

## 介绍

Let's assume that your application front-end has obtained the `access_token` of the logged-in user, and now wants to authenticate the user for some access. You cannot simply place the `access_token` to the HTTP request header to use these APIs, because Casdoor uses the `Authorization` field to check the access permission. Like any other APIs provided by Casdoor, the `Authorization` field consists of the application client id and secret, using the [Basic HTTP Authentication Scheme](#). 它看起来像 `Basic XXX`。因此，Casbin API 应当被应用的后端服务器调用。Here are steps about how to do it.

1. 前端通过 HTTP 请求头将 `access_token` 传递到后端服务器。
2. The backend server gets the user id from the `access_token`.

提前说明，这些接口也几乎是为 `(sub, obj, act)` 模型所设计的（就现阶段而言）。The `permissionId` in the url parameters is the identity of the applied permission policy, which consists of the organization name and the permission policy name (ie `organization name/permission name`). The body is the request format defined by the Casbin model of the permission, usually representing `sub`, `obj` and `act` respectively.

除了请求强制执行权限控制的 API 接口以外，Casdoor 也提供了其它一些有助于外部应用获取权限策略信息的接口，也一并列在此处。

## Enforce

请求：

```
curl --location --request POST 'http://localhost:8000/api/enforce?permissionId=example-org/example-permission' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic client_id_and_secret' \
--data-raw '[{"example-org/example-user", "example-resource", "example-action"}]
```

响应：

```
{  
    "status": "ok",  
    "msg": "",  
    "sub": "",  
    "name": "",  
    "data": [  
        true  
    ],  
    "data2": null  
}
```

## BatchEnforce

请求：

```
curl --location --request POST 'http://localhost:8000/api/batch-enforce?permissionId=example-org/example-permission' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic client_id_and_secret' \
--data-raw '[[{"example-org/example-user", "example-resource", "example-action"}, [{"example-org/example-user2", "example-resource", "example-action"}, {"example-org/example-user3", "example-resource", "example-action"}]]'
```

响应:

```
{  
    "status": "ok",  
    "msg": "",  
    "sub": "",  
    "name": "",  
    "data": [  
        [  
            true,  
            true,  
            false  
        ]  
    ],  
    "data2": null  
}
```

## GetAllObjects

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-objects' \  
--header 'Authorization: Basic client_id_and_secret'
```

响应:

```
[  
    "app-built-in"  
]
```

## GetAllActions

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-actions' \
--header 'Authorization: Basic client_id_and_secret'
```

响应:

```
[  
  "read",  
  "write",  
  "admin"  
]
```

## GetAllRoles

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-roles' \
--header 'Authorization: Basic client_id_and_secret'
```

响应:

```
[  
  "role_kcx661"  
]
```

# Adapter

Casdoor supports using the UI to connect the adapter and manage the policy rules. In Casbin, the policy storage is implemented as an adapter (aka middleware for Casbin). A Casbin user can use an adapter to load policy rules from a storage, or save policy rules to it.

# Adapter

- `type` : Adapter type. Now support database adapter.
- `Host`
- `Port`
- `User`
- `Password`
- `Database type` : Now support MySQL, PostgreSQL, SQL server, Oracle, SQLite 3.
- `Database` : The database name.
- `Table` : The table name. If the table does not exist, it will be created.
- `model` : You can select one model belonging to the organization of the adapter.

Edit Adapter Save Save & Exit

Organization <small>?</small> :	built-in
Name <small>?</small> :	casdoor_adapter
Type <small>?</small> :	Database
Host <small>?</small> :	localhost
Port <small>?</small> :	3306
User <small>?</small> :	root
Password <small>?</small> :	123456
Database type <small>?</small> :	MySQL
Database <small>?</small> :	casdoor
Table <small>?</small> :	casbin_rule
Model <small>?</small> :	casbin_rule
Policies <small>?</small> :	<span>Sync</span>

## ⓘ 信息

After fill all the fields, please don't forget to **save** the config. Then click the **sync** button to load the policy rules. The policy rules will be shown in the below table.

Policies ?:

<span>Sync</span>	<span>Add</span>	Rule Type	V0	V1	V2	V3	V4	V5	Option
p		built-in	*	*	*	*	*	*	<span>edit</span> <span>trash</span>
p		app	*	*	*	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	POST	/api/signup	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	POST	/api/get-email-and-phone	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	POST	/api/login	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	GET	/api/get-app-login	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	POST	/api/logout	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	GET	/api/logout	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	GET	/api/get-account	*	*	*	<span>edit</span> <span>trash</span>
p		*	*	GET	/api/userinfo	*	*	*	<span>edit</span> <span>trash</span>

< 1 2 3 4 5 >

Is enabled ?: toggle

up

# Basic CURD

If you connect the adapter successfully, you can make basic CURD to the policy rules.

- Add

The screenshot shows a table of policy rules. The columns are labeled V0, V1, V2, V3, V4, V5, and Option. The rows show various API endpoints and methods. The 'Option' column contains icons for edit and delete.

Policies ⓘ	Sync	Add	V0	V1	V2	V3	V4	V5	Option
Rule Type									
p	built-in	↳	*	*	*	*	*	*	edit delete
p	*	*	POST	/api/signup	*	*	*	*	edit delete
p	*	*	POST	/api/get-email-and-phone	*	*	*	*	edit delete
p	*	*	POST	/api/login	*	*	*	*	edit delete
p	*	*	GET	/api/get-app-login	*	*	*	*	edit delete
p	*	*	POST	/api/logout	*	*	*	*	edit delete
p	*	*	GET	/api/logout	*	*	*	*	edit delete
p	*	*	GET	/api/get-account	*	*	*	*	edit delete
p	*	*	GET	/api/userinfo	*	*	*	*	edit delete
p	*	*	POST	/api/webhook	*	*	*	*	edit delete



You can only add one policy at one time. The newly added policy is in the first row in the table, but actually, it will be saved in the last row. So next time you sync the policies, they will appear in the last row of the table.

- Edit

casbin_rule																																																																																								
Model :		casbin_rule																																																																																						
Policies :		<input type="button" value="Sync"/> <input type="button" value="Add"/> <table border="1"> <thead> <tr> <th>Rule Type</th><th>V0</th><th>V1</th><th>V2</th><th>V3</th><th>V4</th><th>V5</th><th>Option</th></tr> </thead> <tbody> <tr> <td>p</td><td>built-in</td><td>*</td><td>POST</td><td>*</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>app</td><td>*</td><td>*</td><td>POST</td><td>/api/signup</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>POST</td><td>/api/get-email-and-phone</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>POST</td><td>/api/login</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>GET</td><td>/api/get-app-login</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>POST</td><td>/api/logout</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>GET</td><td>/api/logout</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>GET</td><td>/api/get-account</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>*</td><td>*</td><td>GET</td><td>/api/userinfo</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> </tbody> </table>							Rule Type	V0	V1	V2	V3	V4	V5	Option	p	built-in	*	POST	*	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	app	*	*	POST	/api/signup	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	POST	/api/get-email-and-phone	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	POST	/api/login	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	GET	/api/get-app-login	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	POST	/api/logout	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	GET	/api/logout	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	GET	/api/get-account	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	*	*	GET	/api/userinfo	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>
Rule Type	V0	V1	V2	V3	V4	V5	Option																																																																																	
p	built-in	*	POST	*	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	app	*	*	POST	/api/signup	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	POST	/api/get-email-and-phone	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	POST	/api/login	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	GET	/api/get-app-login	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	POST	/api/logout	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	GET	/api/logout	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	GET	/api/get-account	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
p	*	*	GET	/api/userinfo	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																																																																																	
<	1	2	3	4	5	>																																																																																		

- Delete

User :	root																								
Password :	123456																								
Database type :	MySQL																								
Database :	casdoor																								
Table :	casbin_rule																								
Model :	casbin_rule																								
Policies :	<input type="button" value="Sync"/> <input type="button" value="Add"/> <table border="1"> <thead> <tr> <th>Rule Type</th><th>V0</th><th>V1</th><th>V2</th><th>V3</th><th>V4</th><th>V5</th><th>Option</th></tr> </thead> <tbody> <tr> <td>p</td><td>*</td><td>*</td><td>GET</td><td>/api/get-default-application</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> <tr> <td>p</td><td>test</td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td><input type="button" value="Sync"/> <input type="button" value="Delete"/></td></tr> </tbody> </table>	Rule Type	V0	V1	V2	V3	V4	V5	Option	p	*	*	GET	/api/get-default-application	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>	p	test	*	*	*	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>
Rule Type	V0	V1	V2	V3	V4	V5	Option																		
p	*	*	GET	/api/get-default-application	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																		
p	test	*	*	*	*	*	<input type="button" value="Sync"/> <input type="button" value="Delete"/>																		
<	1	2	3	4	5	>																			



&gt;

提供商

# 提供商

## 概述

添加第三方服务到您的应用程序

## OAuth

21 个项目

## Email

2 个项目

## SMS

5 个项目



## 存储

7 个项目



## SAML

3 个项目



## 支付

4 个项目



## 验证码

7 个项目



## Web3

1 个项目

# 概述

Casdoor 使用提供商为平台提供第三方服务。 在本章中，您将学习如何为Casdoor添加提供商。

## 我们所拥有的

现在，我们有六种类型的提供商：

- **OAuth 提供商**

允许用户通过其他 OAuth 应用程序登录。 您可以将 GitHub、Google、QQ 和其他许多 OAuth 应用程序添加到Casdoor。 欲了解更多详情，请参阅 [OAuth](#)。

- **短信提供商**

当用户想要验证他们的电话号码时，Casdoor将发送短信给他们。 短信提供者被用来发送Casdoor短信。

- **电子邮件提供商**

电子邮件提供者与短信提供者类似。

- **存储提供商**

Casdoor允许用户使用本地文件系统或云端服务存储文件。

- **支付服务提供商**

Casdoor 可以添加付款提供者，用于在产品页面上添加付款方法。 目前，受支持的

付款提供者包括支付宝、微信支付、PayPal和GC。

- **验证码提供商**

Casdoor支持在用户流程中配置验证码。 Currently, the supported captcha providers include Default Captcha, reCAPTCHA, hCaptcha, Aliyun Captcha and Cloudflare Turnstile.

## 如何配置和使用

### 范围

提供商有不同的范围。 提供者的范围取决于创建者。 只有管理员有权添加和配置提供者。 Casdoor有两种管理员。

- **全局管理员:** 内置 `下的所有用户` 组织和用户启用 `IsGlobalAdmin` 由全局管理员创建的供应商可以被所有应用程序使用。
- **组织管理员:** 用户启用 `Ismin`. 组织管理员创建的提供者只能使用 **组织下的应用程序只能使用**。 (正在开发...)

### 添加应用程序

以下是为了您的应用程序添加提供商的步骤。 在没有添加提供商之前，你还不能在应用中使用。

1. 转到应用程序编辑页面并添加一个新的提供商。

Providers [?](#) :

Name	Category	Type
provider_storage_aliyun_oss	Storage	
provider_casdoor_github	OAuth	
provider_casdoor_google	OAuth	
provider_casdoor_qq	OAuth	
provider_casdoor_wechat	OAuth	
Please select a provider		

2. 选择您想要添加到应用程序的提供商。 这里将显示应用程序可以使用的所有提供商。

Providers [?](#) :

Preview [?](#) :

Name	Category	Type	canSignUp
provider_storage_aliyun_oss	Storage		
provider_casdoor_github	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_google	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_qq	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_wechat	OAuth		<input checked="" type="checkbox"/>
Please select a provider			

- provider\_email\_submail
- provider\_4olfdm
- provider\_casdoor\_bilibili
- provider\_casdoor\_okta
- provider\_casdoor\_alipay
- provider\_casdoor\_slack
- provider\_casdoor\_steam
- provider\_casdoor\_infoflow

Copy

3. 对于 OAuth and Captcha 提供商，您可以配置该用法。 详见 [OAuth and Captcha](#)

Type	canSignUp	canSignIn	canUnlink	prompted	Rule
					Always ▾
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

最后, 保存 配置。 然后您可以尝试在应用程序中使用提供商。

# OAuth

## 概述

将 OAuth 提供商添加到您的应用程序

## 自定义提供商

添加您自己的自定义OAuth 提供商

## Twitter

添加Twitter OAuth 提供商到您的应用程序

## 微博

将 Weibo OAuth 提供商添加到您的应用程序

 **微信**

将Wechat OAuth 提供商添加到您的应用程序

 **企业微信**

将 WeCom OAuth 提供商添加到您的应用程序

 **腾讯 QQ**

添加腾讯QQ OAuth提供商到您的应用程序。

 **钉钉**

添加钉钉 OAuth 到您的应用程序

 **Steam**

将 Steam OAuth 添加到您的应用程序



## GitHub

添加Github OAuth 提供商到您的应用程序



## Gitee

添加Gitee OAuth 提供商到您的应用程序



## 领英 (LinkedIn)

添加Linkedin OAuth 提供商到您的应用程序



## Facebook

将 Facebook OAuth 提供商添加到您的应用程序



## 谷歌

添加Google OAuth 提供商到您的应用程序

 **Google One Tap**

Add Google One Tap support to your application

 **百度**

向您的应用程序添加 Baidu OAuth 提供商

 **AD FS**

添加AD FS 作为第三方服务来完成身份验证

 **AzureAD**

添加 AzureAD 作为第三方服务来完成身份验证

 **Infoflow**

在应用程序中添加 Infoflow OAuth 提供商



Okta

将 Octa OAuth 提供商添加到您的应用程序



Lark

Add Lark OAuth provider to your application

# 概述

Cadoor 可以使用其他 OAuth 应用程序登录。

现在，Casdoor 支持许多 OAuth 应用程序提供者。 提供商的图标将在添加到 Casdoor 后显示在登录和注册页面中。 以下是 Casdoor 支持的提供商：

Provider	Logo	Provider	Logo	Provider	Logo	Provider	Logo
Adfs		Alipay		Amazon		Apple	
Auth0		AzureAD		Baidu		Battle.net	
Bilibili		Bitbucket		Box		Casdoor	
Cloud Foundry		Dailymotion		Deezer		DigitalOcean	
DingTalk		Discord		Douyin		Dropbox	
Eve Online		Facebook		Fitbit		Gitea	
Gitee		GitHub		GitLab		Google	
Heroku		InfluxCloud		Infoflow		Instagram	
Intercom		Kakao		Lark		Lastfm	

Provider	Logo	Provider	Logo	Provider	Logo	Provider	Logo
Line		LinkedIn		Mailru		Meetup	
MicrosoftOnline		Naver		Nextcloud		Okta	
OneDrive		Oura		Patreon		Paypal	
QQ		SalesForce		Shopify		Slack	
SoundCloud		Spotify		Steam		Strava	
Stripe		TikTok		Tumblr		Twitch	
Twitter		Typetalk		Uber		VK	
WeChat		WeCom		Weibo		Wepay	
Xero		Yahoo		Yammer		Yandex	
Zoom		Email		SMS			

我们将向您展示如何申请第三方服务并将其添加到Casdoor。

## 申请成为开发者

在此之前，你需要理解一些概念。

- **RedirectUrl**, 认证后重定向地址, 填写您的应用程序地址, 例如 `https://forum.casbin.com/`
- **Scope**, 用户授予您的权限, 如基本个人资料, 电子邮件地址和帖子及其他。
- **ClientId/AppId, ClientKey/AppSecret**, 这是最重要的信息 而且这是您在申请开发者帐户后需要得到的信息。您 无法与任何人共享 的密钥。

## 添加 OAuth 提供商

1. 导航到您的Casdoor索引页面
2. 点击顶部栏中的 `提供商`
3. 点击 `添加`, 然后您可以在列表顶部看到一个新的提供商
4. 点击新的提供商修改它
5. 选择 `OAuth` 在 `类别` 中
6. 在 `类型` 中选择您需要的 OAuth 提供程序
7. 填写最重要的导入信息, `Client ID` 和 `Client Secret`

## 应用中

1. 单击顶部栏中的 `应用程序` 并选择一个应用程序, 编辑
2. 点击提供商添加按钮, 选择您刚刚添加的提供商
3. 修改提供商的权限, 例如允许注册、登录和取消绑定
4. 完成!

# 自定义提供商

## ① 备注

Casdoor 支持自定义提供商，但自定义提供商必须遵循 3-legged OAuth 的标准流程。和 `Token URL` 和 `Userinfo URL` 的返回值必须遵循 Casdoor 指定的格式。

首先，前往 Casdoor 的供应商页面并创建一个新的供应商。在类型项中选择“自定义”。除了 `Client ID` 和 `Client Secret` 您需要填写 `Auth URL`, `Scope`, `Token URL`, `Userinfo URL` 和 `Favicon`

Type ② :

Custom

Auth URL ②

<https://door.casdoor.com/login/oauth/authorize>

Scope ②

openid profile email

Token URL ②

[https://door.casdoor.com/api/login/oauth/access\\_token](https://door.casdoor.com/api/login/oauth/access_token)

Userinfo URL ②

<https://door.casdoor.com/api/userinfo>

Favicon ② :

URL ② :



Preview:

Client ID ②



Client secret ②



- `Auth URL` 是自定义提供商的 OAuth 登录页面地址。

假定我们填写 <https://door.cassdoor.com/login/oauth/auth/authorization> at `Auth URL`, 然后当用户登

录到此自定义提供商时，浏览器将先跳转到

```
https://door.casdoor.com/login/oauth/
authorize?client_id={ClientID}&redirect_uri=https://{{your-casdoor-
hostname}}/callback&state={State_generated_by_Casdoor}&response_type=code&scope={Scope}`
```

授权完成后，自定义提供商应该重定向到

```
https://{{your-casdoor-hostname}}/callback?code={code}
```

然后，此 URL 中的代码参数将会被 Casdoor 识别。

- **Scope** 是访问 **Auth URL** 时携带的范围参数 它是根据自定义提供商的要求填写的。
- **Token URL** 是获取 accessToken 的 API 地址。

在上一步获取代码后，Casdoor 需要使用此代码获取 accessToken。

假定我们在 **https://door.casdoor.com/api/login/oauth/access\_token** 在 **Token URL** 中填写，然后 Casdoor 将访问 Token URL 如下所示：

```
curl -X POST -u "{ClientID}:{ClientSecret}" --data-binary
"code={code}&grant_type=authorization_code&redirect_uri=https://{{your-casdoor-
hostname}}/callback" https://door.casdoor.com/api/login/oauth/access_token
```

自定义提供商至少应该返回以下内容

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6Ixxxxxxxxxxxxxx",
  "refresh_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6xxxxxxxxxxxxxx",
  "token_type": "Bearer",
  "expires_in": 10080,
  "scope": "openid profile email"
}
```

- **Token URL** 是获取 accessToken 的 API 地址。

假定我们在 **https://door.casdoor.com/api/userinfo** 在 **Token URL** 中填写，然后 Casdoor 将访问 Token URL 如下所示：

```
curl -X GET -H "Authorization: Bearer {accessToken}" https://door.casdoor.com/api/
userinfo
```

自定义提供商至少应该返回以下内容

```
{  
  "name": "admin",  
  "preferred_username": "Admin",  
  "email": "admin@example.com",  
  "picture": "https://casbin.org/img/casbin.svg"  
}
```

- **Favicon** 是自定义提供商的标识URL。

这个标识将与其他第三方登录供应商一起显示在Casdoor的登录页面上。

# Twitter

## Twitter(仍在开发中🚧)

Twitter的应用步骤有点麻烦，官方限制有点严格，因此申请开发者帐户可能比其他第三方平台更困难。

访问[开发者门户](#)，如果您没有帐户，请注册。Twitter需要知道您正在申请哪些开发者帐户。你必须仔细填写它，否则它将不会通过。

申请获批后，创建申请书，填写回调地址和其他信息。您需要做两件事，将被设置为**认证设置**部分。

- 手动打开**3-legged OAuth**，用Twitter登录，代表其他账户发布Tweets等。
- 启用**Request email address from users**请求电子邮件地址以获取用户电子邮件地址。

# 微博

## Weibo ✓

申请微博的开发者账户并不困难，但速度相对较慢。大约需要2-3天。

访问 [开发者网站](#)，填写基本信息，等待审核通过。

审核通过后，您将得到 Client Id 和 Client Secret。

# 微信

## WeChat ✓

访问 [微信开放平台](#) 并注册成为开发者。在你的网站应用或移动应用获得批准后，您将得到您的 App ID 和 App Secret。

Edit Provider [Save](#) [Save & Exit](#)

Name ② :	provider_00bws7
Display name ② :	New Provider - 00bws7
Category ② :	OAuth
Type ② :	WeChat
Client ID ②	
Client secret ②	
Client ID 2 ②	
Client secret 2 ②	
Enable QR code ②	<input checked="" type="checkbox"/>
Provider URL ② :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

[Save](#) [Save & Exit](#)

WeChat 提供商提供两套不同的密钥：

- 第一个密钥对 ([客户端 ID](#),, [客户端密钥](#)) 是 [WeChat 打开平台\(从二维码到二维码\)](#)，它仅适用于PC登录场景。它可以在 PC 中显示二维码，用户可以使用手机扫描代

码。 PC 浏览器允许使用 WeChat 登录。

- 第二个密钥对 (客户端 ID 2, 客户端 secretkey 2) 是 WeChat 媒体平台 (conctionscreen condition), 仅适用于WeChat-app 登录场景。它允许用户使用 WeChat 在 WeChat 移动APP 内登录 它会跳转到您的 WeChat 官方帐户 (微信公众号) 以登录。值得注意的是, 在移动场景中, WeChat 本身不支持在 WeChat APP 以外的登录, 比如在其他移动浏览器 (H5) 或 APP 中。这是对WeChat而不是 Casdoor的限制。

如果您填写了第二个密钥对(客户端 ID 2, 客户端密钥 2并启用 启用 QR 码 开关, 当用户点击WeChat 按钮登录时, casdoor将首先要求用户关注WeChat官方帐户 (从二维码中提取), 然后继续登录。值得注意的是, 这只能在PC登录场景中使用, 因为手机无法自行扫描二维码。在移动场景中使用castor 将自动跳过此步(在WeChat mobile APP中使用WeChat 内置浏览器)。

### 💡 提示

我们建议同时设置两套钥匙。 并链接您的 WeChat 开放平台(即时扫描) 账户和 WeChat 媒体平台(即时连接) 账户一起在 WeChat 开放平台(即时连接) 中 所以通过 PC 登录的 WeChat 用户和手机可以被识别为 Cassdoor 中的同一用户。

:::

### ⓘ 备注

由于 WeChat OAuth 的限制, 目前没有办法通过 WeChat 登录到第三方移动APP 或 WeChat APP以外的移动浏览器。必须在 WeChat 中移动登录。

欲了解更多详情, 请访问 [微信开放平台](#)。

# 企业微信

## 介绍

The WeCom provides the authorized login method of OAuth, which can obtain members' identity information from the webpage opened by the WeCom terminal, eliminating the need for login.

有两种不同的应用程序类型： 内部 应用程序和 第三方 应用程序

## 基本设置

要配置 企业微信 提供商，下面的表格描述了所需参数。

参数描述:

参数	描述
Sub type	内部或第三方
Method	静默或正常模式
Client ID	企业CorpID
Client secret	企业密钥
Agent ID	应用程序Agentid

## ① 信息

企业微信有两种授权方法。 **静默** 授权和 **正常** 授权。

**静默授权:** 在用户点击链接后，页面是 `重定向_URI? code=CODE&state=STATE`

**Normal authorization:** After the user clicks the link, a middle page is displayed for the user to choose whether to authorize or not. 用户确认授权后，转到 `重定向uri?code=CODE&state=STATE`

欲了解更多详情，请参阅 [文档](#)。

## 更多

欲了解更多关于内部应用程序的信息，请参阅 [内部应用程序](#)。

关于第三方应用程序，请参阅 [第三方应用](#)。

# 腾讯 QQ

## Tencent QQ ✓

访问QQ认证平台 - [连接 QQ](#)。

首先你需要应用 [成为开发者](#)。 审核通过后，遵循平台的指示，并获取您的客户ID和客户端密钥。

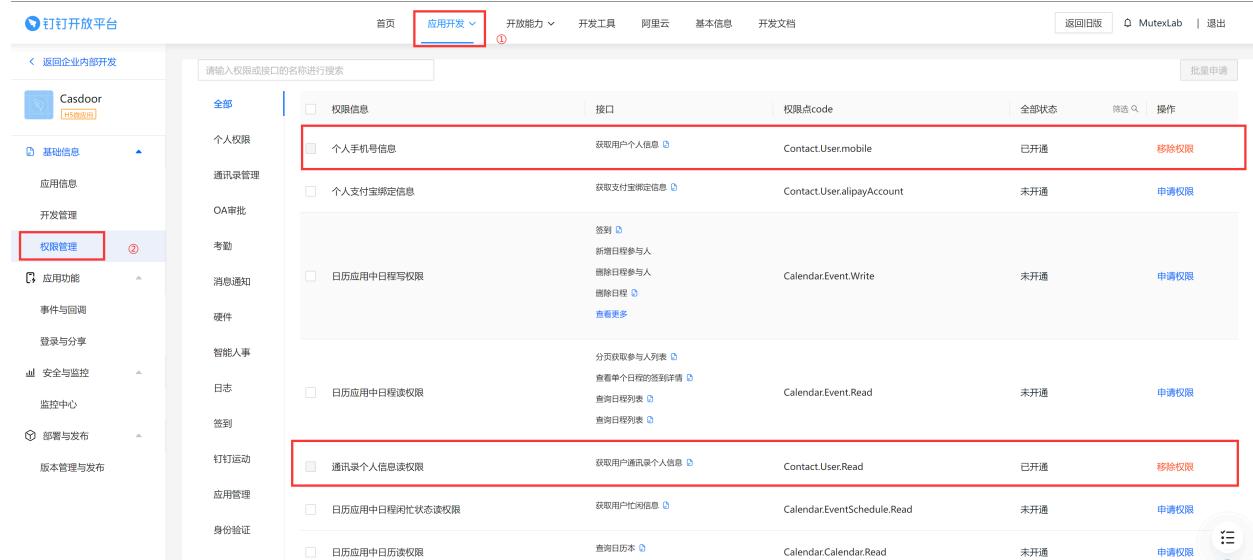
# 钉钉

## DingTalk ✓

访问 [钉钉开放平台](#) 并使用您的钉钉账户登陆，成功登陆后，遵循平台的指导，您将会得到您的 Client Id 和 Client Secret。进入平台后，遵循平台的指示，您将会得到您的客户端ID和客户端的密钥。

欲了解更多信息，请访问 [钉钉开放平台](#)。

此外，您需要将以下权限添加到钉钉中：



The screenshot shows the DingTalk Open Platform's permission management interface. The '应用开发' tab is selected. On the left sidebar, '权限管理' is highlighted with a red box. In the main content area, several permissions are listed under the '个人权限' section. Two specific permissions are highlighted with red boxes:

- 个人手机号信息: 授权用户个人信息, Contact.User.mobile, 已开通, 移除权限
- 通讯录个人信息读权限: 获得用户通讯录个人信息, Contact.User.Read, 已开通, 移除权限

Other visible sections include '基础信息', '应用信息', '开发管理', '考勤', '消息通知', '硬件', '智能人事', '日志', '签到', '钉钉运动', '应用管理', and '身份验证'.

# Steam

## Steam ✓

访问 [Steam WebAPI platform](#) 并通过您的 Steam 账户登陆，之后为您的 Casdoor 域名或ip申请一个 API 密钥，最终将您的 API 密钥作为 Client Secret 填到 Casdoor 中。  
(ClientID 不需要被填写，您的 Steam 账户需要有游戏去申请 API)

欲了解更多详情，请访问 [Steam WebAPI doc](#)。

# GitHub

GitHub OAuth 支持网络应用程序流和设备流量。请继续阅读以获取 OAuth 凭据。

首先，请访问 [GitHub 开发者设置](#) 注册一个新的 GitHub 应用。

## ⚠ 注意事项

**Tricks:** 我们建议您使用 GitHub 应用程序替换 OAuth 应用程序，因为 GitHub 应用程序可以添加多个重定向 URI，可以在部署测试和生产环境时带来方便。[GitHub 官方也建议使用 GitHub 应用程序而不是 OAuth 应用。](#)

## Settings / Developer settings

### GitHub Apps

### OAuth Apps

### Personal access tokens

然后填写 **应用名称**, **主页网址**, **描述** 和 **授权回调URL**.

GitHub App name \*

Casdoor

The name of your GitHub App.

Write

Preview

Markdown supported

A UI-first centralized authentication / Single-Sign-On (SSO) platform supporting OAuth 2.0, OIDC and SAML, integrated with Casbin RBAC and ABAC permission management

Homepage URL \*

http://door.casdoor.com

The full URL to your GitHub App's website.

Add Callback URL

Callback URL

http://localhost:7001/callback

Delete

Callback URL

https://door.casdoor.com/callback

Delete

## ❗ 正确设置授权回调URL

在 Github OAuth 配置中，**应用回调地址** 必须是 **您的 Casdoor 的回调 URL** 并且 Casdoor 中的 **Redirect URL** 应该为 **您的应用回调地址**

更多详情请阅读 [应用配置](#)

注册您的 GitHub 应用程序后，您现在可以生成您的 **客户端密钥**！

## About

Owned by: [REDACTED]

App ID: [REDACTED]

Client ID: lv1 [REDACTED] d2e

[Revoke all user tokens](#)

GitHub Apps can use OAuth credentials to identify users. Learn more about identifying users by reading our [integration developer documentation](#).

## Client secrets

[Generate a new client secret](#)



\*\*\*\*\*dba81954

Added 5 minutes ago by [REDACTED]

[Client secret](#)

[Delete](#)

Last used within the last week



\*\*\*\*\*15822f89

Added on 15 Feb by [REDACTED]

[Client secret](#)

[Delete](#)

添加一个 GitHub OAuth 提供商并填写 [客户端ID](#) and [客户端密钥](#)

Edit Provider

[Save](#) [Save & Exit](#)

Name <small>②</small> :	provider_github_localhost
Display name <small>②</small> :	provider_github_localhost
Category <small>②</small> :	OAuth
Type <small>②</small> :	GitHub
Client ID <small>②</small> :	lv` [REDACTED] .2e
Client secret <small>②</small> :	***
Provider URL <small>②</small> :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

[Save](#)

[Save & Exit](#)

现在您可以使用 GitHub 作为第三方服务来完成认证。

# Gitee

要设置 Gitee OAuth 提供者, 请到 [Gitee 设置](#), 如果您之前没有创建过应用程序, gitee 工作台会是这样:



The screenshot shows the Gitee Workbench interface. At the top, there is a navigation bar with links for '特惠', '企业版', '高校版', '私有云', '博客', and '我的'. On the right side of the top bar are icons for notifications, location, and a user profile. Below the top bar, there is a search bar with the placeholder '搜开源' and several other icons. The main content area has tabs for '我的应用' (selected) and '已授权应用'. Under the '我的应用' tab, it says '无数据' with a magnifying glass icon.

然后您可以创建您的Gitee应用程序。

## 创建第三方应用

应用名称 \*

应用名称

应用描述

应用描述

应用主页 \*

你的应用主页

应用回调地址 \* +

用户授权后, 重定向的地址, 例如: <https://gitee.com/login>

填写 **应用名称**, **应用描述**, **应用主页** 和 **应用回调地址** 并仔细选择 **权限**

### ① 正确设置授权应用回调地址

在 Gitee OAuth 配置中，**应用回调地址** 必须是 **您的 Casdoor 的回调 URL** 并且 Casdoor 中的 **Redirect URL** 应该为 **您的应用回调地址**

更多详情请阅读 [应用配置](#)

然后您可以创建您的 gitee 应用并立即获得 **Client ID** 和 **Client Secrets**！

### Casdoor (今日请求次数: 0 次)

**应用名称 \***

Casdoor

**Client ID**

300ff94d994a7597850bbafb2d5dc67929676dd8e7176b029e067dc6966ef9c4

**Client Secret**

60be2e4e0f3fb8286cfe9f129ab0c3d6b40718a964dade150a8095eb2748730c

[重置 Client Secret](#)

[移除已授权用户的有效 Token](#)

添加一个 Gitee OAuth 提供商并在您的 Casdoor 中填写 **Client ID** 和 **Client Secrets**。

Edit Provider

Save

Name ③ :

my\_gitee\_provider

Display name ③ :

Gitee provider

Category ③ :

OAuth

Type ③ :

Gitee

Client ID ③

300ff94d994a7597850bbafb2d5dc67929676dd8e7176b029e067dc6966ef9c4

Client secret ③

\*\*\*\*\*

现在您可以使用 Gitee 作为第三方服务来完成身份验证！

### ⚠ 注意事项

由于 Casdoor 需要获取用户的电子邮件，必须选中电子邮件选项，否则会导致授权错误。

Permissions (Be careful to select scopes, users might deny authorization when there are too many scopes.)

All

- |   |  |
|---|--|
| <input checked="" type="checkbox"/> user_info | Access and update user data, activities, etc |
| <input type="checkbox"/> projects             | Full control of user projects                |
| <input type="checkbox"/> pull_requests        | Full control of user pull requests           |
| <input type="checkbox"/> issues               | Full control of user issues                  |
| <input type="checkbox"/> notes                | Access, create and edit user comments        |
| <input type="checkbox"/> keys                 | Full control of user public keys             |
| <input type="checkbox"/> hook                 | Full control of user webhook                 |
| <input type="checkbox"/> groups               | Full control of user orgs and teams          |
| <input type="checkbox"/> gists                | Access, create and update user gists         |
| <input type="checkbox"/> enterprises          | Full control of user enterprises and teams   |
| <input checked="" type="checkbox"/> emails    | Access user emails data                      |

Submit

Delete

# 领英 (LinkedIn)

要设置 LinkedIn OAuth 提供商, 请前往 [LinkedIn 开发者](#) 创建一个新的应用。

**in DEVELOPERS** Products Docs and tools ▾ Resources ▾ My apps ▾

## Create an app

\* indicates required

**App name\***

**LinkedIn Page\***  
① This action can't be undone once the app is saved.  
The LinkedIn Company Page you select will be associated with your app. Verification can be done by a Page Admin. Please note this cannot be a member profile page. [Learn more](#)

[+ Create a new LinkedIn Page ↗](#)

**Privacy policy URL**

**App logo\***  
This is the logo displayed to users when they authorize with your app  
 [Upload a logo](#)

填写上面的表单并创建您的应用后, 您需要验证与应用相关联的 LinkedIn 页面



## Identity Cloud Login

Client ID: 860t47n8b4jh7w | Created: Sep 4, 2020

**Settings**

Auth

Products

Analytics

Team members

### App settings

[Delete app](#)

Company:



**Identity Cloud Documentation**

Computer Software; 1-10 employees

[Verify](#)



This app is not verified as being associated with this company.

[Learn more](#)

① 备注

只有公司页面管理员可以验证您的应用，并允许您的应用

在您的应用验证后，您可以继续：

 Identity Cloud Login

Client ID: 860t47n8b4jh7w | Created: Sep 4, 2020

Settings Auth **Products** Analytics Team members

**Products**

Additional available products

 **Marketing Developer Platform**  
Build marketing experiences to reach the right audiences  
[View docs ↗](#) Select

 **Share on LinkedIn**  
Amplify your content by sharing it on LinkedIn  
[View docs ↗](#) Select

 **Sign In with LinkedIn**  
Let users easily sign in with their professional identity  
[View docs ↗](#) Select

为您的应用添加授权重定向URL作为 您的 Casdoor 回调URL。

## Authorized redirect URLs for your app

*No redirect URLs added*

**+ Add redirect URL**

 正确设置授权的重定向 URL

在 Linkedin OAuth 配置中，**应用回调URL** 必须是 **您的Casdoor的回调URL** 并且 Casdoor 中的 **Redirect URL** 应该为 **您的应用回调地址**

更多详情请阅读 [应用配置](#)

然后你就可以获得你的 **Client ID** 和 **Client Secret**

## Application credentials

### Authentication keys

**Client ID:**

860t47n8b4jh7w

**Client Secret:**

.....



添加一个 Linkedin OAuth 提供商并在您的 Casdoor 中填写 **Client ID** 和 **Client Secret**。

Edit Provider

Save

Name [?](#) : my\_linkedin\_provider

Display name [?](#) : Linkedin provider

Category [?](#) : OAuth

Type [?](#) : LinkedIn

Client ID [?](#) : 860t47n8b4jh7w

Client secret [?](#) : \*\*\*\*

现在您可以使用 LinkedIn 作为第三方服务来完成身份验证！

# Facebook

要设置 Facebook OAuth 提供商, 请前往 [Facebook开发人员](#) 创建一个新的应用程序。

选择您要创建的应用类型。

## Select an app type

X

The app type can't be changed after your app is created.



Create or manage business assets like Pages, Events, Groups, Ads, Messenger and Instagram Graph API using the available business permissions, features and products.



### Consumer

Connect consumer products, and permissions, like Facebook Login and Instagram Basic Display to your app.



### Instant Games

Create an HTML5 game hosted on Facebook.



### Gaming

Connect an off-platform game to Facebook Login.



### Workplace

Create enterprise tools for Workplace from Facebook.



### None

Create an app with combinations of consumer and business permissions and products.

[Learn More About App Types](#)

Cancel

Continue

填写姓名并联系电子邮件后，您可以进入 facebook 开发者面板。

FACEBOOK for Developers

Docs Tools Support My Apps Search developer documentation

Casdoor App ID: 1231340483981478 In development

Dashboard Settings Roles Alerts App Review Products Add Product

Add a Product

**Facebook Login**  
The world's number one social login product.  
Read Docs Set Up

**Audience Network**  
Monetize your app and grow revenue with ads from Facebook advertisers.  
Read Docs Set Up

**App Events**  
Understand how people engage with your business across apps, devices, platforms and websites.  
Read Docs Set Up

**Messenger**  
Customize the way you interact with people on Facebook.  
Read Docs Set Up

**Webhooks**  
Subscribe to changes and receive updates in real time.  
Read Docs Set Up

**Instant Games**  
Create a cross-platform HTML 5 game hosted on Facebook.  
Read Docs Set Up

然后设置 Facebook 登录：



## Facebook Login

The world's number one social login product.

Read Docs

Set Up

选择此应用的 Web 平台：

Use the Quickstart to add Facebook Login to your app. To get started, select the platform for this app.



iOS



Android



Web



Other

填写网站URL后，您可以访问 [Facebook Login > 设置](#) 并填写有效的 OAuth Redirect URI

#### Client OAuth Settings

Yes

##### Client OAuth Login

Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking down which token redirect URIs are allowed with the options below. Disable globally if not used. [?]

Yes

##### Web OAuth Login

Enables web-based Client OAuth Login. [?]

Yes

##### Enforce HTTPS

Enforce the use of HTTPS for Redirect URIs and the JavaScript SDK. Strongly recommended. [?]

No

##### Force Web OAuth Reauthentication

When on, prompts people to enter their Facebook password in order to log in on the web. [?]

No

##### Embedded Browser OAuth Login

Enable webview Redirect URIs for Client OAuth Login. [?]

Yes

##### Use Strict Mode for Redirect URIs

Only allow redirects that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [?]

#### Valid OAuth Redirect URIs

A manually specified redirect\_uri used with Login on the web must exactly match one of the URIs listed here. This list is also used by the JavaScript SDK for in-app browsers that suppress popups. [?]

Valid OAuth redirect URIs.

### ❗ 正确设置授权的 REDIRECT URL

在 Facebook OAuth 配置中，有效的 OAuth Redirect URIs 必须是您的 Casdoor 回调 url, and the Redirect URL in Casdoor should be your application callback url

更多详情请阅读 [应用配置](#)

基本应用程序配置即将完成！

在仪表板顶部栏中切换模式从 **In development** 到 **Live**



然后您的 **App ID** 和 **App secrets** 可以在 Casdoor 中使用。



添加 Facebook OAuth 提供商并在您的 Casdoor 填写 **Client ID** 和 **Client Secrets** 使用 **App ID** 和 **App Secrets**。

A screenshot of the Casdoor OAuth provider configuration form for Facebook. The form includes fields for "名称" (Name) set to "my\_facebook\_provider", "显示名称" (Display Name) set to "Facebook provider", "分类" (Category) set to "OAuth", "类型" (Type) set to "Facebook", "Client ID" set to "1231340483981478", and "Client secret" set to "\*\*\*\*\*". A "修改提供商" (Edit Provider) button and a large blue "保存" (Save) button are visible at the top left of the form.

现在你可以使用 Facebook 作为第三方服务来完成身份验证！

# 谷歌

若要设置 Google OAuth 提供商, 请前往 [Google API 控制台](#) 并登录使用您的 Google 帐户。

Project name \*

My Casdoor



Project ID: my-casdoor. It cannot be changed later. [EDIT](#)

Location \*

 No organization

[BROWSE](#)

Parent organization or folder

[CREATE](#)

CANCEL

然后导航到 **OAuth 同意屏幕** 标签来配置 OAuth 同意屏幕。

**API** APIs & Services

## OAuth consent screen

 Dashboard

Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.

 Library Credentials OAuth consent screen Domain verification Page usage agreements

## User Type

 Internal 

Only available to users within your organization. You will not need to submit your app for verification. [Learn more about user type](#)

 External 

Available to any test user with a Google Account. Your app will start in testing mode and will only be available to users you add to the list of test users. Once your app is ready to push to production, you may need to verify your app. [Learn more about user type](#)

**CREATE**

并注册您的 Google 应用。

## Edit app registration

1 OAuth consent screen — 2 Scopes — 3 Test users — 4 Summary

### App information

This shows in the consent screen, and helps end users know who you are and contact you

App name \*

The name of the app asking for consent

User support email \*

For users to contact you with questions about their consent

App logo

BROWSE

Upload an image, not larger than 1MB on the consent screen that will help users recognize your app. Allowed image formats are JPG, PNG, and BMP. Logos should be square and 120px by 120px for the best results.

### App domain

To protect you and your users, Google only allows apps using OAuth to use Authorized Domains. The following information will be shown to your users on the consent screen.

Application home page

然后导航到 Credential 选项卡。

## Credentials

[+ CREATE CREDENTIALS](#) [DELETE](#)

Create credentials to access your enabled APIs. [Learn more](#)

### API Keys

<input type="checkbox"/>	Name	Creation date
No API keys to display		

### OAuth 2.0 Client IDs

<input type="checkbox"/>	Name	Creation date
No OAuth clients to display		

### Service Accounts

<input type="checkbox"/>	Email	Name
No service accounts to display		

并为您的应用创建凭据:

[Create OAuth client ID](#)

A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See [Setting up OAuth 2.0](#) for more information. [Learn more](#) about OAuth client types.

Application type \*



### ① 正确设置授权重定向 URI

在 Google OAuth 配置中，`应用回调URL` 必须是 您的Casdoor的回调URL并且 Casdoor 中的 `Redirect URL` 应该为 您的应用回调地址

更多详情请阅读 [应用配置](#)

创建Client ID并获取 `client ID` 和 `Client Secrets` 后

## OAuth client created

The client ID and secret can always be accessed from Credentials in APIs & Services



OAuth access is restricted to the [test users](#) listed on your [OAuth consent screen](#)

Your Client ID

487708653175-11oih9gfqb2u3tvfp6684qaes5ujjdca.apps.googleusercontent.com



Your Client Secret

HbxoqxxkGSs1lCVRuMTVvK57



DOWNLOAD JSON

OK

添加 Google OAuth 提供程序并在您的 Casdoor 中填写 `client ID` 和 `Client`

Secret

Edit Provider

Save

Name [?](#) : my\_google\_provider

Display name [?](#) : Google provider

Category [?](#) : OAuth

Type [?](#) : Google

Client ID [?](#) : 487708653175-11oih9gfqb2u3tvfp6684qaes5ujjdca.apps.googleusercontent.com

Client secret [?](#) : \*\*\*\*\*

Provider URL [?](#) : <https://console.cloud.google.com/apis/credentials/oauthclient/498643462012-46>

现在你可以使用 Google 作为第三方服务来完成身份验证!

# Google One Tap

## Step1. Configure your application

Casdoor supports login through Google One Tap.

First you need to add Google OAuth Provider to your application according to [Google](#).

Then switch to the application edit page, add the Google OAuth Provider, and switch the Rule from Default to One Tap.

The screenshot shows the Casdoor application edit page. At the top, there is a large text area containing SAML metadata XML. Below this, there is a table titled "Providers" listing various authentication providers. The "Google" provider is selected, indicated by a red box around its row. In the "Rule" column for the Google provider, the "Default" option is selected, also indicated by a red box. A tooltip for the "One Tap" option is visible, pointing to the right of the "Default" button. There are also "Copy SAML metadata URL" and "Copy sign up page URL" buttons at the bottom of the provider list.

## Step2. Login through Google One Tap

Now you can login through Google One Tap.

# 百度

要设置 Baidu OAuth 提供商, 请阅读 [百度的文档](#) 并按其步骤完成 [应用程序创建](#)。

**开发者服务管理**

📍 提示:  
轻应用平台不再支持创建直达号, 如需开通直达号请登录<http://zhida.baidu.com>

**创建工作**

\* 应用名称: CasdoorTest 11/32

传统接入扩展:  合作网站

解决方案:  使用BAE

**创建**

在创建您的应用后, 重定向URL设置在以下位置:

**Casdoor**

**基本信息**

接入类型 —————

其他应用

开发者服务 ————— ✖

Oauth2.0

**安全设置**

**基本信息**

名称: Casdoor 

Icon: 

ID: 25547043

API Key: Hn'...yQmAp61

在以下位置添加您的 Casdoor 域名:

Casdoor

ID API Key Secret Key

基本信息

接入类型

其他应用

开发者服务

Oauth2.0

安全设置

安全设置

Implicit Grant授权方式  启用  禁用

授权回调页:

不配置OAuth授权回调地址，会存在用户授权信息被窃取风险，强烈建议配置该项。授权回调地址的校验规则请参考：[帮助文档](#)

根域名绑定: door.casbin.com

应用服务器IP地址:

应用在访问OpenAPI时须带有Referer信息，且其域名被限制在“根域名绑定”的设置项中

限制访问OpenAPI的Referer

同时绑定访问OpenAPI服务器IP

确定 取消

## ⚠ 注意事项

这部分与百度给出的文档中的实际情况有很大的不同：

1. 将URL添加到回调URL设置中的话很可能验证URL失败，导致登录失败，所以我们要将我们的域名添加到域名设置中。
2. 只能添加一个URL或域名，这与文档有很大不同。

然后您现在就可以获得 `Client ID` 和 `Client Secrets` 了！

# Casdoor

- 基本信息
- 接入类型
- 其他应用
- 开发者服务
- Oauth2.0
- 安全设置

## 基本信息

名称: Casdoor  
Icon:   
ID:   
**Client ID** API Key: HnhK7...QmAp61  
**Client Secret** Secret Key: DTgBZ...ls1bLm1Gha [重置](#)  
创建时间: 2022-01-22 16:20:05  
更新时间: 2022-01-23 15:45:06

添加一个百度 OAuth 提供商并在您的 Casdoor 中填写 `client_id` 和 `client_secrets`。

 Home Organizations Users Roles Permissions Providers Applications Resources Tokens

Edit Provider [Save](#) [Save & Exit](#)

Name: Baidu  
Display name: Baidu  
Category: OAuth  
Type: Baidu  
**Client ID**: HsM...nWT  
**Client secret**: \*\*\*  
Provider URL: <https://github.com/organizations/xxx/settings/applications/1234567>

[Save](#) [Save & Exit](#)

现在你可以使用百度作为第三方服务来完成身份验证!

### ① 一般故障排除

故障排除如果您遇到百度提示您重定向URL不正确， 这里是你可能可以修复错误的一些方式：

1. 将你的域名添加到合适的位置，然后重置Secret(百度重置Secret有bug，会提示错误，但是刷新页面后Secret已经刷新)
2. 如果以上方法都不能解决问题，我们建议您删除应用程序并创建一个新的应用程序，并先设置您的域名。

另一个问题是百度返回的用户名被屏蔽了，不像它的文档显示用户名和显示的名称，所以我们目前只能使用被屏蔽的名称作为用户名。

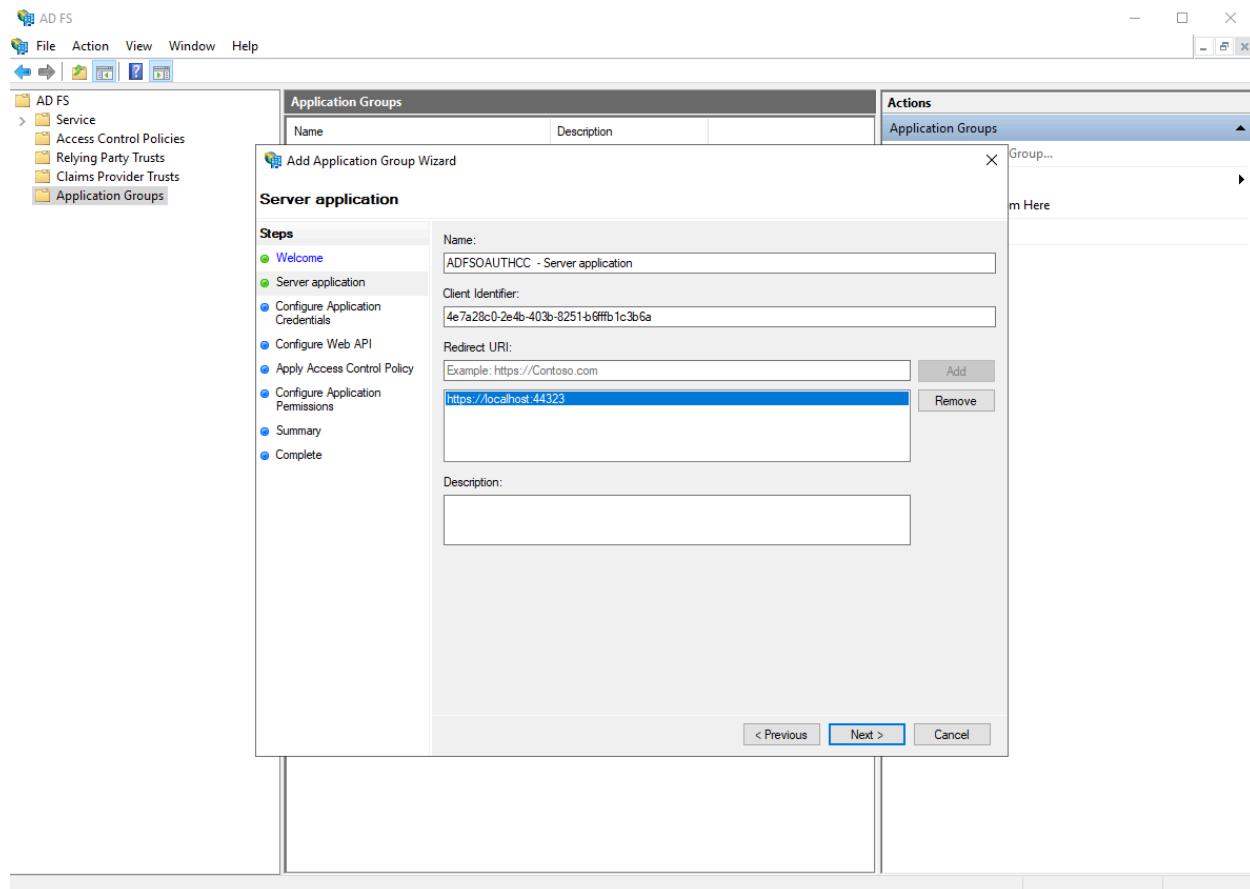
# AD FS

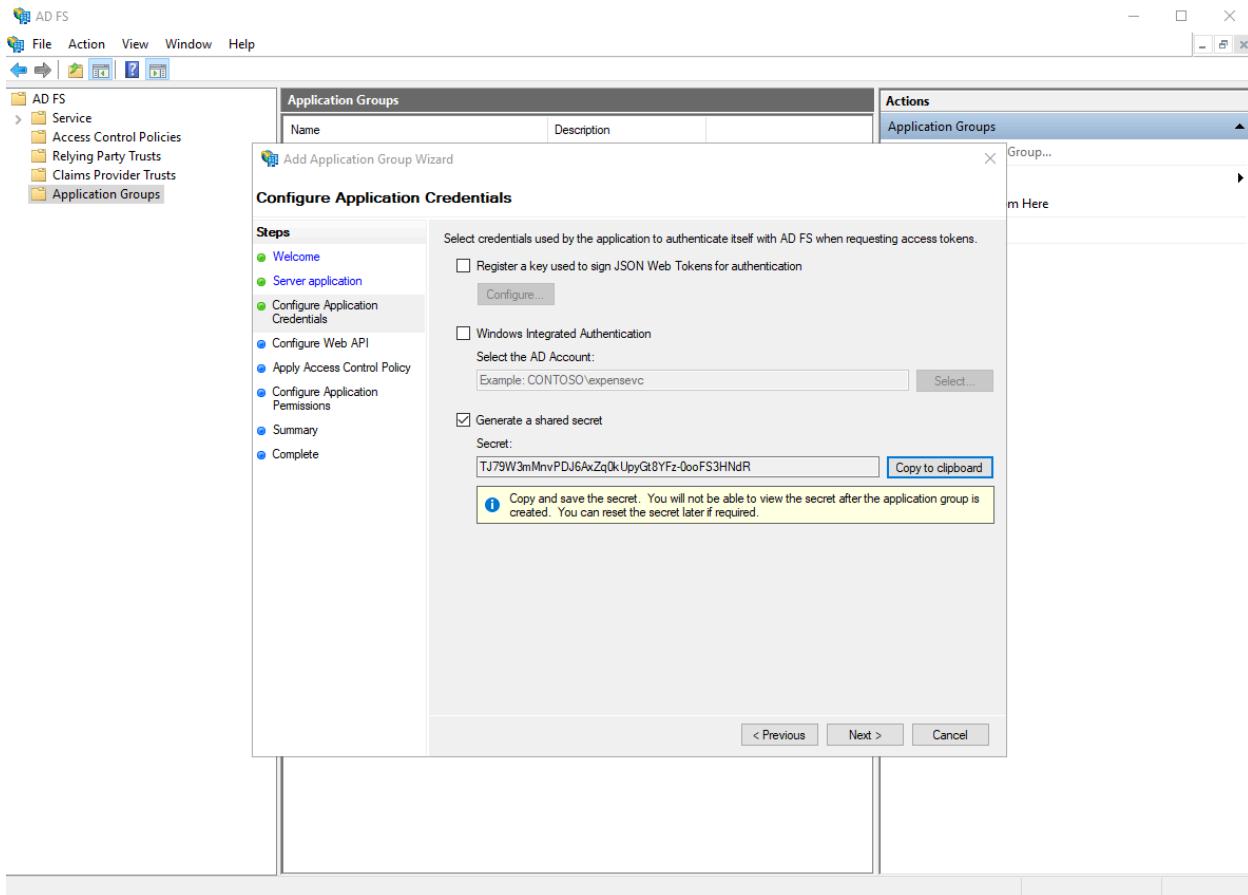
To set up Active Directory Federation Service, please read the [ADFS](#) for the basic knowledge about the ADFS and [AD FS Deployment Guide](#) for how to set up a AD FS server. 确保您有一个完全正常运行的 ADFS 服务器，然后继续下一步。

## 步骤1 通过 ADFS 启用 OAuth

这一步骤详情请参阅 [如何通过 ADFS 启用 OAuth](#)。

当您完成此步骤时，您应该已经获得了clientId 和clientSecret





其中，在Oauth中，第一张图片中的 Client Identifier 和第二张图片中的 Secret 应该是 clientID 和 clientSecret。

## 激活 Casdoor ADFS Provider

添加一个 Gitee OAuth 提供商并在您的 Casdoor 中填写 `client ID` 和 `client Secrets`。

Edit Provider [Save](#) [Save & Exit](#)

Name ②:

Display name ②:

Category ②: OAuth

Type ②: Adfs 

Client ID ②

Client secret ②

Domain ②:

Provider URL ② <https://openhome.alipay.com/platform/appManage.htm#/app/2021003111697088/overview>

[Save](#) [Save & Exit](#)

# AzureAD

## 介绍

Azure Active Directory (Azure AD) 通过为云端和前提条件下的应用提供单一的身份系统，简化了申请管理。可将软件作为服务(SaaS) 应用软件、在前提下的应用软件和业务线路应用程序添加到Azure AD。然后用户可以一次登录来安全和无缝地访问这些应用程序。以及微软公司提供的办公室365项和其他商业应用程序。

## 如何使用？

注册应用程序的步骤如下所示。

### 步骤1. 注册一个应用程序

首先，[注册一个应用程序](#)。然后根据需要选择一个帐户类型。演示站使用下面显示的类型。

Microsoft Azure Search resources, services, and docs (G+)

[Home](#) >

## Register an application

### \* Name

The user-facing display name for this application (this can be changed later).

casdoor



### Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Default only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

[Help me choose...](#)

### Redirect URI (optional)

By proceeding, you agree to the [Microsoft Platform Policies](#)

[Register](#)

## 步骤2. 创建客户端密钥

创建一个 **客户端密钥** 并保存值，这将稍后使用。

casdoor | Certificates & secrets ...

Search (Ctrl+/) Got feedback?

Manage

- Branding & properties
- Authentication
- Certificates & secrets (selected)
- Token configuration
- API permissions
- Expose an API
- App roles
- Owners
- Roles and administrators
- Manifest

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

Certificates (0) Client secrets (1) Federated credentials (0)

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

[New client secret](#)

Description	Expires	Value	Secret ID
casdoor	1/8/2023	3Xr8Q~dFau2Hwyhg6y8Upb53PCFbuF... <span>Copy</span>	f3c7d37c-1def-4e29-b75f-457fa7c081e8 <span>Delete</span>

## 步骤3. 添加重定向 URI

按照图片中的示例添加casdoor的重定向uri。

The screenshot shows the Azure portal's configuration interface for a new application. The left sidebar has 'Authentication' selected. The main area shows platform configurations, supported account types (Azure AD Multitenant), and redirect URLs. A specific redirect URL is highlighted.

## 步骤4. 授予管理员权限

`user.read` API 默认是打开的。 您可以根据您的需要添加更多的范围。 最后，记得 给予管理员权限。

The screenshot shows the Casdoor API permissions page. The left sidebar has sections for Overview, Quickstart, Integration assistant, Manage (with Branding & properties, Authentication, Certificates & secrets, Token configuration, and API permissions selected), Expose an API, App roles, Owners, Roles and administrators, Manifest, Support + Troubleshooting (Troubleshooting and New support request). The main area shows a success message: "Successfully granted admin consent for the requested permissions." A warning message states: "Starting November 9th, 2020 end users will no longer be able to grant consent to newly registered multitenant apps without verified publishers. [Add MPN ID to verify publisher](#)". Below this, a note says: "The 'Admin consent required' column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your org app will be used. [Learn more](#)". The "Configured permissions" section lists Microsoft Graph permissions: email, offline\_access, openid, profile, and User.Read, all marked as "Granted for Default Directory". A red box highlights the "Grant admin consent for Default Directory" button and the "Granted for Default Directory" status for each permission.

## 步骤5. 在casdoor中创建 AzureAD 提供商

最后一步，添加一个AzureAD OAuth提供程序，并在您的Casdoor中填写 **客户端ID** 和 **客户端秘钥**。

Edit Provider

Save

Save & Exit

Name ? : provider\_casdoor\_azuread

Display name ? : Casdoor AzureAD

Category ? : OAuth

Type ? : AzureAD

Client ID ? : 621cc0f0-055f-433f-9894-bfa1bfde169d

Client secret ? : \*\*\*

Provider URL ? : [https://portal.azure.com/#view/Microsoft\\_AAD\\_RegisteredApps/Applications列表](https://portal.azure.com/#view/Microsoft_AAD_RegisteredApps/Applications列表)

Save

Save & Exit

# Infoflow

若要设置 Infoflow OAuth 提供商, 请前往 [Infoflow](#) 并登录使用您的 Infoflow 帐户。

首先, 请访问Infoflow的 [应用中心](#)。



如流 Infoflow 首页 通讯录 应用中心 数据统计 设置

应用(5)

新建应用 应用分组/排序 应用宣传栏

并注册您的 Infoflow 应用。



返回 Casdoor 基本信息 保存 取消

应用logo: 建议使用640\*640, 2M以内的jpg、png图片

应用名称: Casdoor

应用介绍: Casdoor单点登录系统

功能:

应用 (在客户端应用面板中, 为用户提供访问内部系统的入口, [查看客户端示例](#))

机器人 (在企业群聊中, 为用户提供机器人服务, [查看客户端示例](#))

服务号 (以双人会话方式, 为用户提供交流服务, [查看客户端示例](#))

然后你现在可以获取 [AgentID](#)。

< 返回

Casdoor

| 基本信息

应用logo:



应用名称: Casdoor

应用介绍: Casdoor单点登录系统

功能: 应用

AgentID

应用ID: 55

然后导航到 **设置** 标签，并创建一个新的管理组。

如流 Infoway

首页 通讯录 应用中心 数据统计 **设置** ①

基本信息

成员加入

**权限设置** ②

通讯录设置

安全设置

客户端启动页

系统管理组

普通管理组

管理员 ③

**新建下级管理组** ④

暂未设置管理员

通讯录权限

组织架构

将您的结构添加到地址簿权限中，并赋予它以下显示的权限。同时将您刚刚创建的应用程序添加到以下位置。

## 通讯录权限

修改

组织架构

查看

管理 

对部门仅有查看权限时，只可查看被授权的成员资料信息；对部门有管理权限时，可查看成员的所有资料信息

成员ID

姓名

部门

头像

手机号

邮箱

登录帐号

## 应用权限

修改

应用权限

发消息

配置应用

Casdoor



添加敏感接口权限如下所示：

## 敏感接口权限

[修改](#)

接口名称	权限开放
获取部门成员	<input checked="" type="checkbox"/>
获取部门列表	<input type="checkbox"/>
获取成员信息	<input checked="" type="checkbox"/>
获取标签成员	<input type="checkbox"/>
维护通信录	<input type="checkbox"/>
获取成员群组列表	<input type="checkbox"/>
获取群组成员列表	<input type="checkbox"/>
维护群组成员	<input type="checkbox"/>
发送群组消息	<input type="checkbox"/>
维护群组话题	<input type="checkbox"/>
维护勋章	<input type="checkbox"/>
通讯录搜索	<input type="checkbox"/>

您将能够在同一页面看到 `CorpID` and `Secret`

## 开发者凭据

### Client ID

CorpID

hir...1

Secret

HgH...NB

Client Secret

重置

添加 Infoflow OAuth 提供商并填写 **客户端ID**，**客户端密钥** 和 **代理ID** 在您的 Casdoor。

Edit Provider

Save

Save & Exit

Name ② :

Infoflow

Display name ② :

Infoflow

Category ② :

OAuth

Type ② :

Infoflow

Sub type ② :

Internal

Client ID ②

CorpID

Client secret ②

\*\*\*

Secret

Agent ID ② :

55

AgentID

现在您可以使用 Infoflow 作为第三方服务来完成身份验证！

# Okta

要设置 Okta OIDC 提供商, 请首先访问 [Okta Developer](#) 并注册以获得开发人员帐户。

导航到 Applications > Applications 标签, 点击 Create App Integration, 在 Sign-in method 选择 OIDC - OpenID Connect, 和在 Application type 选择 Web Application, 并点击 Next.

### Create a new app integration

**Sign-in method**

[Learn More](#)

- OIDC - OpenID Connect**  
Token-based OAuth 2.0 authentication for Single Sign-On (SSO) through API endpoints. Recommended if you intend to build a custom app integration with the Okta Sign-In Widget.
- SAML 2.0**  
XML-based open standard for SSO. Use if the Identity Provider for your application only supports SAML.
- SWA - Secure Web Authentication**  
Okta-specific SSO method. Use if your application doesn't support OIDC or SAML.
- API Services**  
Interact with Okta APIs using the scoped OAuth 2.0 access tokens for machine-to-machine authentication.

---

**Application type**

What kind of application are you trying to integrate with Okta?

Specifying an application type customizes your experience and provides the best configuration, SDK, and sample recommendations.

- Web Application**  
Server-side applications where authentication and tokens are handled on the server (for example, Go, Java, ASP.NET, Node.js, PHP)
- Single-Page Application**  
Single-page web applications that run in the browser where the client receives tokens (for example, Javascript, Angular, React, Vue)
- Native Application**  
Desktop or mobile applications that run natively on a device and redirect users to a non-HTTP callback (for example, iOS, Android, React Native)

[Cancel](#) [Next](#)

输入 登录重定向 URI , 例如 `https://door.casdoor.com/callback`。

The screenshot shows the 'Sign-in redirect URIs' section of an Okta application configuration. It includes a checkbox for allowing wildcards in the URI, a text input field containing `https://door.casdoor.com/callback`, and a blue 'Add URI' button. There is also a small 'x' icon in the top right corner of the URI input field.

在 分配 部分中, 定义您的应用程序的 控制访问 的类型, 然后点击 保存 以创建应用集成。

现在您获得 `Client ID` , `Client secret` 和 `Okta 域名` 。

The screenshot shows the 'Client Credentials' settings page. It displays the 'Client ID' as `Ooa4we8u8iivyscpb5d7` and the 'Client secret' as a series of dots. Both fields have 'Edit' buttons in the top right corner. Below the client ID, there is a description: 'Public identifier for the client that is required for all OAuth flows.' Below the client secret, there is a description: 'Secret used by the client to exchange an authorization code for a token. This must be kept confidential! Do not include it in apps which cannot keep it secret, such as those running on a client.'

The screenshot shows the 'General Settings' page. It displays the 'Okta domain' as `dev-53555475.okta.com`. The 'Edit' button is located in the top right corner of the domain input field.

在 Casdoor 控制面板中添加 Okta OAuth 提供商，输入您的 `Client ID`，`Client secret` 和 `Domain`

Edit Provider Save Save & Exit

Name ?: provider\_casdoor\_okta

Display name ?: Casdoor Okta

Category ?: OAuth

Type ?: Okta

Client ID ?: 0oa4we8u8iivyscpb5d7

Client secret ?: \*\*\*

Domain ?: <https://dev-53555475.okta.com/oauth2/default>

Provider URL ?: <https://dev-53555475.okta.com>

Save Save & Exit

### ❗ 正确设置域名

请注意，`域名` 不仅是 `Okta 域名`，`/oauth2/default` 应该附加到它。

在授权服务器上访问 [Okta 文档](#) 获取更多详情。

现在你可以使用 Github 作为第三方服务来完成身份验证！

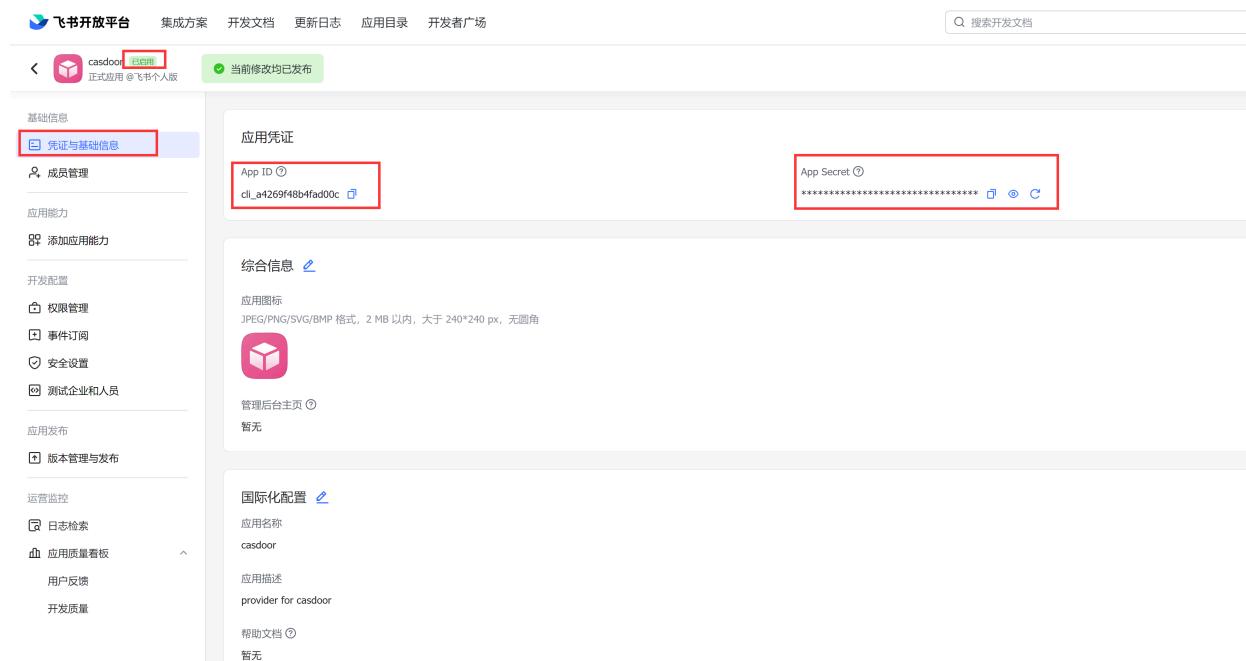
# Lark

## ⓘ 备注

This is an example of how to configure a Lark OAuth provider.

## Step1. Create a Lark application

First, you need to create a new application in [Lark Open Platform](#) and enable it. You can find the [App ID](#) and [App Secret](#) in the basic information of your application.



The screenshot shows the Lark Open Platform application configuration interface. On the left, there's a sidebar with various settings like basic info, member management, capabilities, development configuration, monitoring, and logs. The main area shows the application details for 'casdoor'. The '凭证与基础信息' tab is selected. It displays the '应用凭证' section with the 'App ID' field containing 'cli\_a4269f48b4fad00c' and the 'App Secret' field containing a redacted string. Below this is the '综合信息' section, which includes the application logo (a pink cube icon), the management console URL ('暂无'), and the '国际化配置' section with the application name 'casdoor' and description 'provider for casdoor'. A search bar at the top right says '搜索开发文档'.

Next, add the redirect URL [`<your-casdoor-domain>/callback`](#) (e.g., <http://localhost:7001/callback>) in the security settings of your application.

The screenshot shows the Feishu Open Platform developer console interface. On the left, there is a sidebar with various menu items: 基础信息, 先决与基础信息, 成员管理, 应用能力, 添加应用能力, 开发配置, 权限管理, 事件订阅, 安全设置 (which is highlighted with a red box), 测试企业和人员, 应用发布, 版本管理与发布, 运营监控, 日志检索, 应用质量看板, 用户反馈, and 开发质量. At the top right, there is a search bar labeled '搜索开发文档' and a developer status indicator '当前修改均已发布'. Below the search bar, there are links for '开发者后台' and '回到日报'. The main content area has tabs for '重定向 URL' and 'IP 白名单'. Under '重定向 URL', it says '添加重定向 URL 作为免登录授权码跳转地址。其他重定向 URL 将无法获取免登录授权码。' and has a text input field containing 'http://localhost:7001/callback' with a '添加' (Add) button and a '批量修改' (Batch Modify) button. Under 'IP 白名单', it says '开启 IP 白名单后，仅白名单中的来源请求可以正常调用开放平台 API，不在白名单中的来源请求会被拒绝。' and has a text input field containing '有效的 IP' with a '添加' (Add) button and a '批量修改' (Batch Modify) button. On the far right, there are buttons for '技术支持' (Technical Support), '》' (More), and '收起' (Collapse).

## Step2. Create a Lark OAuth provider

Now create a Lark OAuth provider in Casdoor. Fill the necessary information.

Name	Name in Lark
Category	choose OAuth
Type	choose Lark
Client ID	App ID obtained from Step1
Client secret	App Secret obtained from Step1

The image displays two screenshots side-by-side. On the left, the 'Edit Provider' page in the Casdoor application is shown. It includes fields for Name (lark), Display name (lark-display), Organization (admin (Shared)), Category (OAuth), Type (Lark), Client ID (cli\_a4269f48b4fad00c), Client secret (\*\*\*), and Provider URL (casdoor). Buttons for Save and Save & Exit are at the bottom. On the right, the '飞书开放平台' (Feishu Open Platform) application interface is shown. It shows the '应用凭证' (Application Credentials) section with 'App ID' (cli\_a4269f48b4fad00c) and 'App Secret' (redacted). Red arrows point from the 'Client ID' and 'Client secret' fields in the Casdoor provider configuration to their respective counterparts in the Feishu application.

Now you can use Lark as the third party service to complete authentication.



&gt; 提供商

&gt; Email

# Email



## Overview

Using Email to complete authentication



## MailHog

Using MailHog as the SMTP server

# Overview

## Add an Email provider

1. Click [Add](#) to add a new provider.

2. Select [Email](#) in [Category](#)

Name [?](#) :

Display name [?](#) :

Category [?](#) :

Type [?](#) :

3. Fill [Username](#), [Password](#), [Host](#), [Port](#) of your smtp service.

Username [?](#) :

Password [?](#) :

Host [?](#) :

Port [?](#) :

4. Fill customized **Email Title** and **Email Content** and save.

# MailHog

Here we use MailHog as the SMTP server. [MailHog](#) is an email-testing tool with a fake SMTP server underneath.

## Step1. Deploy the MailHog service

Here the IP address for the MailHog service is `192.168.24.128`, and the SMTP service port is `1025`.

```
[HTTP] Binding to address: 0.0.0.0:8025
2023/07/13 03:06:43 Serving under http://0.0.0.0:8025/
Creating API v1 with WebPath:
Creating API v2 with WebPath:
[APIv1] KEEPALIVE /api/v1/events
[HTTP] Binding to address: 0.0.0.0:8025
Creating API v1 with WebPath:
Creating API v2 with WebPath:
2023/07/13 03:10:36 Using maildir message storage
2023/07/13 03:10:36 Maildir path is /tmp/mailhog641072855
2023/07/13 03:10:36 [SMTP] Binding to address: 0.0.0.0:1025
2023/07/13 03:10:36 Serving under http://0.0.0.0:8025/
[APIv2] GET /api/v2/jim
[APIv2] GET /api/v2/messages
2023/07/13 03:10:50 [HTTP] Connection closed by client: 127.0.0.1:641072855
```

## Step2. Create an email provider

Fill the necessary information and save.

Category <a href="#">?</a> :	Email
Type <a href="#">?</a> :	Default
Username <a href="#">?</a> :	
Password <a href="#">?</a> :	
From address <a href="#">?</a> :	notification@casdoor.com
From name <a href="#">?</a> :	Casdoor Notification
Host <a href="#">?</a> :	192.168.24.128
Port <a href="#">?</a> :	1025
Disable SSL <a href="#">?</a> :	<input checked="" type="checkbox"/>
Email title <a href="#">?</a> :	Casdoor Verification Code (Test)
Email content <a href="#">?</a> :	You have requested a verification code at <a href="#">Casdoor (Test)</a> . Here is your code: <u>%s</u> , please enter in 5 minutes.
Test Email <a href="#">?</a> :	admin@example.com
	<a href="#">Test SMTP Connection</a>
	<a href="#">Send Testing Email</a>
<a href="#">Save</a>	<a href="#">Save &amp; Exit</a>

## Step3. Send the test email

First, click on the [Test SMTP Connection](#) button, if you see [provider: SMTP connected successfully](#), it means that your Casdoor service can access the MailHog service.

Second, click on the [Send Testing Email](#) button, if you see [Email sent successfully](#), it means that the test email has been successfully sent from the [From address](#) to the [Test Email](#).

Name ?: email\_provider provider:SMTP connected successfully

Display name ?: Email Provider Email sent successfully

Organization ?: admin (Shared)

Category ?: Email

Type ?: Default

Username ?:

Password ?:

From address ?: notification@casdoor.com

From name ?: Casdoor Notification

Host ?: 192.168.24.128

Port ?: 1025

Disable SSL ?:

Email title ?: Casdoor Verification Code (Test)

Email content ?: You have requested a verification code at Casdoor (Test). Here is your code: 123456, please enter in 5 minutes.

Test Email ?: admin@example.com **Test SMTP Connection** **Send Testing Email**

Provider URL ?: <https://github.com/organizations/xxx/settings/applications/1234567>

 MailHog

Connected

Inbox (4)

Delete all messages

**Jim**  
Jim is a chaos monkey.  
[Find out more at GitHub.](#)

Enable Jim

**Message Preview:**

From: "Casdoor Notification" <notification@casdoor.com>  
 Subject: Casdoor Verification Code (Test)  
 To: admin@example.com

HTML Plain text Source

You have requested a verification code at Casdoor (Test). Here is your code: 123456, please enter in 5 minutes.

# SMS

## Overview

Using SMS to complete authentication

## Twilio

Using Twilio as a SMS provider for Casdoor

## Alibaba Cloud

Using Alibaba Cloud as a SMS provider for Casdoor

## Amazon SNS

Using Amazon SNS as a SMS provider for Casdoor

 Azure ACS

Using ACS as a SMS provider for Casdoor

# Overview

We use [casdoor/go-sms-sender](#) to send SMS for Casdoor. Now, [go-sms-sender](#) supports Twilio, Submail, SmsBao, Alibaba Cloud, Tencent Cloud, Huawei Cloud and Volc SMS APIs. You can raise an issue, or make a pull request if you want to support other SMS providers.

## Add a SMS provider

1. Click [Add](#) to add a new provider.

2. Select [SMS](#) in [Category](#)



3. Select your provider type

Category [?](#) : SMS

Type [?](#) : Aliyun SMS

Client ID [?](#) : Aliyun SMS  
Huawei Cloud SMS  
SmsBao SMS  
SUBMAIL SMS  
Tencent Cloud SMS

Client secret [?](#) : Twilio SMS  
Volc Engine SMS

Sign Name [?](#) :

Template code [?](#) :

4. Get your information from SMS provider and fill them out.

# Twilio

## Fill the necessary information in Casdoor

There are four required fields. `Client ID`, `Client secret`, `Sender number`, `Template code`. The relationship corresponding to the Tencent Cloud COS account is as follows:

Name	Name in Twilio	is required
Client ID	Account SID	required
Client secret	Auth Token	required
Sender number	Twilio phone number	required
Template code		required

### Twilio information

- Account SID, Auth Token and Twilio phone number

**Step 4: Invite and upgrade**

**Develop Monitor**

**Invite teammates**  
Invite developers to your Twilio account to start building! [Learn more about user access management](#)

**Upgrade your account**  
Upgrade your account to send to any number, buy local numbers, and more. [Learn more about trial account limitations](#)

**Account Info**

- Account SID**: AC06b73d65c8ee67ce8e448edcc64b6ec6
- Auth Token**: ..... [Show](#)
- My Twilio phone number**: +12186751069

**Helpful links**

- [How does Twilio work?](#)
- [SMS Quickstart guides](#)
- [Support help center](#)

## Config Casdoor provider

You can configure the `template code` to suit your requirements, and then enter your phone number in `SMS Test` to test.

Name <a href="#">?</a> :	twilio
Display name <a href="#">?</a> :	twilio
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	SMS
Type <a href="#">?</a> :	Twilio SMS
Client ID <a href="#">?</a> :	AC06b73d65c8ee67ce8e448edcc64b6ec6
Client secret <a href="#">?</a> :	***
Sender number <a href="#">?</a> :	+12186751069
Template code <a href="#">?</a> :	get the message
SMS Test <a href="#">?</a> :	+1 <a href="#">▼</a> Input your phone num... <a href="#">Send Testing SMS</a>
Provider URL <a href="#">?</a> :	<a href="#">🔗</a>



# Alibaba Cloud

## Fill the necessary information in Casdoor

There are four required fields. `Client ID`, `Client secret`, `Sign Name`, `Template code`. The relationship corresponding to the Alibaba Cloud account is as follows:

Name	Name in Alibaba	is required
Client ID	AccessKey ID	required
Client secret	AccessKey Secret	required
Sign Name	Signature	required
Template code	Template code	required

## Alibaba information

- AccessKey ID and AccessKey Secret

After I logged in my Aliyun workbench, click AccessKey to create ID and Secret.

The screenshot shows the Alibaba Cloud SMS service interface. At the top, there's a search bar and navigation links like '费用' (Cost), '工单' (Work Orders), 'ICP 备案' (ICP Registration), '企业' (Enterprise), '支持' (Support), 'App', and '帮助中心'. On the right, there are icons for notifications and user profile. Below the header, a banner says '【有奖调研】阿里云短信服务易用性有奖调研 点击进入'. The main area has tabs for '新手引导' (Newbie Guide), 'OpenAPI 开发者门户' (OpenAPI Developer Portal), '开发者指南' (Developer Guide), and 'AccessKey' (highlighted with a red circle). A sidebar on the left shows '发送量数据' (Delivery Volume Data) and '用户状态/类型: 正常 / 个人用户' (User Status/Type: Normal / Individual User). The central content area includes sections for '快速上手短信服务, 从这里开始!' (Get started with SMS service here!), '国内消息' (Domestic Messages), and '国际/港澳台消息' (International/Hong Kong/Macau/Taiwan Messages). A large button at the bottom says '快速学习短信服务' (Quickly learn about SMS service).

By creating AccessKey, I get my AccessKey ID and AccessKey Secret:

The screenshot shows the Alibaba Cloud security management interface under '安全信息管理' (Security Information Management). It displays a table for '用户 AccessKey' (User AccessKey). The table columns are 'AccessKey ID', 'AccessKey Secret', '状态' (Status), '最后使用时间' (Last Used Time), '创建时间' (Creation Time), and '操作' (Operations). One row is shown: 'LTAI4Fy4mVoMjAzC95rt5Wh7' with '显示' (Visible) status, '启用' (Enabled) status, '2020年7月19日 20:24:58' last used time, '2020年7月11日 17:52:50' creation time, and '禁用' (Disable) and '删除' (Delete) buttons.

- Signature

The screenshot shows the Alibaba Cloud Signature management interface. On the left, there's a sidebar with options like 'Go China', 'Go Globe', 'Analytics', 'Dashboard', 'Delivery Report', 'Messaging Logs', 'Bills', 'Resource Plan Usage', 'Short URL Statistics', 'System Configurations', and 'General Settings'. The main area has a QR code with the text 'Join the group chat to try new features.' and a note about prohibited content. Below that is a section for 'Signatures' with tabs for 'Signatures' (selected), 'Message Templates', and 'Mass Messaging'. A note says 'Generally, signatures are reviewed within 2 hours. Enterprise or institution signatures are reviewed within 2 business days. Recently, a review process took about 1 hour on average. More time may be required due to system upgrades or workload spikes. Business hours: Signatures are reviewed from 9:00 to 21:00 (UTC+8) every day, excluding statutory holidays. Thanks for your cooperation.' A table lists signatures, with one entry for 'casdoor' (status: Approved) highlighted with a red box.

- Template code

**Short Message Service**

- Overview
- Quick Start & Delivery Test
- Go China**
- Go Globe
- Analytics
- Dashboard
- Delivery Report
- Messaging Logs
- Bills
- Resource Plan Usage
- Short URL Statistics
- System Configurations
- General Settings
- Domestic SMS Settings

Create Dedicated DingTalk Group Chat



Scan the QR code to create your dedicated DingTalk group chat.  
You can use the group chat to submit and modify signatures and message templates, and check whether the signatures and message templates are effective.  
Join the group chat to try new features.

Alibaba Cloud SMS prohibits illegal content such as illegal financial marketing, gambling, fraud, obscenity, pornography, and violence in a message. If you send Alibaba Cloud will suspend your service and account according to Short Message Service Terms of Service. If your message is against the law, Alibaba Cloud will not be limited to the personal information authenticated in Alibaba Cloud.

**Message Templates**

Signatures    **Message Templates**    Mass Messaging

1. Generally, signatures are reviewed within 2 hours. Recently, a review process took about **1 hour** on average. More time may be required due to system up hours: Signatures are reviewed from 9:00 to 21:00 (UTC+8) every day, excluding statutory holidays. Thanks for your cooperation.  
2. Message templates provided by Alibaba Cloud SMS do not require submission for approval. However, you are still charged for message delivery.

<b>Create Message Template</b>	Select a template type	Select a review status	Search by message template name	Tag	Template Type	Created At
<input type="checkbox"/> casdoor 测	20020389144	SMS_462155126	Verification Code Message	2023-07-26 17:54:00		

## Config Casdoor provider

Enter your phone number in **SMS Test** to test

Name <small>②</small> :	alibaba
Display name <small>②</small> :	alibaba
Organization <small>②</small> :	admin (Shared)
Category <small>②</small> :	SMS
Type <small>②</small> :	Aliyun SMS
Client ID <small>②</small> :	LTAI5tFwxoA51CnSiQFyyPU5
Client secret <small>②</small> :	***
Sign Name <small>②</small> :	casdoor
Template code <small>②</small> :	SMS_462155126
SMS Test <small>②</small> :	+86 <input type="button" value="▼"/> Input your phone num... <input type="button" value="Send Testing SMS"/>
Provider URL <small>②</small> :	<input type="text" value=""/>

# Amazon SNS

## Obtain the necessary information in Amazon

There are four required fields. Access Key, Secret access key, Region, Template code. I will show you how to obtain these infomations from Amazon SNS.

- Access Key and secret

In Identity and Access Management (IAM), you can create Access Key and Secret access key

The screenshot shows the AWS IAM Access Keys page. On the left, there's a sidebar with options like Dashboard, Access management, and Access reports. The main area has two sections: 'Access keys (1)' and 'CloudFront key pairs (0)'. The 'Access keys' section contains a table with one row:

Access key ID	Created on	Access key last used	Region last used	Service last used	Status
AKIAYOMMXHVZLH4LYKGL	16 days ago	None	N/A	N/A	Active

The 'CloudFront key pairs' section is currently empty, showing 'No CloudFront key pairs'.

- Region

The Region is related to the topic you created

The screenshot shows the AWS Amazon SNS Dashboard. On the left, there's a sidebar with 'Amazon SNS' at the top, followed by 'Dashboard', 'Topics', 'Subscriptions', and a 'Mobile' section with 'Push notifications', 'Text messaging (SMS)', and 'Origination numbers'. Below the sidebar is the main 'Dashboard' area with a title 'Resources for ap-southeast-1'. It displays three metrics: 'Topics' (1), 'Platform applications' (0), and 'Subscriptions' (0). Underneath these metrics is a section titled 'Overview of Amazon SNS' with a sub-section 'Application-to-application (A2A)' which describes Amazon SNS as a managed messaging service for decoupling publishers from subscribers.

## Config Casdoor provider

template code is the message you want to send. Enter your phone number in SMS Test to test.

Name <a href="#">?</a> :	amazon_sns
Display name <a href="#">?</a> :	amazon_sns
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	SMS
Type <a href="#">?</a> :	aws Amazon SNS
Access key <a href="#">?</a> :	AKIAYOMMXHVZACW5RFMX
Secret access key <a href="#">?</a> :	***
Region <a href="#">?</a> :	ap-southeast-1
Template code <a href="#">?</a> :	enter the message you want to send
SMS Test <a href="#">?</a> :	+1 <input type="button" value="▼"/> <input type="text" value="Input your phone num..."/> <input type="button" value="Send Testing SMS"/>
Provider URL <a href="#">?</a> :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

# Azure ACS

## Obtain the necessary information in Azure

There are four required fields. `Client secret`, `Sender number`, `Template code`, `Provider Url`. I will show you how to obtain these infomations from Azure ACS.

- `Client secret`

In Communication Service, you can create User Access Token which is the `client secret` in Casdoor.

The screenshot shows the Azure Communication Service Identity & User Access Tokens page in Casdoor. The left sidebar includes links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Sample applications, Events, Settings, Keys, Push notifications, Properties, Locks, Telephony and SMS, and Phone numbers. The main content area has a search bar, a feedback link, and a Services section with checkboxes for Voice and video calling (VOIP) and Chat, both of which are checked. A 'Generate' button is present. Below this is an 'Identity' section with a text input field containing the value '8.acs:7f19cba5-66d0-4194-b061-1e1a3ac6d68c\_0000001a-6b97-e8f5-2c8a-08482200600c'. A red box highlights the 'User Access token' input field. To the right, a tooltip indicates the token expires at 08/08/23, 12:37:47 AM. Another text input field contains the value 'eyJhbGciOiJSUzI1NlsltmpZC16ljVFODQ4MjE0Qzc3MDczQUU1QzJCREU1Q0NENTQ0ODIERYyQzRDODQjLCj4NXQjOjYb1NDRk1kd2M2N...'. A red box highlights the 'Generate' button.

- Sender number

`Sender number` is the phone number you create in Communication Service

Communication Service

Search (Cmd+/) Get Port Release Give feedback

LOCKS

Tools

- Keys
- Identities & User Access Tokens
- Push notifications

Voice Calling - PSTN

- # Phone numbers
- Direct routing (Preview)

SMS

- Short Codes (Preview)

Monitoring

- Insights (preview)
- Metrics
- Diagnostic settings
- Logs

	Number	Status	Cost (monthly)
<input checked="" type="checkbox"/>	1-833-920-3625	Active	\$2

- Provider Url

Provider Url is the endpoint in Communication Service

Communication Service

Search Move Delete Give feedback

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Sample applications

Events

Settings

Keys

Identities & User Access Tokens

Endpoint : https://casdoor.unitedstates.communication.azure.com

Status : Active

Location : Global

Subscription : 免费试用

Subscription ID : e054e27a-96e8-4cca-a1cf-32717dcd303c

Tags : Add tags

Build engaging communication experiences at scale

Azure Communication Services brings rich communication APIs to all of your apps across any device, on any platform, using the same reliable and secure infrastructure that powers Microsoft Teams.

[Learn more](#)

# Config Casdoor provider

The `template code` is the message you want to send. Enter your phone number in `SMS Test` to test.

---

Name [?](#) :

Display name [?](#) :

Organization [?](#) :

Category [?](#) :

Type [?](#) :  

Client ID [?](#) :

Client secret [?](#) :

Sender number [?](#) :

Template code [?](#) :

SMS Test [?](#) :

Provider URL [?](#) :

---

# 存储

## 概述

设置上传文件的 Casdoor 存储提供商

## Local File System

Using Local File System as a storage provider for Casdoor

## Amazon S3

Using Amazon S3S as a storage provider for Casdoor

## Azure Blob

使用 Azure Blob 作为Casdoor的存储提供商

 **MinIO**

Using MinIO as a storage provider for Casdoor

 **Aliyun OSS**

使用 Azure OSS作为Casdoor的存储提供商

 **Tencent Cloud COS**

Using Tencent Cloud COS as a storage provider for Casdoor

# 概述

如果您需要使用文件存储服务，例如 `头像上传`，您需要设置存储提供商，并在您的 `应用` 中应用它。

Casdoor 支持两种类型的批量操作 - `Local` 或 `Cloud`。在本章中，您将学习如何添加一个存储提供商来使用这些服务。

## 项目

- `Client ID`: A unique identifier provided by the cloud storage provider.
- `Client secret`: A secure value known only to Casdoor and the cloud storage service.
- `Endpoint`: The public URL or domain of the cloud storage service.
- `Endpoint (Intranet)`: The internal or private URL or domain of the cloud storage service.
- `Path prefix`: Path prefix for the file location.

### ① 信息

Default `Path prefix` is `/`. For example, when the `Path prefix` is empty, a default file path:

```
https://cdn.casbin.com/casdoor/avatar.png
```

您可以用 `abcd/xxxx` 填充它，然后您可以将您的头像存储在：

<https://cdn.casbin.com/abcd/xxxx/casdoor/avatar.png>

- Bucket : A container used for storing files and data.
- 域: 您的云存储的 CDN 自定义域名。
- Region ID : An identifier used to specify the data center region where the cloud storage service is located

## 本地设置

We support uploading files to the local system.

## Cloud

We support AWS S3, Azure Blob Storage, MinIO, Alibaba Cloud OSS, Tencent Cloud COS and are adding more Cloud storage services.

# Local File System

## ① 信息

Local File System provider will store your uploaded files in the Casdoor `files` folder

For example, When your Casdoor is located in the `/home/user/casdoor` directory, the uploaded files will be stored in the `/home/user/casdoor/files` folder.

## Config the Casdoor provider

Name <a href="#">?</a> :	localfile
Display name <a href="#">?</a> :	localfile
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	Local File System
Path prefix <a href="#">?</a> :	
Domain <a href="#">?</a> :	http://localhost:8000
Provider URL <a href="#">?</a> :	<a href="#"></a>

**Path prefix** is the prefix of the location path for your files, you can fill it in as needed. In the following example, you can see the difference with or without prefix.

## With prefix

Path prefix [?](#) :

images

casdoor > files > images > resource > built-in > admin > withPrefix.png

## Without prefix

Path prefix [?](#) :

casdoor > files > resource > built-in > admin > withoutPrefix.png

# Amazon S3



This is an example of Amazon S3.

## Create security credentials

Follow the document: [Managing access keys](#), Create and save your `access key` and `secret access key` in amazon console.

## Config your bucket

In your bucket permissons options, uncheck the "block" then save changes.

## Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

### **Block all public access**

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

#### **Block public access to buckets and objects granted through *new* access control lists (ACLs)**

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

#### **Block public access to buckets and objects granted through *any* access control lists (ACLs)**

S3 will ignore all ACLs that grant public access to buckets and objects.

#### **Block public access to buckets and objects granted through *new* public bucket or access point policies**

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

#### **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

[Cancel](#)

[Save changes](#)

Edit the object ownership, check ACLs enabled.

## Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

**ACLs disabled (recommended)**

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

**ACLs enabled**

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

**⚠️** We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

### Object Ownership

**Bucket owner preferred**

If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

**Object writer**

The object writer remains the object owner.

**ⓘ** If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more ↗](#)

Cancel

Save changes

## Config Casdoor

Name	Name in amazon	is required
Client ID	Access key	required
Client secret	Secret access key	required
Endpoint	Endpoint	required
Endpoint (intranet)	VPC endpoint	

Name	Name in amazon	is required
Bucket	Bucket name	required
Path prefix		
Domain	CloudFront domain	
Region ID	AWS region	required

Fill the necessary information, includes the `Client ID` and `Client Secret` obtained from the `access key` and `secret access key` in the previous step. You can refer to this documentation for information on the formatting of the `endpoint`: [Website endpoints](#)

Name <a href="#">?</a> :	awss3
Display name <a href="#">?</a> :	awss3
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	AWS S3
Client ID <a href="#">?</a> :	AKIAYOMMXHVZC5CHBPNR
Client secret <a href="#">?</a> :	***
Endpoint <a href="#">?</a> :	<a href="http://kininaru.s3-website.ap-northeast-1.amazonaws.com">http://kininaru.s3-website.ap-northeast-1.amazonaws.com</a>
Endpoint (Intranet) <a href="#">?</a> :	
Bucket <a href="#">?</a> :	kininaru
Path prefix <a href="#">?</a> :	
Domain <a href="#">?</a> :	
Region ID <a href="#">?</a> :	ap-northeast-1
Provider URL <a href="#">?</a> :	<a href="#">🔗</a>

## (Optional) Use VPC

You can refer to the official documentation for configuration: [Access AWS services through AWS PrivateLink](#)

## (Optional) Use CloudFront distribution

Follow the document to config CloudFront: [config CloudFront](#)

In the domain field, enter your distribution domain name

Endpoint [?](#) : <http://kininaru.s3-website.ap-northeast-1.amazonaws.com>

Bucket [?](#) : kininaru

Path prefix [?](#) :

Domain [?](#) : <https://d20zlk9foisfk0.cloudfront.net>

Region ID [?](#) : ap-northeast-1

Provider URL [?](#) : [🔗](#)

# Azure Blob

① 备注

这是 Azure Blob 的一个示例

- 您必须拥有一个 [Azure storage](#) 帐户。

## 步骤1. 选择 Azure Blob

选择 Azure Blob 作为存储类型。

Edit Provider Save Save & Exit

Name ② :	provider_ftfzes
Display name ② :	New Provider - ftfzes
Category ② :	Storage
Type ② :	Azure Blob
Client ID ② :	Local File System AWS S3
Client secret ② :	Aliyun OSS Tencent Cloud COS
Endpoint ② :	Azure Blob

## 步骤2. 填写 Casdoor 的必要信息

There are four required fields. `Client ID`, `Client secret`, `Endpoint`, `Bucket`.  
Azure Blob账户的对应关系如下：

名称	Azure 中的名称	是否必填项
Client ID	AccountName	必填
Client secret	AccountKey	必填
Endpoint	ContainerUrl	必填
Endpoint (intranet)	PrivateEndpoint	
Bucket	ContainerName	required
Path prefix		
Domain	DomainName	

- 帐号名称

`AccountName` 是您的帐户名称。

- 账户密钥

`AccountKey` 是您访问 Azure API 的密钥。

#### ⓘ 备注

您可以在 Azure Portal 中，在您的存储帐户左手窗格的“访问密钥”部分下获取您的帐户密钥。

The screenshot shows the 'Access keys' section of the Azure Storage account settings for a storage account named 'casbin'. The left sidebar lists various storage management options like Storage browser, Storage Mover, Data storage (Containers, File shares, Queues, Tables), Security + networking (Networking, Azure CDN, Access keys, Shared access signature, Encryption, Microsoft Defender for Cloud), and Data management. The 'Access keys' section is highlighted. At the top, there's a search bar, a 'Set rotation reminder' button, a 'Refresh' button, and a 'Give feedback' link. A note says: 'Access keys authenticate your applications' requests to this storage account. Keep your keys in a secure location like Azure Key Vault, and replace them often with new keys. The two keys allow you to replace one while still using the other.' It also reminds users to update keys with Azure resources and provides a link to 'Learn more about managing storage account access keys'. The 'Storage account name' field contains 'casbin'. Below it, 'key1' is listed with a 'Rotate key' button, showing a redacted 'Key' value and a 'Show' button. A note says 'Last rotated: 2023/7/22 (0 days ago)'. The 'Connection string' for key1 is also shown with a redacted value and a 'Show' button. Below that, 'key2' is listed with a 'Rotate key' button, showing a redacted 'Key' value and a 'Show' button. A note says 'Last rotated: 2023/7/22 (0 days ago)'. The 'Connection string' for key2 is also shown with a redacted value and a 'Show' button.

- ContainerUrl

In your container properties, you can obtain ContainerUrl

 default | Properties

Container

Search Refresh Give feedback

Overview Diagnose and solve problems Access Control (IAM)

Shared access tokens Access policy Properties (selected)

Metadata

**NAME**  
default

**URL**  
`https://casbin.blob.core.windows.net/default`

**LAST MODIFIED**  
7/22/2023, 5:18:03 PM

**ETAG**  
0x8DB8A948D644055

- (Optional) PrivateEndpoint

Azure Private Endpoint is a feature that allows connecting Azure services to Azure Virtual Network (VNet) private subnets. You can refer to the official Azure documentation for configuration: [private endpoint config](#)

- ContainerName

In my example, Create a default container called 'default'.

The screenshot shows the Azure Storage account interface for 'casbin'. The left sidebar has a 'Containers' link highlighted with a red circle. The main area displays a list of containers with two entries: '\$logs' and 'default', where 'default' is also circled in red.

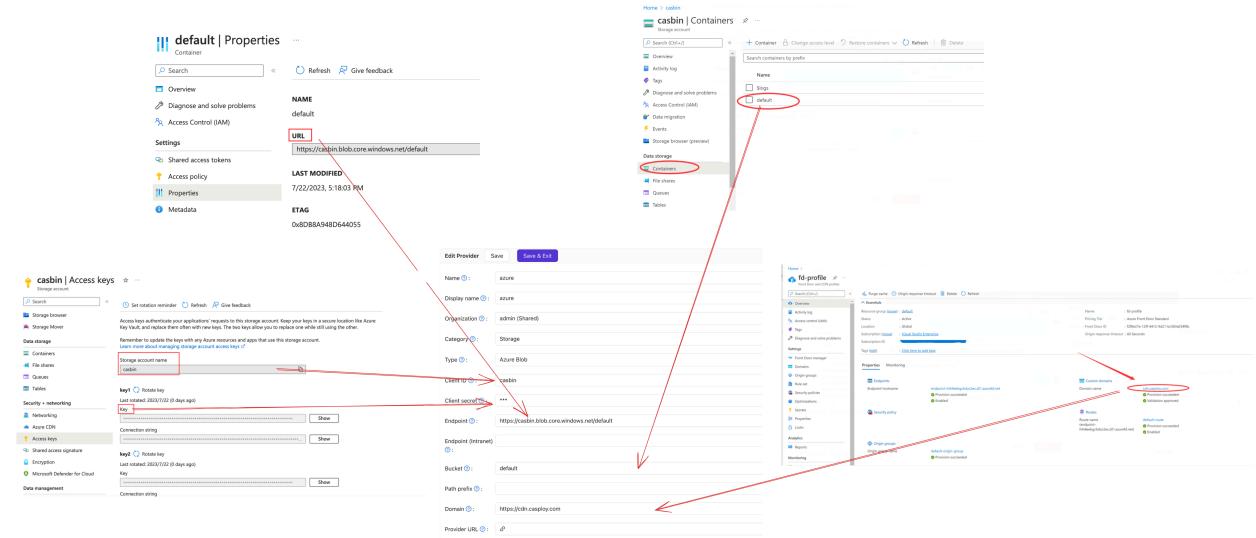
- (Optional) DomainName

The custom domain name in your Azure CDN.

The screenshot shows the Azure Front Door 'fd-profile' blade for 'fd-profile'. In the 'Custom domains' section, the domain 'cdn.casploy.com' is listed with status 'Provision succeeded' and 'Validation approved'. A red arrow points from the 'Custom domains' section in the 'fd-profile' blade to the 'Custom domains' section in the 'fd-profile' blade.

# 步骤3. 保存您的配置

The final result is as follows:



Then you can use Azure Blob Storage services in your application.

# MinIO

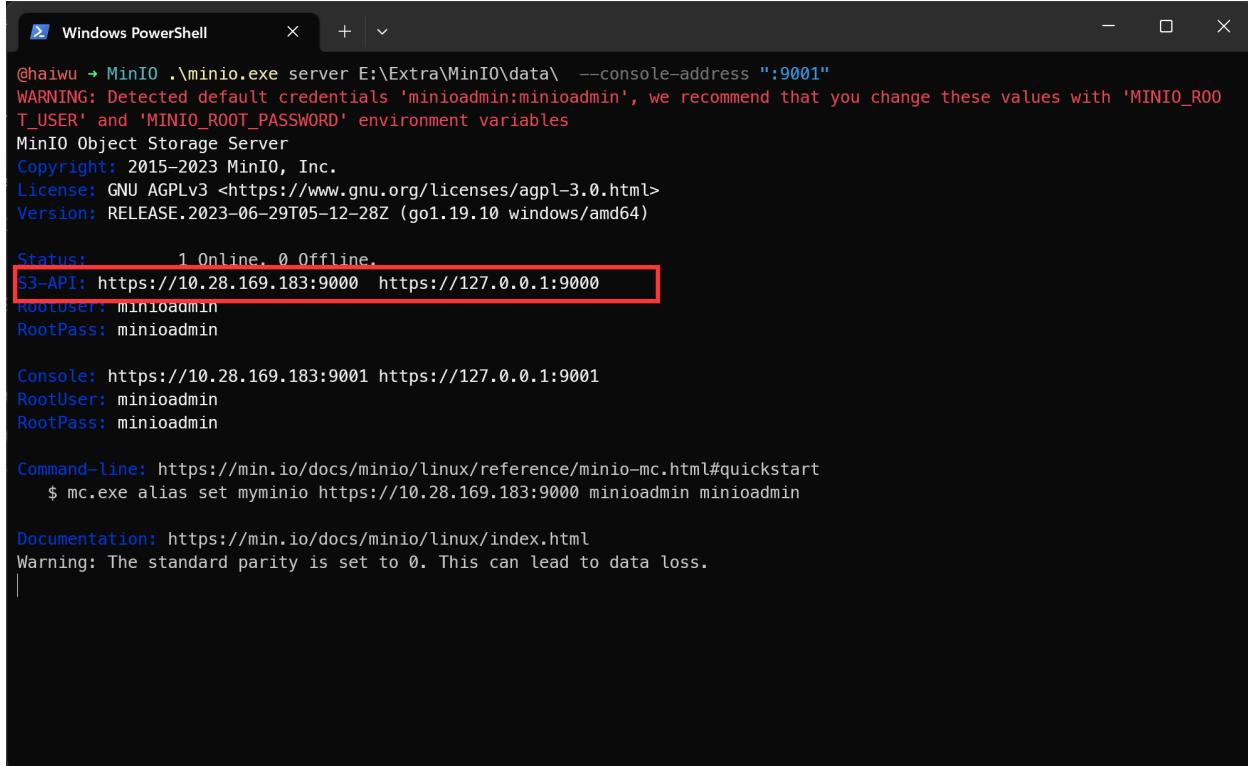
## ⓘ 备注

This is an example of how to configure a MinIO provider.

MinIO is a High Performance Object Storage Service that is API compatible with Amazon S3 cloud storage service.

## Step1. Deploy the MinIO service

First, deploy the MinIO service with TLS enabled. You can get the [API address](#) from console.



```
@haiwu ~ MinIO .\minio.exe server E:\Extra\MinIO\data\ --console-address ":9001"
WARNING: Detected default credentials 'minioadmin:minioadmin', we recommend that you change these values with 'MINIO_ROOT_USER' and 'MINIO_ROOT_PASSWORD' environment variables
MinIO Object Storage Server
Copyright: 2015-2023 MinIO, Inc.
License: GNU AGPLv3 <https://www.gnu.org/licenses/agpl-3.0.html>
Version: RELEASE.2023-06-29T05-12-28Z (go1.19.10 windows/amd64)

Status: 1 Online, 0 Offline.
S3-API: https://10.28.169.183:9000 https://127.0.0.1:9000
RootUser: minioadmin
RootPass: minioadmin

Console: https://10.28.169.183:9001 https://127.0.0.1:9001
RootUser: minioadmin
RootPass: minioadmin

Command-line: https://min.io/docs/minio/linux/reference/minio-mc.html#quickstart
$ mc.exe alias set myminio https://10.28.169.183:9000 minioadmin minioadmin

Documentation: https://min.io/docs/minio/linux/index.html
Warning: The standard parity is set to 0. This can lead to data loss.
```

Second, create the Access Key and Secret key.

The screenshot shows the MinIO Object Store interface. On the left, there's a sidebar with sections like User, Administrator, and Subscription. Under 'User', 'Access Keys' is highlighted with a red box. In the main content area, there's a 'Create Access Key' form. It has fields for 'Access Key' (containing 'ulqg2f67OrI9mmTnphPA') and 'Secret Key' (redacted). A 'Restrict beyond user policy' toggle switch is set to 'OFF'. Below the form are 'Clear' and 'Create' buttons. To the right, there's a 'Learn more about Access Keys' section with links for 'Create Access Keys' and 'Assign Custom Credentials'. The 'Create Access Keys' link is underlined. Below that, there's a note about access keys inheriting policies from parent users and a note about randomized access credentials being recommended. At the bottom, there's a note about access keys supporting programmatic access and a note about assigning access policies.

Third, create the Bucket.

The screenshot shows the MinIO Object Store interface. The sidebar has 'Access Keys' highlighted with a red box. In the main content area, there's a 'Create Bucket' form. It has a 'Bucket Name\*' field containing 'casdoor'. Below it, there's a 'View Bucket Naming Rules' link. Under 'Features', there are three toggle switches: 'Versioning' (ON), 'Object Locking' (ON), and 'Quota' (OFF). Below the form are 'Clear' and 'Create Bucket' buttons. To the right, there's a 'Buckets' section with a note about buckets being similar to folders in a filesystem. It also includes sections for 'Versioning', 'Object Locking', 'Quota', and 'Retention'.

## Step2. Create a MinIO provider in Casdoor

Now create a MinIO provider in Casdoor. Fill the necessary information.

Name	Name in MinIO
Category	choose Storage
Type	choose MinIO
Client ID	Access Key obtained from Step1
Client secret	Secret Key obtained from Step1
Endpoint	API address obtained from Step1
Bucket	Bucket obtained from Step1

### Step3. Use MinIO storage service in your application

Now you can use MinIO storage service in your application.

# Aliyun OSS

① 备注

这是 Azure OSS 的一个示例.

AccessKey 是您访问阿里云 API 的密钥，拥有完全的帐户权限。

所以 在阿里云工作台中创建了 AccessKey

然后创建 OSS 服务：

创建 Bucket

② 创建存储空间 X

注意：Bucket 创建成功后，您所选择的 存储类型、区域、存储冗余类型 不支持变更。

Bucket 名称	mycasdoor	9/63 ✓
地域	华北2 (北京)	▼
相同区域内的产品内网可以互通；订购后不支持更换区域，请谨慎选择。		
Endpoint	oss-cn-beijing.aliyuncs.com	

在 Casdoor 填写必要的信息并保存：

Name <a href="#">?</a> :	provider_storage_aliyun_oss
Display name <a href="#">?</a> :	Storage Aliyun OSS
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	Aliyun OSS
Client ID <a href="#">?</a>	LTAIxFoNpNAnPoiT
Client secret <a href="#">?</a>	***
Endpoint <a href="#">?</a> :	oss-cn-beijing.aliyuncs.com
Endpoint (Intranet) <a href="#">?</a> :	oss-cn-beijing-internal.aliyuncs.com
Bucket <a href="#">?</a> :	casbin
Domain <a href="#">?</a> :	<a href="https://cdn.casbin.com/casdoor/">https://cdn.casbin.com/casdoor/</a>
Provider URL <a href="#">?</a> :	<a href="https://oss.console.aliyun.com/bucket/oss-cn-beijing/casbin/object">https://oss.console.aliyun.com/bucket/oss-cn-beijing/casbin/object</a>

然后您可以在应用程序中使用阿里云云存储服务。

# Tencent Cloud COS

ⓘ 备注

This is an example of Tencent Cloud COS.

## Fill the necessary information in Casdoor

There are five required fields. `Client ID`, `Client secret`, `Endpoint`, `Bucket`, `Region ID`. The relationship corresponding to the Tencent Cloud COS account is as follows:

Name	Name in Tencent	is required
Client ID	SecretId	required
Client secret	SecretKey	required
Endpoint	Endpoint	required
Bucket	BucketName	required
Path prefix		
Domain	CDNDomain	
Region ID	Region	required

## Tencent Cloud cos information

- SecretId and SecretKey

The screenshot shows the Tencent Cloud API Key Management interface. On the left sidebar, under '访问管理' (Access Management), 'API密钥管理' (API Key Management) is selected. The main content area is titled 'API密钥管理' (API Key Management). It contains two sections: '安全提示' (Security Tips) and '使用提示' (Usage Tips). Below these is a table titled '新建密钥' (Create New Key) showing the details of a newly created key:

APPID	密钥	创建时间	最近访问时间	状态
1319606438	<div style="border: 1px solid red; padding: 2px;">SecretId: AKIDdAlMuNrJn8GHI6mLi6NSWbheNr7MViec</div> <div style="border: 1px solid red; padding: 2px;">SecretKey: ***** 显示</div>	2023-07-22 19:01:...	2023-07-22 22:09	已启用

- Endpoint, BucketName and Region

The screenshot shows the Tencent Cloud COS bucket management interface. On the left sidebar, under '存储' (Storage), '桶管理' (Bucket Management) is selected. The main content area shows basic bucket information for 'casdoor-1319606438':

基本信息	域名信息
存储桶名称: casdoor-1319606438 (存储桶不支持改名)	访问域名: https://casdoor-1319606438.cos.ap-guangzhou.myqcloud.com 使用访问域名进行内网访问
所属地域: 广州 (中国) (ap-guangzhou)	自定义CDN加速域名: 0条
创建时间: 2023-07-22 18:57:50	自定义源站域名: 0条
访问权限: 私有读写	全球加速域名: 未开启
	静态网站域名: 未开启

- (Optional) CDNDomain

You can refer to the official documentation for configuration: [config CDN](#)

## Config Casdoor provider

The screenshot shows the Tencent Cloud API Management console. A new secret key is being created under the 'API密钥管理' (API Key Management) section. The 'Secret ID' field contains 'AKIDAMuJn8GHbmLjNSWbheN7Mveic' and the 'Secret Key' field contains 'SecretKey'. Red arrows from the Casdoor provider configuration fields point to these respective fields.

**Tencent Cloud API Management - API密钥管理**

Name: tencentcos  
Display name: tencentcos  
Organization: admin (Shared)  
Category: Storage  
Type: Tencent Cloud COS  
Client ID: AKIDAMuJn8GHbmLjNSWbheN7Mveic  
Client secret: \*\*\*  
Endpoint: casdoor-1319606438.cos.ap-guangzhou.myqcloud.com  
Bucket: casdoor-1319606438  
Path prefix:  
Domain:  
Region ID: ap-guangzhou  
Provider URL: dP

**腾讯云 - casdoor-1319606438**

用量概况 (不作计费计量数据, 供实时数据 (约有2小时延时) 及时参考, 如需计算计费数据, 请在【费用中心】下载并查看用明细。)

**基本信息**

对象数量: 4个  
存储量: 0 B  
创建时间: 2023-07-22 18:57:50  
访问权限: 私有读写

**域名信息**

自定义CDN连接名: 0  
自定义源站域名: 0  
公网IP: 0  
静态加速: 0  
动态加速: 0  
主域名: https://casdoor-1319606438.cos.ap-guangzhou.myqcloud.com 使用访问域名进行公网访问  
注: COS 的访问名做了下限 DNS 解析, 如果你将域名指向内网做了重定向到 COS, 则可能会造成内部回包失败, 因此建议使用公网地址, 路由缓存不支持内网回包, 以从外网访问为好, 有关内外网访问的的相关设置, 请参见[相关链接](#)。



&gt; 提供商

&gt; SAML

# SAML

## 概述

使用来自支持SAML 2.0 的外部身份提供商的身份

## Aliyun IDaaS

使用 Aliyun IDaaS 验证用户

## Keycloak

使用 Keycloak 验证用户

# 概述

可以配置Casdoor来支持用户使用外部身份提供者的身份登录到界面，这些身份提供者支持 SAML 2.0。在这种配置中，Casdoor 不能为用户存储任何登录凭证。

现在，Casdoor 支持许多SAML应用程序提供者。供应商的图标将在添加到Casdoor后显示在登录页面中。以下是 Casdoor 支持的提供商：

阿里云 IDaaS	Keycloak
	
	

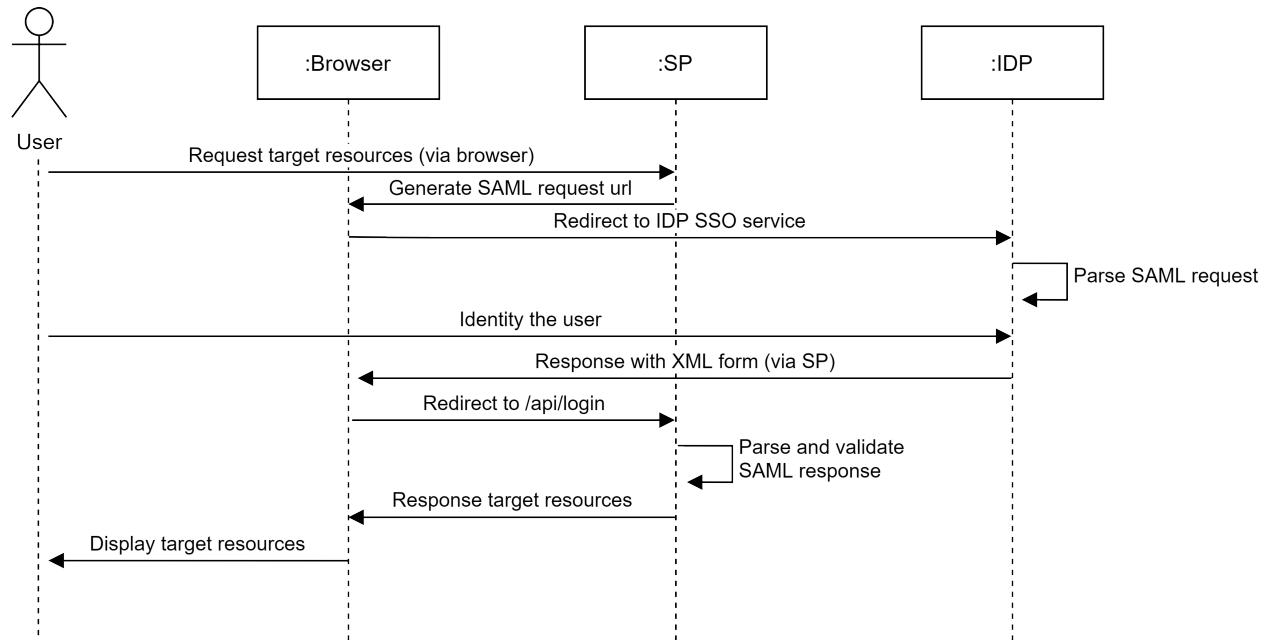
# 条款

- 身份提供商 (IDP) —— 储存身份数据库并向Casdoor提供身份和认证服务的服务。
- 服务提供商(SP) - 为终端用户提供资源的服务，在这种情况下，就是Casdoor 部署。
- 申述消费者服务——身份提供者提出的SAML断言的消费者。

# SAML 集成工作方式

当使用SAML SSO时，用户通过身份提供者登录到Casdoor，而没有向Casdoor传递凭

证。进展情况见下图表。



# Aliyun IDaaS

## 在 Aliyun IDaaS 中创建 SAML 应用程序

登录到 [Aliyun 管理控制台](#), 搜索并前往应用程序标识服务 (IDaaS)。

The screenshot shows the Aliyun Management Console with the following details:

- 左侧导航栏:** 应用身份管理 > 概览页
- 中间主区域:**
  - IDaaS (Identity-as-a-Service) 是为企业客户提供的身份访问管理服务, IDaaS支持 EIAM 和 CIAM**
  - EIAM 不同版本区别:** 比较了标准版和专属版的功能差异。
- 右侧工具栏:** 阿里云售后 (包含提交工单、二维码)
- 右侧帮助文档:** 包含 EIAM 相关的常见问题和 API 文档链接。

点击 EIAM 实例列表 并打开免费版本。

The screenshot shows the Aliyun Management Console with the following details:

- 左侧导航栏:** 应用身份管理 > EIAM 实例列表
- 中间主区域:** EIAM 实例列表, 显示没有相关实例。
- 右侧工具栏:** 开通免费版 (带有红色边框)

打开后将自动创建并运行一个实例。点击实例名称或 管理 按钮来输入 IDaaS 管理控制台。

The screenshot shows the EIAM Instance List page. On the left, there's a sidebar with '应用身份管理' (Application Identity Management) and 'EIAM 实例列表' (EIAM Instance List). The main area displays a table with one instance: 'idaas-cn-shanghai' (Status: Running, Free Tier, 100 users, V1.9.6-GA). A red box highlights the '管理' (Manage) button next to the instance name. Below the table are navigation buttons: '< 上一页' (Previous Page), '1' (Page 1), and '下一页 >' (Next Page).

输入了 IDaaS 管理控制台后，点击 添加应用程序，搜索 SAML，然后点击 添加应用程序

The screenshot shows the 'Add Application' page under '应用' (Application) in the sidebar. The '添加应用' (Add Application) button is highlighted with a red box. In the search bar, 'SAML' is typed. The results table lists several SAML-related applications: '云安全访问服务SASE' (Cloud Access Service Edge), '阿里云RAM-用户SSO', '阿里云RAM-角色SSO', '阿里邮箱', 'WordPressSaml', 'SAML', and 'GitLab'. The 'SAML' application row has a red box around its '操作' (Operation) column, which contains a blue '添加应用' (Add Application) button.

点击 添加签名密钥。

## 添加应用 (SAML)

×

导入SigningKey	添加SigningKey				
别名	序列号	有效期	秘钥算法	算法长度	操作
暂无数据					

填写所有必需的信息并提交。

## 添加SigningKey

×

* 名称	CASDOOR-TEST
部门名称	请输入部门名称
公司名称	请输入公司名称
* 国家	CN
* 省份	Beijing
城市	请输入城市
* 证书长度	1024
* 有效期	3 年
<button>提交</button>	<button>取消</button>

选择添加的签名密钥。

## 添加应用 (SAML)

×

		导入SigningKey	添加SigningKey			
别名	序列号	有效期	秘钥算法	算法长度	操作	
CN=CASDOOR-TEST, ST=Beijing, C=CN	3322747020095790430	1095	RSA	1024	<button>选择</button> <button>导出</button>	

填写下面所需的所有信息并保存。

- IDP 身份ID：保持与签发者地址在 Casdoor 中相同。
- SP 实体 ID & SP ACS URL (SSO 定位)：现在填写您想要的任何东西。完成 Casdoor 配置后，您需要修改。
- 描述属性：直接填充为用户名。
- 账户关联模式：账户协会

## 添加应用 (SAML)

X

图片大小不超过1MB

应用ID

idaas-cn-shanghai-pvl0hq0ojppugin\_saml

\* 应用名称

CASDOOR-SAML

\* IDP IdentityId

CASDOOR

IDP IdentityId is required

\* SP Entity ID

http://localhost

SP Entity ID is required

\* SP ACS URL(SSO Location)

http://localhost

\* NameIdFormat

urn:oasis:names:tc:SAML:2.0:nameid-format:transient

v

\* Binding

POST

v

SP 登出地址

请输入 SP 登出地址

Assertion Attribute

username

应用子账户

v

-

+

断言属性。设值后，会将值放入SAML断言中。名称为自定义名称，值为账户的属性值。

Sign Assertion



IDaaS发起登录地址

IDaaS发起登录地址

以 http://、https:// 开头，填写后使用 IDaaS 发起登录将会跳转到该地址，而不会使用 SAML 的idp发起登录流程

\* 账户关联方式

账户关联 (系统按主子账户对应关系进行手动关联，用户添加后需要管理员审批)

账户映射 (系统自动将主账户名称或指定的字段映射为应用的子账户)

提交

取消

## 帐户授权 & 关联

在SAML应用成功添加后，授权提示会高亮。现在不要授权它，添加一个帐户，然后授权它。

转到组织和组，然后点击新帐户。

The screenshot shows the Alibaba Cloud Organization Structure Management interface. On the left sidebar, under the '账户' (Account) category, the '机构及组' (Organizational Units and Groups) option is selected and highlighted with a red box. In the main content area, there is a large callout box titled '机构及组' (Organizational Units and Groups) with the following text:  
管理员在当前页面对组织架构、部门及其包含的组、账户进行管理，也可以使用AD、LDAP或Excel文件的方式配置导入或同步。  
在左侧的组织架构树中，可以右键点击某个部门对其进行操作，也可以左键选择某个部门，并在右侧为其进行创建账户、创建组、创建部门等操作。

The main panel is titled '组织架构' (Organizational Structure) and contains a search bar with '新账账户' (Create New Account) highlighted by a red box. Below the search bar is a table with one row of data:

编号	账户名称	显示名称	类型	目录	操作
1	idaas_manager	默认管理员	自建账户	/	<a href="#">修改</a> <a href="#">账户同步</a> <a href="#">同步记录</a>

At the bottom right of the table, there is a pagination bar showing '共1条' (1 item), page number '1', and a '10条/页' (10 items per page) dropdown.

填写所有必需的信息并提交。

## 新建账户

X

### 账户属性

### 扩展属性

### 父级组

父级

阿里云IDaaS

\* 账户名称

casdoor

账户名称不能以特殊字符开始，可包含大写字母、小写字母、数字、中划线(-)、下划线(\_)、点(.)，长度至少 4 位

\* 显示名称

casdoor

\* 密码

\*\*\*\*\*

密码中必须包含大小写字母+数字+特殊字符的组合;长度至少 10 位，密码不能包含"<"和">"。

邮箱

请输入有效的邮箱地址

手机号或邮箱至少填写一个。

手机号

+86 ▾ 151 123456789

手机号或邮箱至少填写一个。

外部ID

外部ID

IDaaS 平台中的唯一身份标识, 若不填将由系统自动生成。

过期时间

过期时间

不填将使用系统默认过期时间 2116-12-31

备注

备注

用户备注信息

提交

取消

转到 **应用程序授权**, 选择您想要授权的帐户, 然后点击 **保存**。

The screenshot shows the 'Application Authorization' section of the Alibaba Cloud console. On the left sidebar, under the '授权' (Authorization) category, '应用授权' (Application Authorization) is selected and highlighted with a red box. In the main content area, there's a search bar and several filter options at the top. Below that is a table with columns: 账户 (Account), 显示名称 (Display Name), and 邮箱 (Email). Two rows are listed: 'casdoor' (display name casdoor, email casdoor) and 'idaas\_manager' (display name idaas\_manager, email manager@idaas.com). A blue '保存' (Save) button is located at the bottom of the table. The entire screenshot is framed by a light gray border.

前往 应用程序列表, 点击 查看应用子账户, 然后点击 添加帐户关联。

The screenshot shows the 'Application List' section of the Alibaba Cloud console. On the left sidebar, '应用列表' (Application List) is selected and highlighted with a red box. In the main content area, there's a table with columns: 应用图标 (Application Icon), 应用名称 (Application Name), 应用ID (Application ID), 设备类型 (Device Type), 应用状态 (Application Status), 二次认证状态 (Two-factor Authentication Status), and 操作 (Operations). One row is listed: 'CASDOOR-SAML' (icon S, application ID idaas-cn-shanghai-pv0lhq0ojppugin\_saml, Web应用, status green, two-factor auth off). To the right of the table, there are buttons for '授权' (Authorization) and '详情' (Details). The entire screenshot is framed by a light gray border.

The screenshot shows the Alibaba Cloud IDaaS application management interface. On the left, there is a sidebar with various navigation options like '概览', '快速入门', '应用' (selected), '账户', '认证', '授权', etc. The main content area has a title '应用列表 / 子账户' and a sub-section '← 子账户'. A modal window titled '子账户' is open, explaining what a子账户 is and how it relates to the main account. At the top right of the main page, there are buttons for '添加账户关联' (highlighted with a red box), '批量导入', and '批量导出'. Below the modal, there is a table titled 'CASDOOR-SAML' with columns: 账户名称, 显示名称, 子账户, 子账户密码, 是否关联, 审批状态, 关联时间, 操作. The table shows '暂无数据'. At the bottom right of the page, there are pagination controls.

填写需要关联的主账户和子账户，然后点击 **保存**。

主账户存在于IDaaS中，子账户是Cassdoor用户的ID。

The screenshot shows the 'Add Account Association' dialog box. It has two input fields: one for the primary account labeled '主账户' containing 'casdoor', and another for the child account labeled '子账户' containing '52908237-fa4c-4681-b636-a6afce22fb2e'. At the bottom, there are two buttons: a blue '保存' button and a white '返回' button.

## 导出 IDaaS 元数据

转到 **应用程序列表**, 点击 **查看应用程序详细信息** 然后点击 **导出 IDaaS SAML Metadada**

The screenshot shows the Aliyun Idaas application management interface. On the left, there is a sidebar with various navigation options like '概览', '快速入门', '应用' (with '应用列表' selected), '账户', '认证', '审计', '其它管理', and '设置'. The main area has tabs for '应用列表' and '添加应用'. Under '应用列表', there is a table with columns '应用图标', '应用名称', and '应用ID'. One row shows 'CASDOOR-SAML', 'CASDOOR-SAML', and 'idaas-cn-shanghai...saml'. Below this table are two sections: '应用信息' and '认证信息'. The '应用信息' section contains '应用的详细信息', '查看详情', '修改应用', and '删除应用'. The '认证信息' section contains '应用的单点登录地址', 'IDaaS发起地址'. To the right, there is a detailed view for the 'CASDOOR-SAML' application. It includes sections for '图标' (SAML logo), '应用ID' (idaas-cn-shanghai...jin\_saml), '应用名称' (CASDOOR-SAML), '应用Uuid' (d9bd59093c5c031b7abbb0efa849f7bRxS506SQ3y7), 'SigningKey' (3322747020095790430(CN=CASDOOR-TEST)), 'NameIdFormat' (urn:oasis:names:tc:SAML:2.0:nameid-format:transient), 'SP ACS URL' (http://localhost), 'IDP IdentityId' (CASDOOR), 'SP Entity ID' (http://localhost), 'Binding' (POST), 'Sign Assertion' (是), 'Assertion Attribute' (username:APPLICATIONUSERNAME), 'IDaaS发起登录地址' (https://dvmoykbkx.login.aliyunidaas.com/enduser/api/application/plugin\_saml/idaas-cn-shanghai-login\_saml/sp\_sso?SAMLRequest=jxx&RelayState=yyy), and 'SP发起地址' (https://dvmoykbkx.login.aliyunidaas.com/enduser/api/application/plugin\_saml/idaas-cn-shanghai-login\_saml/sp\_sso?SAMLRequest=jxx&RelayState=yyy). A red box highlights the 'IDP IdentityId' field.

# 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

选择分类为 **SAML**, 输入 **Aliyun Idaas**. 复制元数据内容并粘贴到 **元数据** 输入。 **端点**, 的值, **IdP** 和 **发行商URL** 将在点击 **分析** 按钮后自动生成。

Name ⓘ: casdoor-idaas

Display name ⓘ: casdoor-idaas

Category ⓘ: SAML

Type ⓘ: Aliyun IDaaS

Client ID ⓘ:

Client secret ⓘ:

Metadata ⓘ:

```
<md:IDPSSODescriptor>
<md:EntityDescriptor>
```

**Parse**

Endpoint ⓘ: https://dvmoykbkx.login.aliyundaas.com/enduser/api/application/plugin\_saml/idaas-cn-shanghai...\_saml/sp\_sso

IdP ⓘ: MIIIBzCCAVCgAwIBAgILhzEz2NMHV4wDQYJKoZIhvCNAAQEFBQAwnjELMAkGA1UEBhMCQ04xEAOBgNVBAgTB0JlaWppbmcsFTATBgNVBAMTDENBU0RPT1ItVEVTDAeFw0yMTExMDkwNzEyMTEyMDgwNzEyMTEyMDgwNzEyMTEyMDYxCzAJE

Issuer URL ⓘ: CASDOOR

SP ACS URL ⓘ: http://localhost:8000/api/acs

SP Entity ID ⓘ: http://localhost:8000/api/acs

Provider URL ⓘ: https://github.com/organizations/xxx/settings/applications/1234567

**Save**

复制 SP ACS URL 和 SP 实体 ID 并点击 保存 按钮

编辑您想要在 Cassdoor 中配置的应用程序。选择刚刚添加的提供者，然后点击按钮 保存。

Providers ⓘ:

Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
casdoor-idaas	SAML						

Preview ⓘ: **Test signup page...** **Test signin page...**

## 修改 Aliyun IDaaS 的 SAML 应用程序

禁用应用程序，然后点击 修改应用程序。

The screenshot shows the Alibaba Cloud Application Management interface. On the left, there's a sidebar with categories like Application, Account, Authentication, Authorization, Audit, and Settings. The main area is titled '应用列表' (Application List) and shows a table with columns: 应用图标 (Application Icon), 应用名称 (Application Name), 应用ID (Application ID), 设备类型 (Device Type), 应用状态 (Application Status), 二次认证状态 (Two-factor authentication status), and 操作 (Operations). A red box highlights the '操作' column for the first row, which has an icon with a red border. Below the table, there are tabs for '应用信息' (Application Information), '认证信息' (Authentication Information), '账户信息 - 同步' (Account Information - Sync), and '账户信息 - 子账户' (Account Information - Sub-account). The '应用信息' tab is active, showing fields like '应用的详细信息' (Detailed information about the application), '应用的单点登录地址' (Single sign-on address), and 'IDaaS发起地址' (IDaaS initiation address). The '账户信息 - 同步' tab shows 'SCIM协议设置以及把组织机构、组同步推送至应用' (SCIM protocol settings and pushing organization structures and groups to the application) and '同步机构' (Sync organization) and 'SCIM配置' (SCIM configuration). The bottom right of the page includes pagination controls: '共 1 条' (1 page), a page number input field with '1', and a '10条/页' (10 items per page) dropdown.

填写 SP 实体 ID and SP ACS URL (SSO location) 将内容复制到Casdoor。 提交并启用应用程序。

## 修改应用 (CASDOOR-SAML)

X

图标



上传文件

图片大小不超过1MB

应用ID

idaas-cn-shanghai-...\\login\_saml

\* 应用名称

CASDOOR-SAML

\* IDP IdentityId

CASDOOR

IDP IdentityId is required

\* SP Entity ID

http://localhost:8000/api/acs

SP Entity ID is required

\* SP ACS URL(SSO Location)

http://localhost:8000/api/acs

\* NameIdFormat

urn:oasis:names:tc:SAML:2.0:nameid-format:transient

\* Binding

POST

SP 登出地址

请输入SP 登出地址

Assertion Attribute

username

应用子账户



断言属性。设值后，会将值放入SAML断言中。名称为自定义名称，值为账户的属性值。

Sign Assertion



IDaaS发起登录地址

IDaaS发起登录地址

以 http://、https://开头，填写后使用 IDaaS 发起登录将会跳转到该地址，而不会使用 SAML 的idp发起登录流程

## 验证效果

转到您刚刚配置的应用程序，您可以在登录页面找到一个图标。

点击图标并跳转到 Aliyun IDaaS 登录页面，然后在验证后成功登录到Casdoor。



---

username, Email or phone

Password

Auto sign in      [Forgot password?](#)

[Sign In](#)

[Sign in with code](#)    No account? [sign up now](#)

A row of social media icons for GitHub, LinkedIn, and others.

---

# Keycloak

JBoss KeyCloak 系统是一个广泛使用的开源身份管理系统，它支持通过 SAML 和 OpenID 连接与应用程序集成。它还可以作为其他提供商，例如LDAP或其他SAML提供商和支持SAML或OpenID连接的应用程序之间的身份经纪人运行。

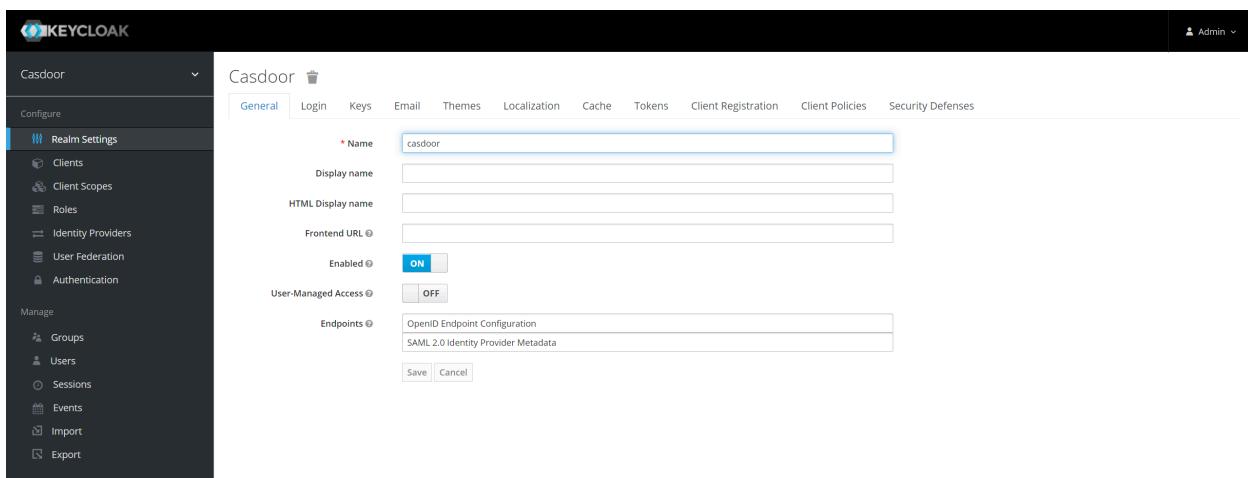
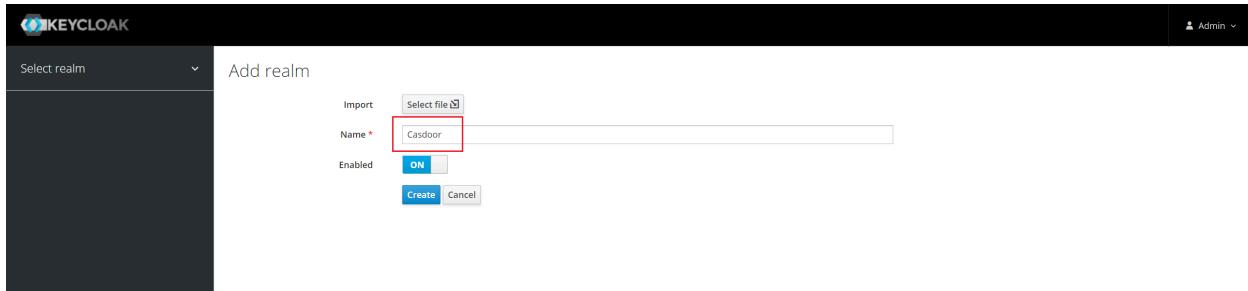
下面是如何在 Keycloak 中配置新客户端条目，并配置Castoor 来允许Keycloak 用户登录UI，这些用户通过Keycloak 配置被授予访问权限。

## Configure Keycloak

一些配置选项和假设，特别是这个示例：

- 我们假设您正在以本地开发模式运行 Casdoor。 Casdoor UI is available at: <http://localhost:7001> and server is available at <http://localhost:8000>. 必要时用适当的URL替换。
- 让我们假设你正在本地运行Keycloak。 Keycloak UI 可在以下网址查阅：<http://localhost:8080/auth>。
- 在此基础上，用于此部署的SPACS URL将是：<http://localhost:8000/api/acs>。
- 我们的 SP 实体 ID 将使用相同的URL：<http://localhost:8000/api/acs>。

使用默认领域或创建一个新领域。



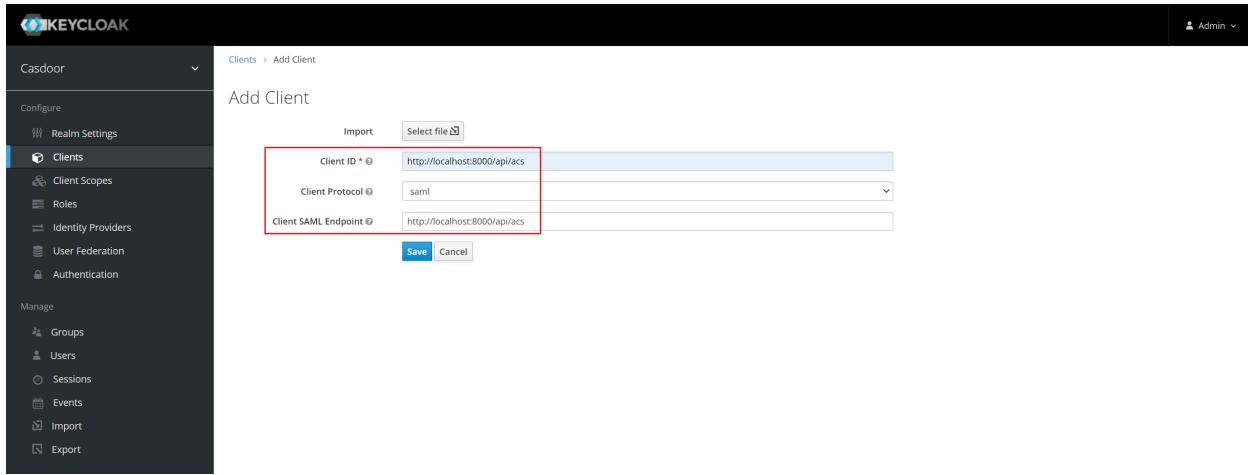
# 在 Keycloak 中添加客户端条目

## ① 信息

在 [Keycloak 文档](#) 中查看更多关于 Keycloak 客户端的详细信息。

在菜单中点击 **客户端** 然后点击 **创建** 去到 **添加客户端** 页面。 填写如下。

- **客户端 ID:** `http://localhost:8000/api/acs` - 这将是以后在 Casdoor 配置中使用的 SP 实体ID。
- **Client Protocol:** `saml`.
- **Client SAML Endpoint:** `http://localhost:8000/api/acs`. - 此 URL 是您想要 Keycloak 服务器发送SAML 请求和响应的地方。一般情况下，应用程序有一个用于处理 SAML 请求的URL。可以在客户端的设置选项卡中设置多个URL。



单击 **Save** (保存)。此动作创建客户端并将您带到 **设置** 选项卡。

以下设置列表部分：

1. **名称** - **Casdoor**. 这只用于在Keycloak UI中向Keycloak 用户显示友好的名称。它可以供你使用。
2. **启用** - 开启选择。
3. **包括作者声明** - 选择开启
4. **签名文档** - 选择
5. **签名声明** - 关闭。
6. **加密说明** - 关闭
7. **需要客户端签名** - 请关闭
8. **强制命名格式** - 选择开启
9. **名称 ID 格式** - 选择用户名。
10. **有效重定向 URI** - 添加 **http://localhost:8000/api/acs**.
11. **Master SAML 处理 URL** - **http://localhost:8000/api/acs**.
12. 精良的谷物SAML端点配置
  - i. **声明消费者服务公开绑定URL** - **http://localhost:8000/api/acs**。
  - ii. **声明消费者服务重定向绑定URL** - **http://localhost:8000/api/acs**。

保存该配置。

The screenshot shows the Keycloak administration interface with the following details:

- Left Sidebar:** Shows the navigation menu with "Casdoor" selected under "Clients". Other options include "Configure", "Realm Settings", "Client Scopes", "Roles", "Identity Providers", "User Federation", and "Authentication".
- Header:** Shows the URL "Clients > http://localhost:8000/api/acs" and a user "Admin".
- Main Content:** The "Settings" tab is active, showing the following configuration for the client "Casdoor":
  - Client ID:** http://localhost:8000/api/acs
  - Name:** Casdoor
  - Description:** (empty)
  - Enabled:** ON
  - Always Display in Console:** OFF
  - Consent Required:** OFF
  - Login Theme:** (dropdown menu)
  - Client Protocol:** saml
  - Include AuthnStatement:** ON
  - Include OneTimeUse Condition:** OFF
  - Force Artifact Binding:** OFF
  - Sign Documents:** ON
  - Optimize REDIRECT signing key lookup:** OFF
  - Sign Assertions:** OFF
  - Signature Algorithm:** RSA\_SHA256
  - SAML Signature Key Name:** KEY\_ID
  - Canonicalization Method:** EXCLUSIVE
  - Encrypt Assertions:** OFF
  - Client Signature Required:** OFF
  - Force POST Binding:** OFF
  - Front Channel Logout:** ON
  - Force Name ID Format:** ON
  - Name ID Format:** username
  - Root URL:** (empty)
  - Valid Redirect URIs:** http://localhost:8000/api/acs (with a minus sign icon)
  - Base URL:** (empty)
  - Master SAML Processing URL:** http://localhost:8000/api/acs
  - IDP Initiated SSO URL Name:** (empty)
  - IDP Initiated SSO Relay State:** (empty)
- Advanced Options:** "Fine Grain SAML Endpoint Configuration" and "Advanced Settings" sections are collapsed.
- Bottom:** "Save" and "Cancel" buttons.

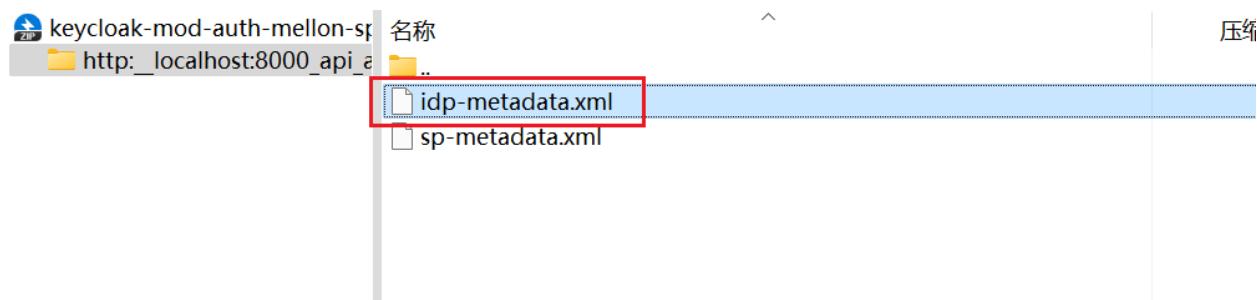
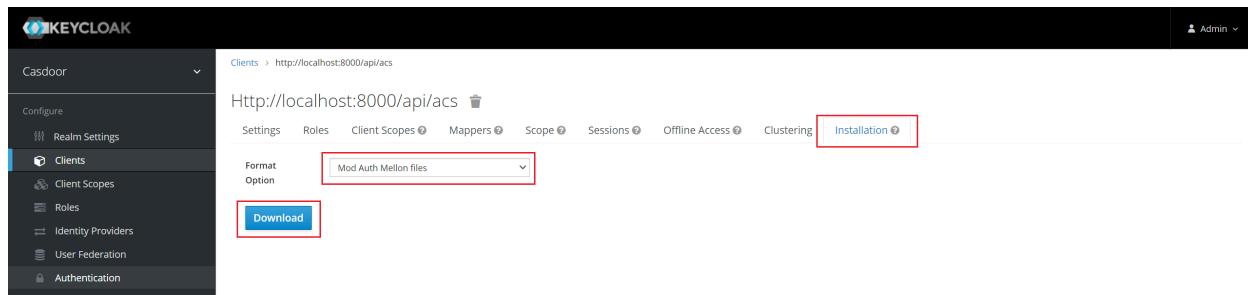
## 💡 提示

如果您想要签名authn请求，您需要启用 **客户端签名需要** 选项并上传您自己生成的证书。在Cassdoor, `token_jwt_key.key` 和 `token_jwt_key.pem` 中使用的私钥和证书位于 **对象** 目录。在Keycloak下，需要点击 **Keys** 页签，点击 **Import** 按钮，选择 **Archive Format** 为 **Certificate PEM** 并上传证书。

点击 **安装** 标签页。

对于Keycloak <= 5.0.0.0, 选择格式选项 - **SAML 元数据 IDPSODescriptor** 并复制元数据。

对于Keycloak 6.0.0+, 选择格式选项 - **Mod Mellon 文件** 并点击 **下载**。解压缩下载的 `.zip`, 定位 `idp-metadata.xml` 并复制元数据。



## 在Casdoor配置

在 Casdoor 中创建一个新的提供商。

选择分类为 **SAML**, 输入 **Keycloak**. 复制元数据内容并粘贴到 **元数据输入**。 端点 (**Endpoint**) 的值, **IdP** 和 **Issuer URL** 将在点击 **分析** 按钮后自动生成。 最后点击按钮 **保存**。

### 💡 提示

如果您在 **Keycloak** 中启用 **客户端签名需要** 选项并上传证书, 请在 **Casdoor** 中启用 **签名请求** 选项。

Name ⓘ: keycloak-casdoor

Display name ⓘ: keycloak-casdoor

Category ⓘ: SAML

Type ⓘ: Keycloak

Client ID ⓘ:

Client secret ⓘ:

Sign request ⓘ:

Metadata ⓘ:  Parse

Endpoint ⓘ: http://localhost:8080/auth/realm/casdoor/protocol/saml

IdP ⓘ: MIICnTCCAYUCBgF9pAmxSDANBqkqhkIG9w0BAQsFADASMRawDgYDVQQDDAdjYXNkb29yMB4XDThMTIxMDExMDg1OFoXDTMxMTIxMDExMTAzOFowEjEQMA4GA1UEAwwHY2FzZG9vcjCCASlwDQYJKoZIhvcNAQEBBQADggEPADCCAQ:

Issuer URL ⓘ: http://localhost:8080/auth/realm/casdoor

SP ACS URL ⓘ: http://localhost:8000/api/acs Copy

SP Entity ID ⓘ: http://localhost:8000/api/acs Copy

Provider URL ⓘ: <https://github.com/organizations/xxx/settings/applications/1234567>

编辑您想要在 **Cassdoor** 中配置的应用程序。 选择刚刚添加的身份提供商, 然后点击按钮 **保存**。

Providers ⓘ:	Providers <button>Add</button>	Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
		casdoor-idaas							  
		keycloak-casdoor							  

# 验证效果

转到您刚刚配置的应用程序，您可以在登录页面找到Keycloak图标。

点击图标跳转到Keycloak登录页面，验证后成功登录到Casdoor。

A screenshot of a login form. At the top left is the Casdoor logo. Below it are two input fields: one for "username, Email or phone" and one for "Password". Underneath the password field is a "Forgot password?" link. To the left of the "Sign In" button is a "Auto sign in" checkbox which is checked. Below the "Sign In" button are links for "Sign in with code" and "No account? sign up now". At the bottom right of the form are two small circular icons.

# 支付

## Overview

Add Payment providers to your application

## PayPal

Add PayPal Payment provider to your application

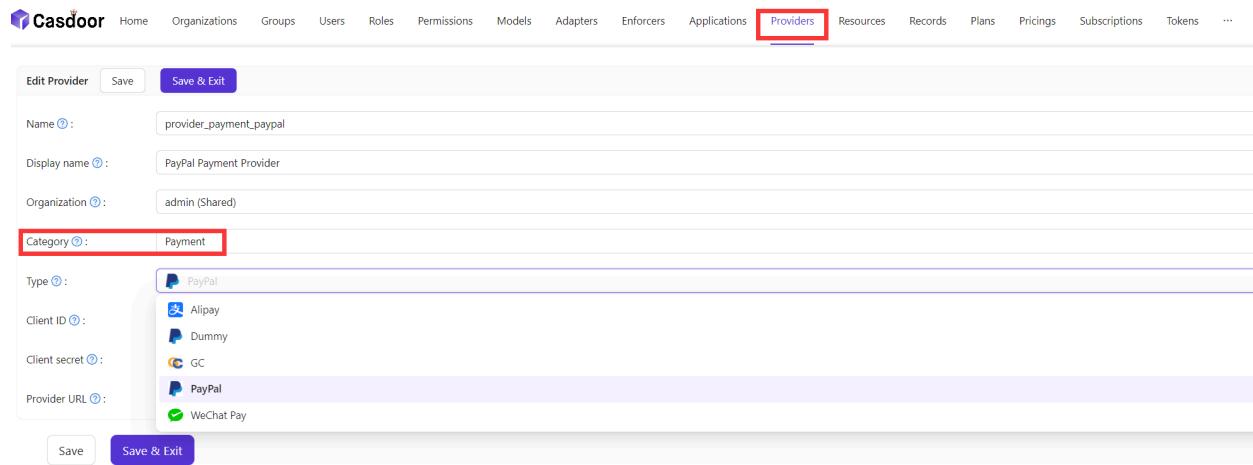
## 支付宝

## WeChatPay

Add Wechat OAuth provider to your application

# Overview

If you want to use payment services in Casdoor, you need to create a Payment provider and add it to your products.



The screenshot shows the Casdoor interface for managing payment providers. The top navigation bar includes links for Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Enforcers, Applications, **Providers** (which is highlighted with a red box), Resources, Records, Plans, Pricings, Subscriptions, Tokens, and more. Below the navigation is a form titled "Edit Provider". The "Name" field contains "provider\_payment\_paypal", and the "Display name" field contains "PayPal Payment Provider". The "Organization" field is set to "admin (Shared)". The "Category" field is set to "Payment" (highlighted with a red box). Under the "Type" section, "PayPal" is selected from a dropdown menu. In the "Client ID" and "Client secret" fields, there are dropdown menus showing "Alipay", "Dummy", and "GC". The "Provider URL" field has a dropdown menu with "PayPal" and "WeChat Pay" options, where "PayPal" is checked. At the bottom of the form are "Save" and "Save & Exit" buttons.

To learn how to configure a product, see [Product](#). After configure a product, you can add Payment providers for the product so that users can purchase the product through the Payment providers.

# PayPal

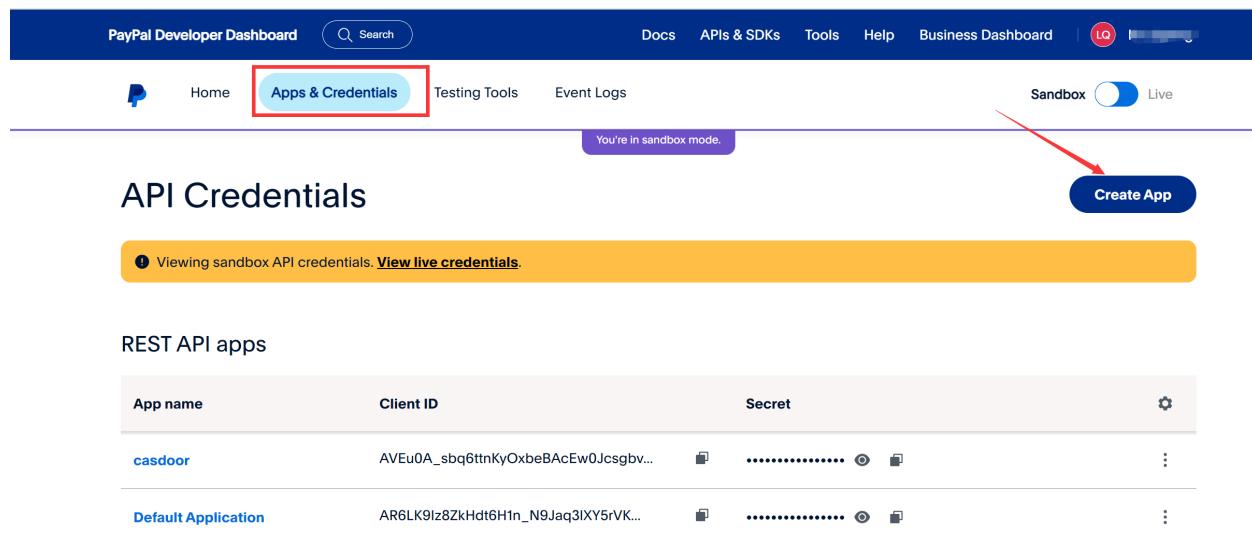
## ⓘ 备注

This is an example of how to configure a PayPal Payment provider.

## Step1. Create a PayPal application

First you need to create an application in PayPal. To access PayPal Developer site, you should have a PayPal business account. If you don't have an account then [create one](#) first.

After you create a PayPal business account, login the [Developer Dashboard](#) via the account and then click on [Create App](#) under [Apps & Credentials](#).



The screenshot shows the PayPal Developer Dashboard interface. At the top, there's a navigation bar with links for Docs, APIs & SDKs, Tools, Help, and Business Dashboard. On the left, there's a sidebar with Home, Apps & Credentials (which is highlighted with a red box), Testing Tools, and Event Logs. A message 'You're in sandbox mode.' is displayed. On the right, there's a toggle switch for Sandbox (which is turned on) and Live. Below the navigation, the main content area is titled 'API Credentials'. It shows a yellow banner stating 'Viewing sandbox API credentials. [View live credentials](#)'. Under the heading 'REST API apps', there's a table listing two applications:

App name	Client ID	Secret	⋮
casdoor	AVEu0A_sbq6tnKyOxbeBAcEw0Jcsgbv...	.....	⋮
Default Application	AR6LK9lz8ZkHdt6H1n_N9Jaq3IXY5rVK...	.....	⋮

You can find the [Client ID](#) and [Secret key](#) in the basic information of your

application.

← Back

## casdoor

Viewing sandbox API credentials. [View live credentials.](#)

### API credentials

App name	casdoor
Client ID	AVEu0A_sbq6trnKyOxbeBAcEw0Jcsgbv2JZvQAtK JFnaULI-EK-U2XIXcEpEouO9olknbU7c3m_lIRT5
Secret key 1	*****   

+ Add Second Key

### Sandbox account info

[View details](#)

Sandbox URL	<a href="https://sandbox.paypal.com">https://sandbox.paypal.com</a> 
Sandbox Region	C2
Email	sb-qqaiv26894991@business.example.com 
Password	*****  

### Features

## Step2. Create a PayPal Payment provider

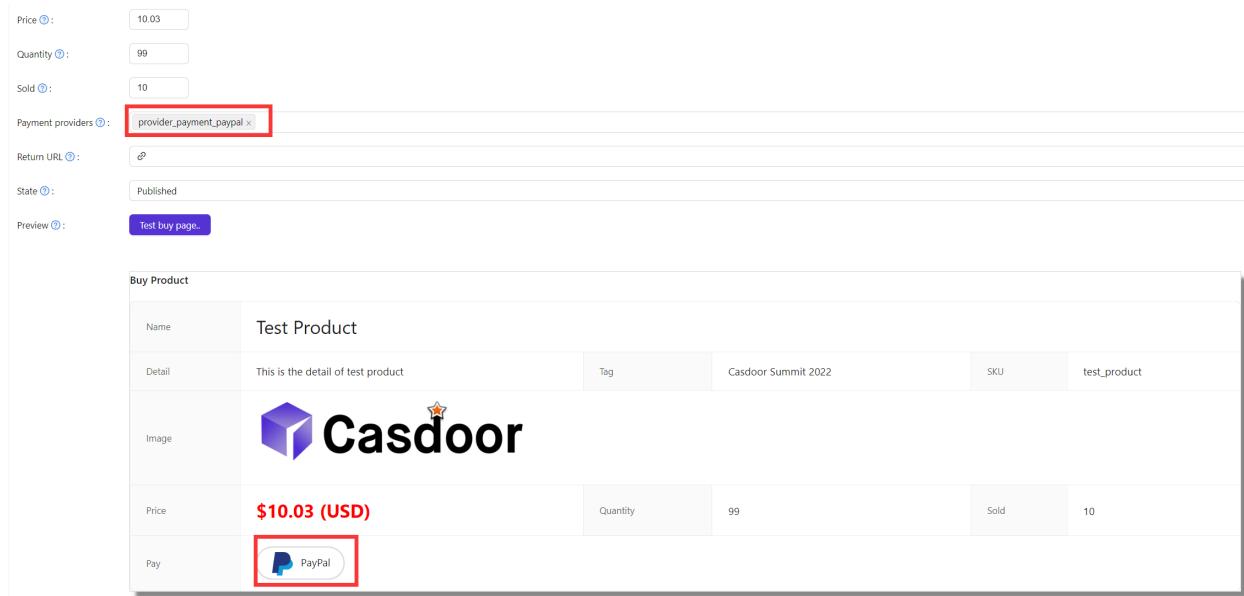
Then create a PayPal Payment provider in Casdoor. Fill the necessary information.

Name	Name in PayPal
Category	choose 

Name	Name in PayPal
Type	choose <code>PayPal</code>
Client ID	<code>Client ID</code> obtained from Step1
Client secret	<code>Secret key</code> obtained from Step1

## Step3. Add the PayPal Payment provider for your product

Finally, Add the PayPal Payment provider for your product so that users can purchase the product using PayPal.



The screenshot shows the Casdoor product management interface. A new product is being created with the following details:

- Price:** 10.03
- Quantity:** 99
- Sold:** 10
- Payment providers:** provider\_payment\_paypal (highlighted with a red box)
- Return URL:** (empty)
- State:** Published
- Preview:** Test buy page..

Below the form, a preview window shows the product listing for "Test Product". The product details include:

Buy Product					
Name	Test Product				
Detail	This is the detail of test product				
Image	 Casdoor				
Price	\$10.03 (USD)	Quantity	99	Sold	10
Pay					

① 备注

The above operations are all performed in the PayPal `Sandbox` mode. If you want to use it in a live production environment, you need to create an application in PayPal `Live` mode and set `runmode=prod` in Casdoor's configure file `conf/app.conf`.

 > 提供商 > 支付 > **支付宝**

# 支付宝

# WeChatPay

## Step1. Deploy Casdoor

Firstly, the Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#).

Please deploy your Casdoor instance in production mode.

After a successful deployment, you need to ensure:

- Open your favorite browser and visit <http://localhost:8000>, you will see the login page of Casdoor.
- Input `admin` and `123` to test login functionality is working fine.

Then you can quickly implement a casdoor based login page in your own app with the following steps.

## Step2. Configure payment callback notification address

Before that, please log in to the WeChat Pay Merchant Platform. In order for Casdoor to receive payment result notifications, you need to set a callback notification address in the WeChat Pay Merchant Platform.

- In Merchant Dashboard, go to `Development Configuration` → `Payment Configuration`.
- Find the `Callback Notification Address` setting, and click the `Modify`

button.

- Fill in the payment callback notification address of the Casdoor instance. For example:

```
https://your-casdoor-instance.com/api/wechat-payment-callback
```

## Step3. Configure API Security

### Get APIv3 key

To ensure the security of API calls, you need to configure API security settings in the WeChat Pay Merchant Platform.

Log in the [WeChat merchant platform](#) → [Account center](#) → [Account settings](#) → [API security](#) → [APIv3 key](#) → [set up](#)

### Get the serial number of the merchant certificate

It is also necessary to obtain the serial number of the merchant certificate. The following are the steps to obtain it.

Log in the [WeChat merchant platform](#) → [Account center](#) → [Account settings](#) → [API security](#) → [API certificate management](#) → copy the serial number.

You can refer to the [WeChat payment merchant ID query guide](#), [APIv3 key settings](#) and [How to view the certificate serial number](#) for help.

## Step4. Add payment provider

Select the WeChat Pay as the Payment type

New Provider    Save    Save & Exit    Cancel

Name ? : wechat-provider

Display name ? : New wechat-provider

Organization ? : admin (Shared)

Category ? : **Payment**

Type ? : WeChat Pay

Client ID ? : Alipay  
GC

Client secret ? : PayPal  
**WeChat Pay**

Provider URL ? : <https://github.com/organizations/xxx/settings/applications/123456/>

## Get Payment URL

Log in the [WeChat merchant platform](#) → [Commodity centered](#) → [Development arrangement](#) → [payment arrangements](#) → [Payment URL](#) → [copy](#)

## Fill the necessary information in Casdoor

There are four required fields, `Client ID`, `Client secret`, `appId`, `Provider URL`. The relationship corresponding to the Azure Blob account is as follows:

Name	Name in WeChat Pay	is required
Client ID	merchant ID	required
Client secret	APIv3 key	required
appId	appId	required
Provider URL	Payment URL	required

The acquisition of `merchant ID` and `APIv3 key` is as mentioned before. For `appId`, see [here](#) for more help.

Edit Provider Save Save & Exit

Name ? : wechat-provider

Display name ? : New wechat-provider

Organization ? : admin (Shared)

Category ? : Payment

Type ? : WeChat Pay

Client ID ? e75c5c9f0056d80849a7 Your merchant ID

Client secret ? \*\*\* Your APIv3 key

appId ? appldadaasdaffafgaass Your appId

Provider URL ? https://api.mch.weixin.qq.com/pay/unifiedorder Your Payment URL

Save Save & Exit

Client ID ?	e75c5c9f0056d80849a7	Your merchant ID
Client secret ?	***	Your APIv3 key
appId ?	appldadaasdaffafgaass	Your appId
Provider URL ?	https://api.mch.weixin.qq.com/pay/unifiedorder	Your Payment URL

## Step5. Add WeChat Cert

In this step, you need two required fields, `serial number`, `private key`.

How to obtain the serial number has been stated in the third step.

To get `private key`, click [here](#) for help.

Edit Cert Save Save & Exit

Name ? : cert\_wechat  
x509

Display name ? : New Cert - wechat

Scope ? : JWT

Type ? : x509

Crypto algorithm ? : RS256

Bit size ? : 4096

Expire in years ? : 20

Your merchant certificate serial number

Certificate ? : Copy certificate Download certificate

```
-----BEGIN CERTIFICATE-----
MIIE2TCAsGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMCYxDjAMBg
NVBAoTBWFk
bWluMRQwEgYDVQQDAjZXJ0X2dmamRqZzAeFwOyMzA0MTAxMTEy
NTIaFw00MzA0
-----END CERTIFICATE-----
```

Private key ? : Your private key Copy private key Download private key

```
-----BEGIN RSA PRIVATE KEY-----
MIJKgIBAAKCAgEA4Fn+yt0cUCUlrMx/zXd0JnEbRCxBqo8weFf1LgBKi2h
6R1vD
cZfFRRvlHA/Oktl5nKXQPuQzwuxHy6Cz2HRoAR0ayNVDcASJTWOtOF/
z0vYRKT/
-----END RSA PRIVATE KEY-----
```

## Step6. Add WeChat Payment Provider in Products

Currency ? : USD

Price ? : 300

Quantity ? : 99

Sold ? : 10

Payment providers ? : wechat-provider x

The payment provider you just add

wechat-provider

Return URL ? : ↲

The payment callback notification address same in WeChat merchant platform

State ? : Published

Preview ? : Test buy page..

Final effect:

Buy Product					
Name	New Product - 4emdrq				
Detail		Tag	Casdoor Summit 2022	SKU	product_4emdrq
Image	 Casdoor				
Price	\$300 (USD)	Quantity	99	Sold	10
Pay	 WeChat Pay				

## What's more

You can explore the following projects/docs to learn more about WeChat Pay.

- [wechatpay-apiv3-go-sdk](#)
- [Wechat payment development document](#)

# 验证码

## 概述

添加验证码到您的应用程序

## 默认

在您的应用程序中使用 Casdoor 默认验证码

## Cloudflare Turnstile

Add Cloudflare Turnstile to your application

## reCAPTCHA

添加reCAPTCHA 到您的应用程序

 **hCaptcha**

将 hCaptcha 添加到您的应用程序

 **Aliyun Captcha**

将Aliyun Captcha添加到您的应用程序

 **Geetest**

Add Geetest Captcha to your application

# 概述

Casdoor可以配置支持不同的验证码，以检查是否是人为操作。如果您添加了验证码提供商并在应用程序中应用，用户登录时，注册或忘记密码并需要发送代码，然后验证码检查对话框将会检查是否是人为操作。

现在，Casdoor支持许多验证码提供商。以下是Casdoor支持的提供商：

默认	Cloudflare Turnstile	reCAPTCHA	hCaptcha	Aliyun Captcha	Geetest

我们将向您展示如何申请验证码并将其添加到casdoor。

## 添加验证码提供商

1. 导航到您的Casdoor索引页面
2. 点击顶部栏中的[提供商](#)
3. 点击[添加](#)，然后您可以在列表顶部看到一个新的提供商
4. 点击新的提供商修改它
5. 选择[验证码](#)在[类别](#)中
6. 在[类型](#)中选择您需要的验证码提供程序

7. 填写最重要的信息，不同的验证码提供者有不同的信息需要填写

## 在应用中使用

1. 单击顶部栏中的 **应用程序** 并选择一个应用程序，编辑.
2. 点击提供商添加按钮，选择您刚刚添加的提供商.
3. 完成！

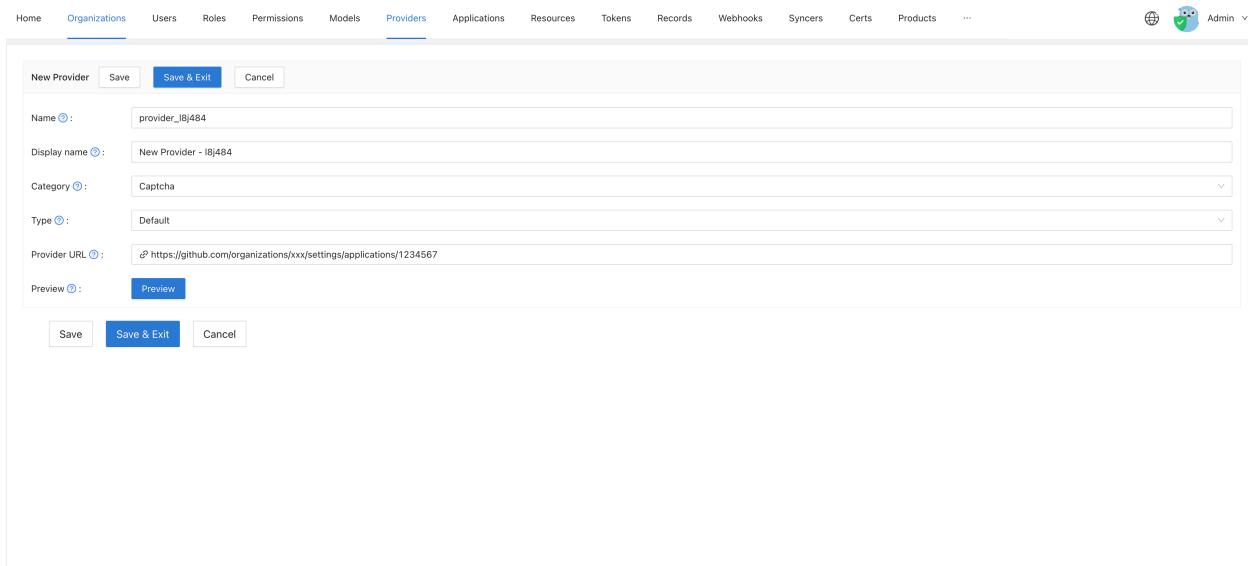
# 默认

默认验证码实现图像生成和验证。默认的验证码图像是带有定义长度(5)的位数0-9的顺序。

## 在 Casdoor 配置

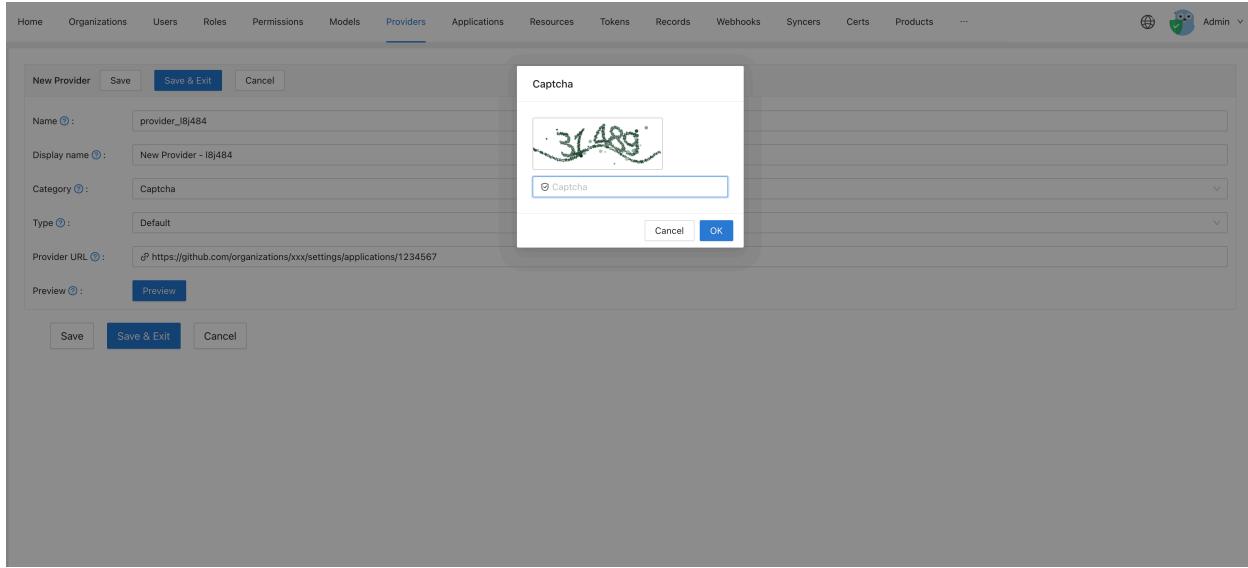
在 Casdoor 中创建一个新的提供商。

选择类别为 验证码，类型为 hCaptcha.



The screenshot shows the Casdoor provider configuration interface. The top navigation bar includes Home, Organizations, Users, Roles, Permissions, Models, Providers (which is highlighted), Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, Products, and a three-dot menu. On the far right, there is a user icon and Admin dropdown. The main content area is titled 'New Provider' with buttons for 'Save', 'Save & Exit', and 'Cancel'. It contains fields for Name (provider\_I8j484), Display name (New Provider - I8j484), Category (Captcha), Type (Default), and Provider URL (https://github.com/organizations/xxx/settings/applications/1234567). Below these fields is a 'Preview' button. At the bottom of the form are 'Save', 'Save & Exit', and 'Cancel' buttons. The entire form is contained within a light gray box.

你可以点击 预览 button 来预览这个验证码的样式。



# 在应用中使用

编辑您想要在 Cassdoor 中配置的应用程序。选择刚添加的提供商。There are three kinds of rules:

- **Always** Always turned on human-machine verification when login.
- **None** Never require human-machine verification, the account will be blocked when it attempted to login into the same application with wrong password for the 5st time within 15 minutes. And it will be unblocked after 15 minutes.
- **Dynamic** After 5 failed login attempts, the account will not be blocked but instead, human-machine verification will be required.

Providers								
Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Rule	Action
provider_4olfdm	Captcha		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Always	
provider_casdoor_github	OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
provider_casdoor_google	OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		

We also provide a demo video to demonstrate the differences in rules, which we hope will be helpful to you



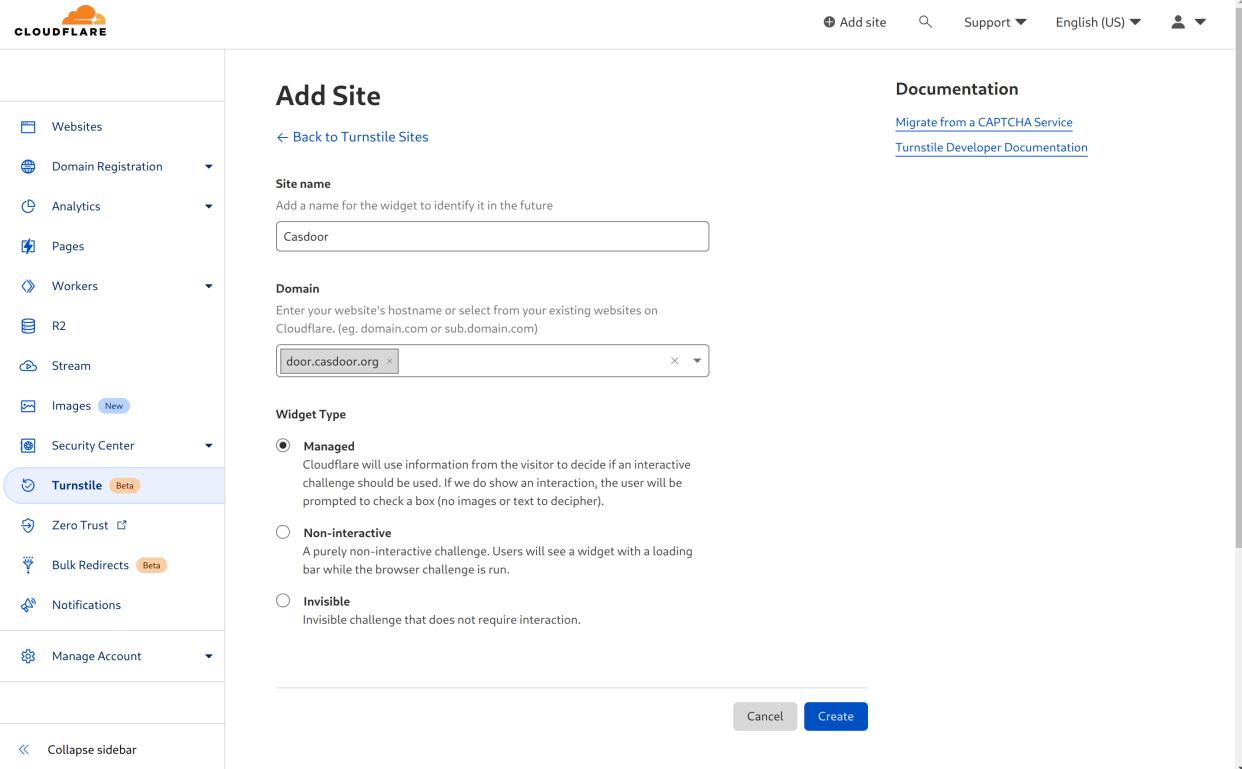
# Cloudflare Turnstile

Cloudflare Turnstile is a captcha service provided by Cloudflare, which is a user-friendly, privacy preserving alternative to captcha. You can see more details from [Turnstile Docs](#).

## Create an API key pair

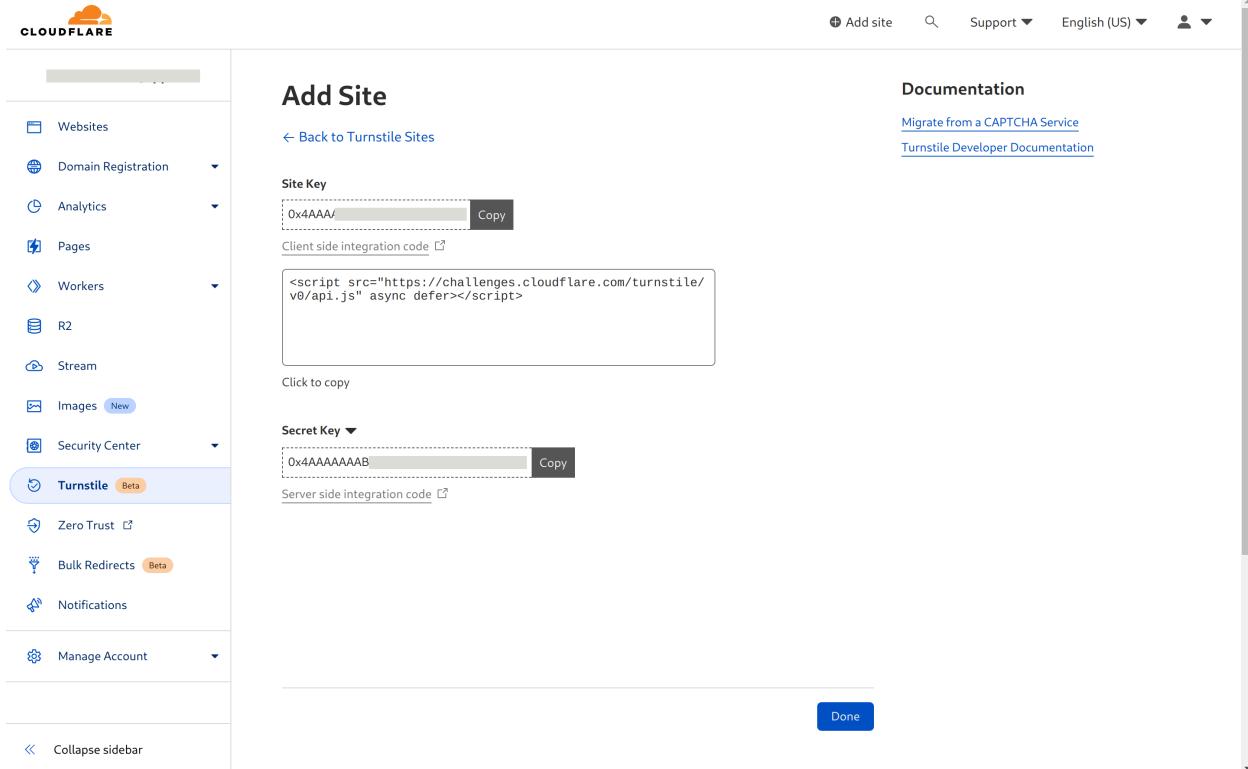
To start using Cloudflare Turnstile, you need to [Create a Cloudflare account](#), navigate to the [Turnstile](#) tab on the navigation bar, and get the Site Key and Secret Key.

First, add a name for the widget to identify it in the future and enter your website's hostname. Then choose the widget type. [Managed](#) is recommended. Click [Create](#).



The screenshot shows the Cloudflare dashboard with the sidebar open. The sidebar includes options like Websites, Domain Registration, Analytics, Pages, Workers, R2, Stream, Images (New), Security Center, Turnstile (Beta), Zero Trust, Bulk Redirects (Beta), Notifications, and Manage Account. The Turnstile option is selected and highlighted with a blue border. The main content area is titled "Add Site" and shows a "Site name" input field containing "Casdoor". Below it is a "Domain" input field containing "door.casdoor.org". Under "Widget Type", the "Managed" option is selected, with a description explaining Cloudflare's decision-making process for interactive challenges. There are also "Non-interactive" and "Invisible" options. At the bottom right are "Cancel" and "Create" buttons.

Then you can get a site key and a secret key.



The screenshot shows the Cloudflare dashboard with the sidebar open. The 'Turnstile' option under the 'Security Center' section is selected, indicated by a blue background and a 'Beta' badge. The main content area is titled 'Add Site' and shows the configuration for a Turnstile site. It includes fields for 'Site Key' (containing '0x4AAA/...' with a 'Copy' button) and 'Secret Key' (containing '0xAAAAAAAB...' with a 'Copy' button). Below these are sections for 'Client side integration code' (containing a script tag) and 'Server side integration code'. A 'Done' button is at the bottom right. The top navigation bar includes links for 'Add site', 'Support', 'English (US)', and user profile.

# Configure in Casdoor

Create a new provider in Casdoor.

Select category as **Captcha** , type as **Cloudflare Turnstile** . And you need to fulfill the site key and the secret key which is created by last step.

 Casdoor Home Organizations Users Roles Permissions Models Adapters Applications Providers Resources ... Admin

Edit Provider

Name ②: Cloudflare Turnstile

Display name ②: Cloudflare Turnstile

Organization ②: admin (share)

Category ②: Captcha

Type ②: Cloudflare Turnstile

Site key ②: 0x4AAAAAAABXhq3vOlgpUTmk

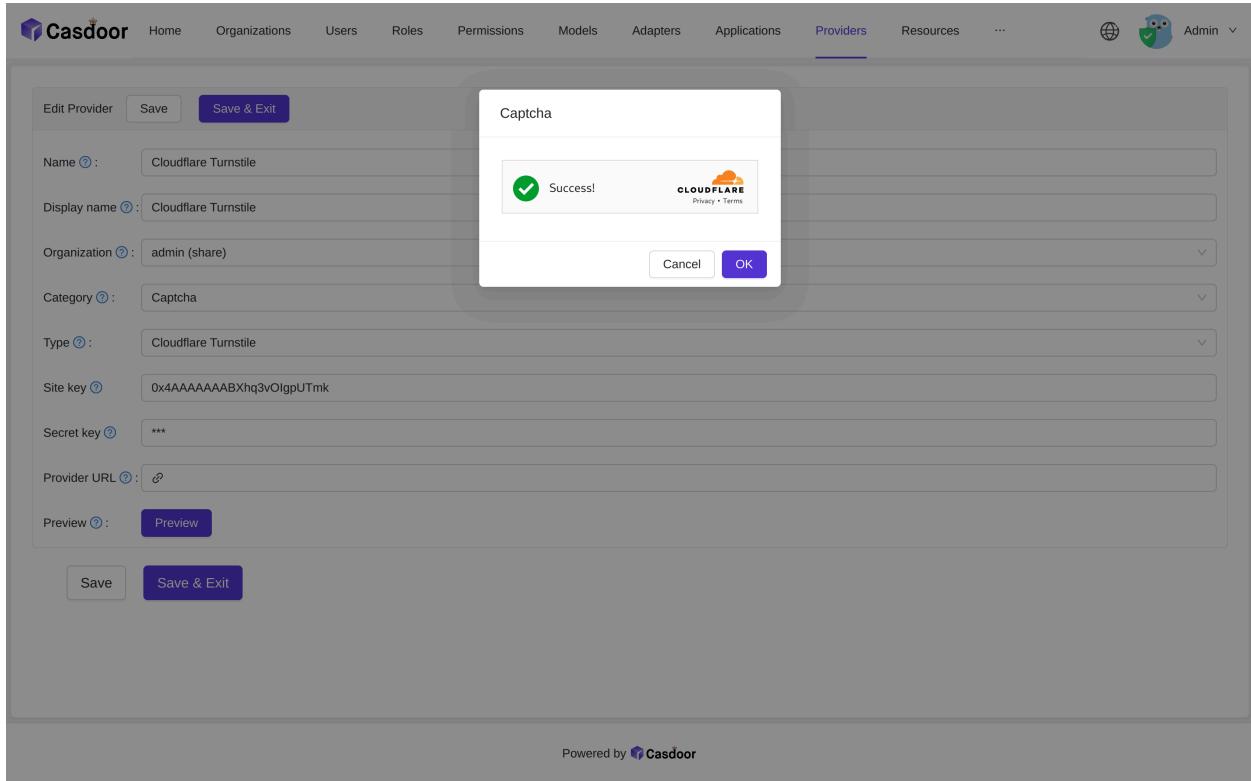
Secret key ②: \*\*\*

Provider URL ②:

Preview ②:

Powered by  Casdoor

And you can click the **Preview** button to preview the style of this captcha.



## Applied in application

Edit the application you want to configure in Casdoor. Select the provider just added and click the button Save.

Providers								
Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
Cloudflare Turnstile	Captcha						None	

# reCAPTCHA

reCAPTCHA 由 Google 提供。 我们使用 reCAPTCHA v2 Checkbox。 您可以从此 [链接](#) 中看到更多详细信息。

## 创建一个 API 密钥对

要开始使用 reCAPTCHA，您需要 [注册您的站点的 API 密钥对](#)。 配对的密钥由一个站点密钥组成。 站点密钥用于在您的站点或移动应用程序上调用 reCAPTCHA 服务。 密钥授权您的应用程序后端和 reCAPTCHA 服务器之间的通信来 [验证用户的回应](#)。

First, choose the [type of reCAPTCHA](#) and then fill in authorized domains or [package names](#). After you have accepted the terms of service, click **Register** to get a new API key pair.

The screenshot shows the Google reCAPTCHA registration interface. At the top, there's a blue header bar with the text "Google reCAPTCHA". Below it, a light blue bar says "← Register a new site". A yellow bar at the bottom says "Get unlimited assessments using reCAPTCHA Enterprise". The main area has a white background. It starts with a "Label" section containing a "reCaptcha" input field. Below that is a "reCAPTCHA type" section with a radio button selected for "reCAPTCHA-v2 Verify requests with a challenge". Underneath are three options: "I'm not a robot" Checkbox (selected), Invisible reCAPTCHA badge, and reCAPTCHA Android. The next section is "Domains" with a "+ casdoor.org" button. The "Owners" section lists "resultelee@gmail.com (You)" and a "Enter email addresses" input field. At the bottom, there's a checked checkbox for "Accept the reCAPTCHA Terms of Service" and a small note about agreeing to Google's Terms of Use and Additional Terms.

然后你可以获得一个站点密钥和一个秘密密钥。

The screenshot shows the Google reCAPTCHA registration interface. At the top, it says 'Adding reCAPTCHA to your site'. Below that, it states "'reCaptcha' has been registered.''. It provides instructions to use the site key in HTML code and the secret key for communication between the site and reCAPTCHA. Two text input fields are shown: 'Site key' containing '6Lc...' and 'Secret key' containing '6U...'. Each field has a 'COPY' button next to it. At the bottom, there are 'GO TO SETTINGS' and 'GO TO ANALYTICS' buttons.

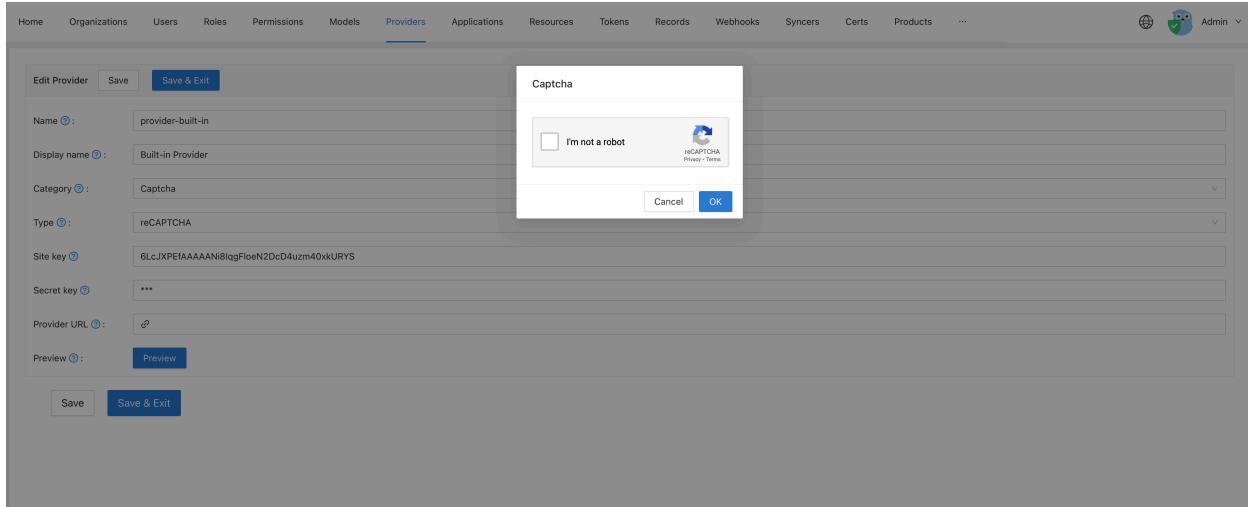
## 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

选择类别为 **Captcha** , 输入 reCAPTCHA. 你需要完成最后一步创建的站点密钥和秘密密钥。

The screenshot shows the Casdoor provider configuration interface. The top navigation bar includes Home, Organizations, Users, Roles, Permissions, Models, Providers (which is selected), Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, Products, and more. On the right, there is a user icon labeled 'Admin'. The main form is titled 'New Provider' and contains fields for Name (reCaptcha), Display name (reCaptcha), Category (Captcha), Type (reCAPTCHA), Site key (6Lc...), Secret key (6U...), Provider URL (https://github.com/organizations/xx/settings/applications/1234567), and Preview. Buttons at the bottom include Save, Save & Exit, and Cancel.

你可以点击 **预览** button 来预览这个验证码的样式。



# 在应用中使用

编辑您想要在 Cassdoor 中配置的应用程序。选择刚刚添加的身份提供商，然后点击按钮 **保存**。

Providers	Add	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
reCaptcha	Captcha							

# hCaptcha

hCaptcha 是一个验证码服务提供商，类似于reCAPTCHA。您可以从此 [链接](#) 中看到更多详细信息。

## 创建一个 API 密钥对

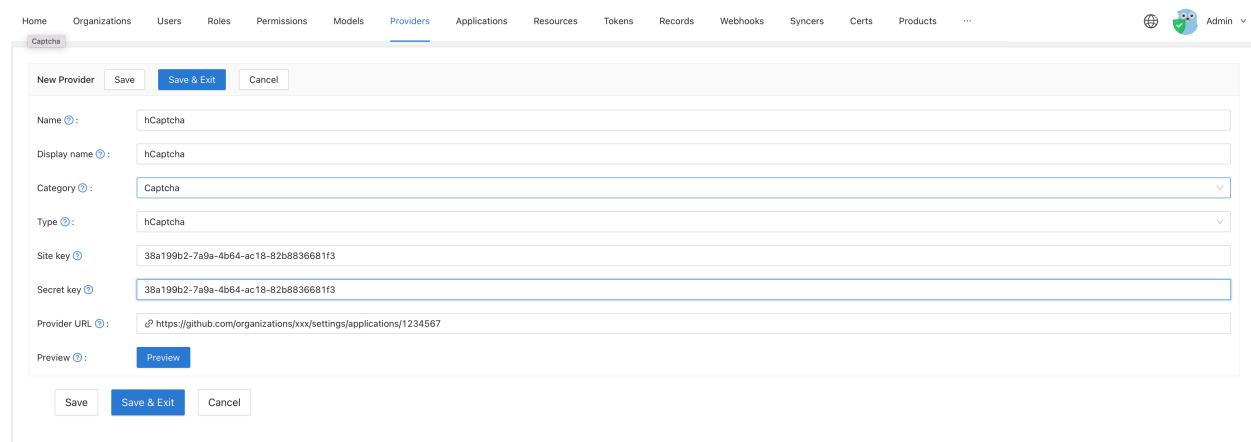
若要开始使用 hCaptcha，您需要 [注册您的网站的 API 密钥对](#)。您可以在您的 [个人资料页面](#) 找到您的网站密钥。

然后你可以获得一个站点密钥和一个秘密密钥。

## 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

选择类别为 **验证码**，类型为 **hCaptcha**。你需要完成最后一步创建的站点密钥和秘密密钥。

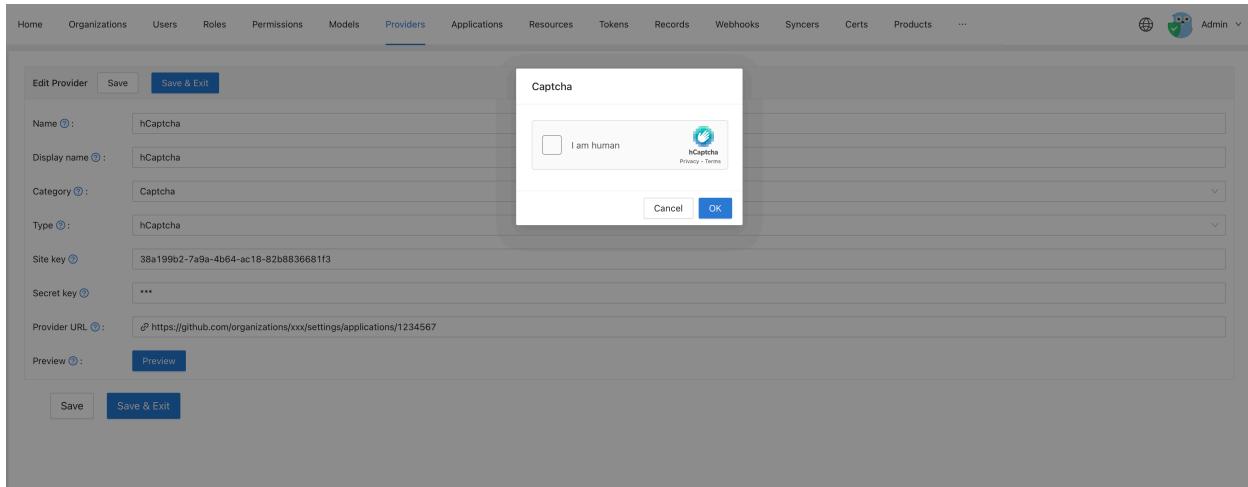


The screenshot shows the Casdoor web interface for managing providers. The top navigation bar includes links for Home, Organizations, Users, Roles, Permissions, Models, Providers (which is currently selected), Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, Products, and more. On the far right, there's a user icon labeled 'Admin'.

The main content area is titled 'Captcha'. It contains a form with the following fields:

- New Provider (button)
- Save (button)
- Save & Exit (button)
- Cancel (button)
- Name: hCaptcha
- Display name: hCaptcha
- Category: Captcha
- Type: hCaptcha
- Site key: 38a199b2-7a9a-4b64-ac18-82b8836681f3
- Secret key: 38a199b2-7a9a-4b64-ac18-82b8836681f3
- Provider URL:
- Preview (button)
- Save (button)
- Save & Exit (button)
- Cancel (button)

你可以点击 **预览** button 来预览这个验证码的样式。



## 在应用中使用

编辑您想要在 Cassdoor 中配置的应用程序。选择刚刚添加的身份提供商，然后点击按钮 **保存**。



# Aliyun Captcha

Aliyun Captcha是由Aliyun提供的验证码服务。 它包括两种验证验证码方式： [滑动验证](#) and [智能验证](#)。 您可以从此 [链接](#) 中看到更多详细信息。

## 在 Aliyun 中添加验证码配置

登录到 [Aliyun 管理控制台](#), 搜索并前往验证码服务。 然后点击 **确认打开** 以启用验证码服务。



输入验证码关联控制台后，点击 **添加配置**。

公告: 2021年3月18日起, 人机验证产品统一更名为验证码。

配置名称	appkey	scene	验证方式	业务类型	使用场景	最后更新	操作
测试验证码	F [REDACTED] B	nc_other	滑动验证	PC	其它	2022-06-20 23:47:37	自定义样式 系统代码集成
测试智能验证码	FF [REDACTED] B	lc_other	智能验证	PC	其它	2022-06-21 00:44:08	自定义样式 系统代码集成

共有2条 < 1 >

填写所有必需的信息并提交。

① 配置服务内容

② 系统代码集成&测试

③ 完成

然后您可以在您的控制台中看到您的 场景 and App key

The screenshot shows the Alibaba Cloud Workbench interface under the '验证码' (Captcha) section. It displays a table of configurations with two rows highlighted by red boxes. The first row contains 'appkey' and 'nc\_other'. The second row contains 'F...8B' and 'lc\_other'. The table includes columns for Configuration Name, Value, Validation Method, Business Type, Usage Scenario, Last Update, and Operation.

配置名称	值	验证方式	业务类型	使用场景	最后更新	操作
测试验证码	F...8B	滑动验证	PC	其它	2022-06-20 23:47:37	自定义样式   系统代码集成
测试智能验证码	F...8B	智能验证	PC	其它	2022-06-21 00:44:08	自定义样式   系统代码集成

访问密钥，秘密访问密钥 在您的个人资料中。

## 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

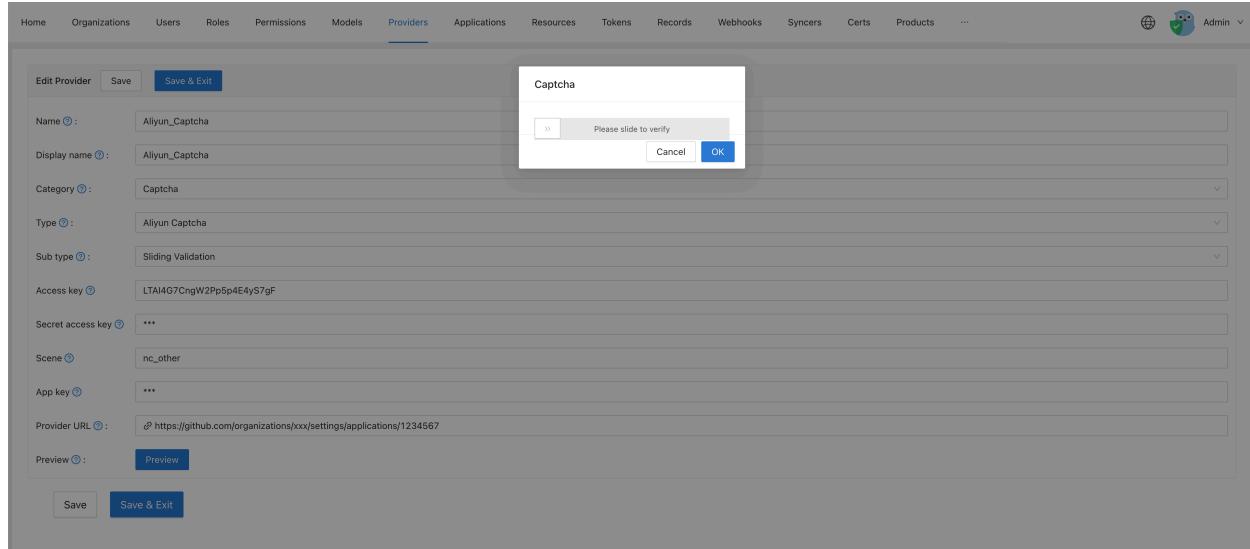
选择类别为 验证码，类型为 hCaptcha. 然后选择子类型： 滑动验证 或 智能验证。 您需要完成 访问密钥，秘密访问密钥，场景 和 应用密钥，这都是最后一步创建。

The screenshot shows the Casdoor provider configuration page for 'Aliyun\_Captcha'. The form includes fields for Name, Display name, Category, Type, Sub type, Access key, Secret access key, Scene, App key, and Provider URL. The 'Save & Exit' button is highlighted in blue.

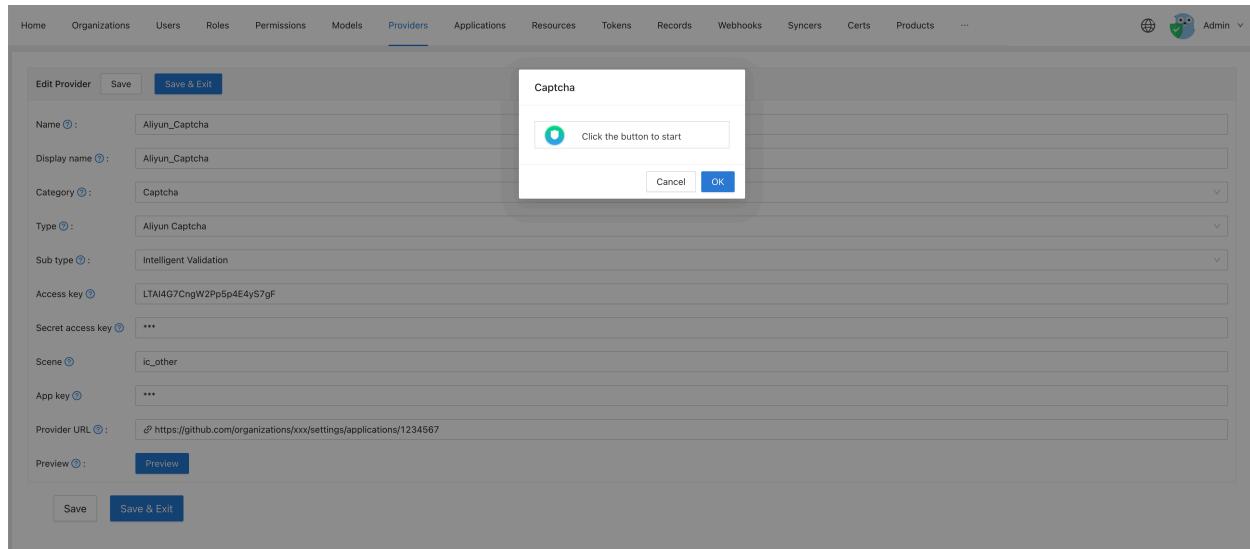
Name	Aliyun_Captcha
Display name	Aliyun_Captcha
Category	Captcha
Type	Aliyun Captcha
Sub type	Sliding Validation
Access key	L...gF
Secret access key	jIP...N
Scene	nc_other
App key	F...BB
Provider URL	https://github.com/organizations/xxx/settings/applications/1234567

你可以点击 预览 button 来预览这个验证码的样式。

下面的图像是 滑动验证 预览：



以下图像是 智能验证 预览：



# 在应用中使用

编辑您想要在 Cassdoor 中配置的应用程序。选择刚刚添加的身份提供商，然后点击按钮 **保存**。

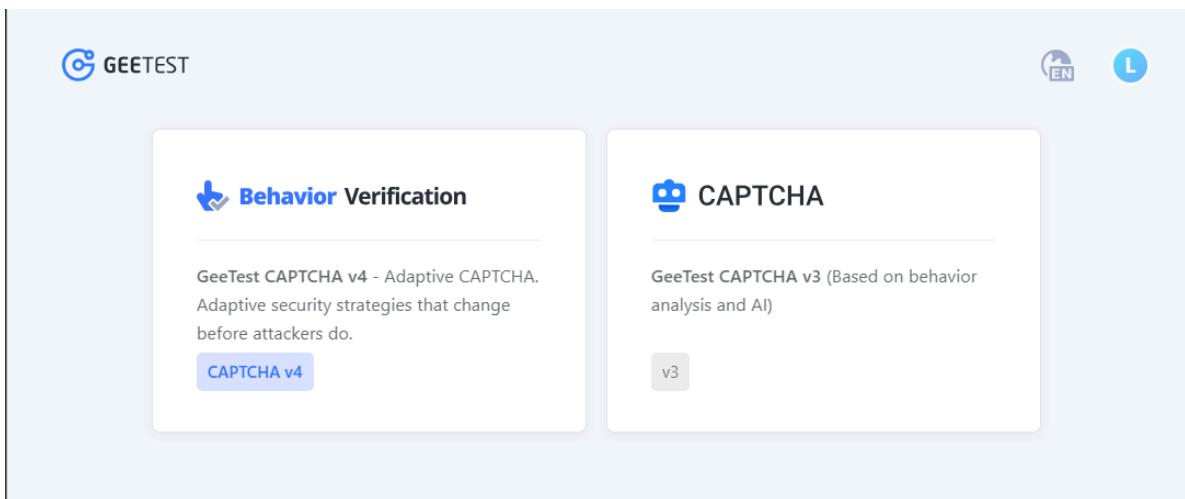
Providers <small>(1)</small>	Add	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
Aliyun_Captcha	<input checked="" type="checkbox"/> Captcha							

# Geetest

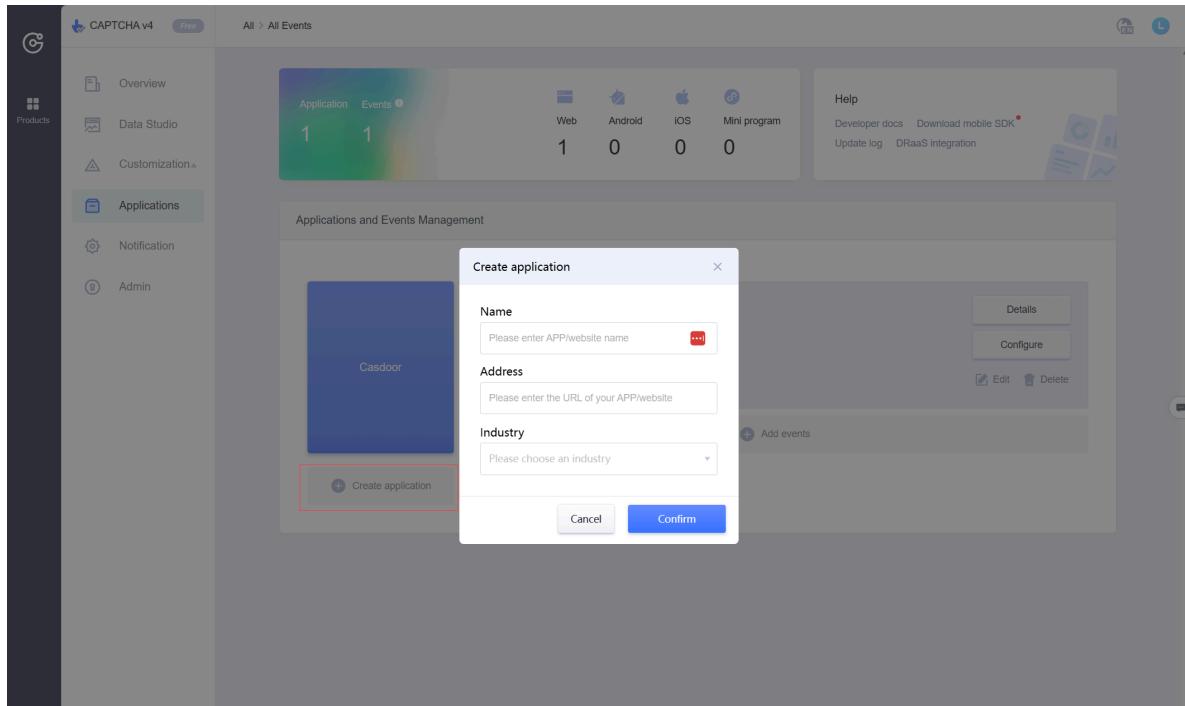
## Configure Geetest

Configure Geetest and get the public and secret key

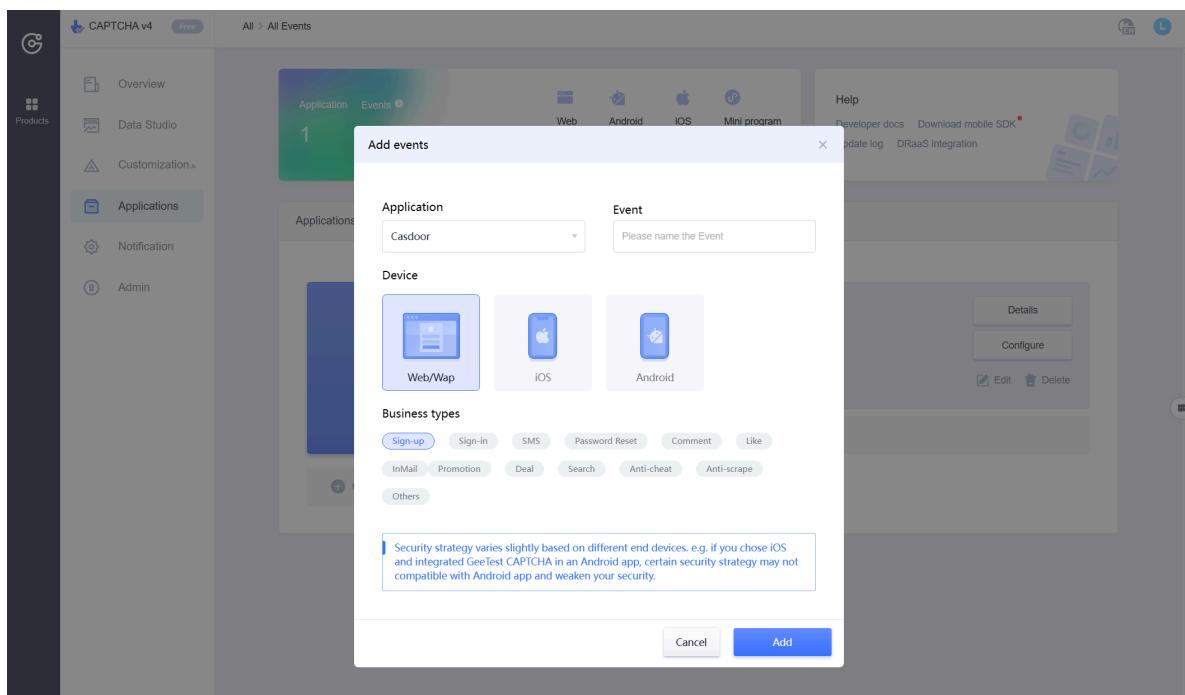
1. Go to the Geetest CAPTCHA V4 in [geetest product page](#)



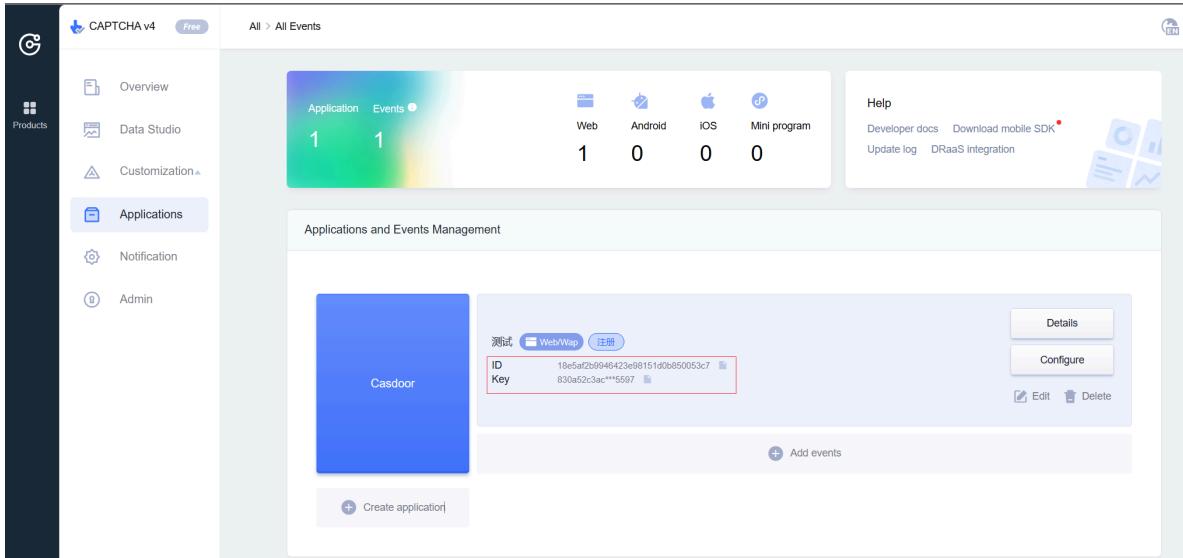
2. Create Application : enter the Name and address to your application.



### 3. Add events: choose web for device



### 4. Get ID and Key



# Configure casdoor

1. Create a new provider in Casdoor

Select category as **Captcha**, type as **Geetest**. Fill in the **Site key** and **Secret key** with ID, Key in Geetest.

2. Click Preview button to preview the style of this captcha.

The screenshot shows the Casdoor web interface for managing providers. The top navigation bar includes links for Home, Organizations, Users, Roles, Permissions, Models, Providers (which is the active tab), Applications, Resources, Tokens, Records, and Admin. The main content area is titled "Edit Provider" and contains fields for Name, Display name, Category, Type, Site key, Secret key, Provider URL, and Preview. Buttons for Save and Save & Exit are at the bottom. The footer indicates the application was made with ❤️ by Casdoor.

# Applied in application

Edit the application you want to configure in Casdoor. Select the provider just added and click the button Save.

Providers	Add	Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
geetest			Captcha	Geetest						



&gt; 提供商

&gt; Web3

# Web3



## MetaMask

Add MetaMask Web3 provider to your application

# MetaMask



备注

This is an example of configure MetaMask as Web3 provider.

MetaMask is a browser extension and app that describes itself as a crypto wallet and a gateway to blockchain apps. Casdoor allows using MetaMask as an identity provider and enables Web3 login with MetaMask.

## Step1. Create a MetaMask Web3 provider

First, you need to create a MetaMask Web3 provider in Casdoor.

Name	Description
Category	choose <a href="#">Web3</a>
Type	choose <a href="#">MetaMask</a>

Edit Provider
Save
Save & Exit

---

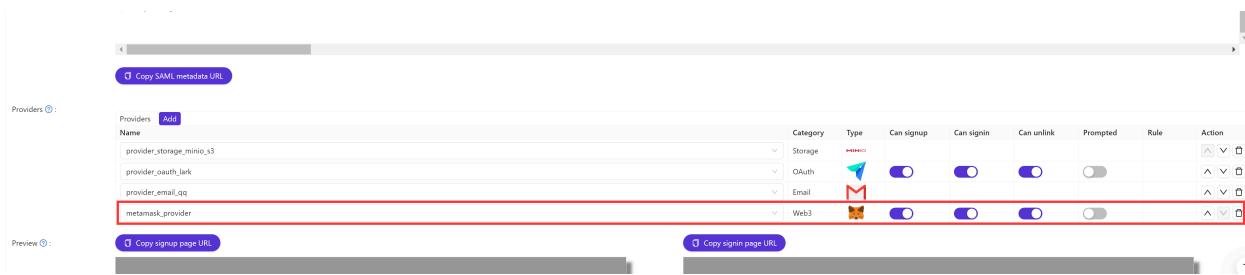
Name <span>?</span> :	metamask_provider
Display name <span>?</span> :	MetaMask Provider
Organization <span>?</span> :	admin (Shared)
Category <span>?</span> :	Web3
Type <span>?</span> :	MetaMask
Provider URL <span>?</span> :	<a href="#">🔗</a>

---

Save
Save & Exit

## Step2. Add provider to your application

Second, add MetaMask Web3 provider to your application.



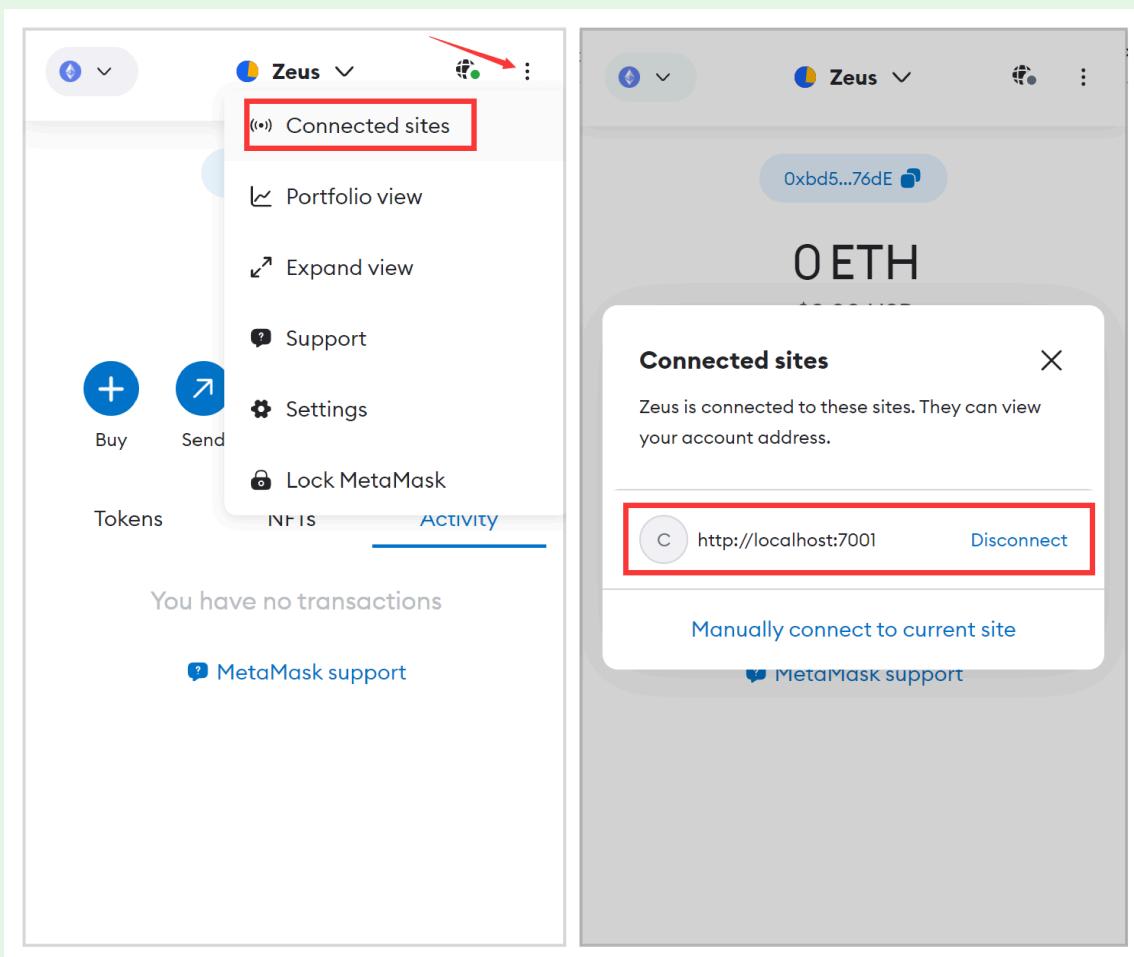
The screenshot shows a provider configuration interface. On the left, there's a sidebar with 'Providers' and an 'Add' button. Below it is a list of providers: 'provider\_storage\_minio\_s3', 'provider\_oauth\_lark', 'provider\_email\_qq', and 'metamask\_provider'. The 'metamask\_provider' entry is highlighted with a red rectangle. On the right, there's a main panel with columns for 'Category', 'Type', and various permissions like 'Can signup', 'Can signin', 'Can unlink', and 'Prompted'. The 'metamask\_provider' row has 'Web3' selected in the Type column. At the bottom, there are buttons for 'Copy SAML metadata URL' and 'Copy sign-in page URL'.

## Step3. Login with MetaMask

Now you can login with MetaMask. Here is a demo video.

 提示

1. When login with MetaMask, please authorize only one Ethereum address. Casdoor will only bind one Ethereum address per user.
2. If you want to switch to another Ethereum address for login, please disconnect the connection between current Ethereum address and Casdoor first.





&gt;

资源

# 资源

## 概述

上传资源到Casdoor

# 概述

您可以在 casdoor 上传资源。在上传资源之前，您需要配置一个存储提供商。请查看[存储提供商](#)

您现在至少配置了一个存储提供程序，并将该提供程序添加到您的应用程序中。

Providers [?](#) :

Name	Category	Type
Provider_azure	Storage	A
Github_1	OAuth	G
provider_Alipay	Payment	支

好了！让我们看看如何[上传](#) and [删除](#) 资源的例子。

## 上传资源

用户可以将文件和图像等资源上传到先前配置的[云存储](#)。

Home    Organizations    Users    Roles    Permissions    Providers    Applications    Resources    Tc

Resources [Upload a file...](#)

Provider	Created time	Tag
provider_storage_aliyun_oss	2022-05-18 17:25:21	custom
provider_storage_aliyun_oss	2022-05-18 12:28:01	custom
provider_storage_aliyun_oss	2022-05-17 16:25:39	custom

# 删除资源

如果你不再需要资源，你可以选择点击“删除”按钮来删除它。

	Created time	Tag	Type	Format	File size	Preview	URL	Action
	2022-05-19 23:16:55	custom	image	.jpg	70.3 KB		<button>Copy Link</button>	<button>Delete</button>



&gt;

产品

# 产品



产品

添加您想要销售的产品



支付

查看支付中产品的交易信息

# 产品

您可以添加您想要销售的产品 (或服务)。下面将告诉您如何添加产品。

## 配置产品属性

第一：您需要了解产品的基本属性

[标签](#)

[详情](#)

[货币](#)

[价格](#)

[数量](#)

[售价](#)

Tag [?](#) :

Casdoor Summit 2022

Detail [?](#) :

This is a description

Currency [?](#) :

USD

Price [?](#) :

19

Quantity [?](#) :

100

Sold [?](#) :

10

## 支付服务提供商

当然，除了设置这些属性外，您还需要为产品添加付款提供者。和多个付款提供者可以添加到产品。

了解如何配置付款提供商, 请参阅 [付款提供商](#)

Payment providers [provider\\_Alipay](#) [X](#)

?: [provider\\_Alipay](#)

Return URL ?: <http://localhost:8000/products/callback>

最后, 填写 **返回 URL**。当付款完成时, 这是从付款提供者页面跳转到的URL。

## 预览产品

搞定! 查看评论并保存:

Preview ?:

[Test buy page..](#)

Buy Product

Name	Product
Detail	This is a subscription.
Image	
Price	\$300 (USD)
Pay	<a href="#">Alipay</a>

# 支付

付款成功后，您可以在 **付款** 中看到产品的交易信息，组织、用户、购买时间、产品名称等

## 发票

您可以输入编辑界面来开具发票

Type	Product	Price	Curre	Action
	A notebook computer	300	USD	<button>Result</button> <button>Edit</button> <button>Delete</button>

填写发票信息。发票类型是 **个人** 和 **机构**。

Message ② :

Person name ② :

Person ID card ② :

Person Email ② :

Person phone ② :

Invoice type ② :

Invoice title ② :

Invoice tax ID ② :

Invoice remark ② :

Invoice URL ② :

Invoice actions ② : Issue Invoice Return to Website

最后，点击“发出发票”按钮。



&gt;

Pricing

# Pricing



## Overview

Casdoor pricing overview



## Plan

Casdoor plan overview



## Pricing

Casdoor pricing overview



## Subscription

Casdoor subscription overview

# Overview

Casdoor can be used as subscription management system via [plan](#), [pricing](#) and [subscription](#).

You can choose which plans to include in your price list like on pictures below:

**Pricing**

Casdoor + Casbin SaaS hosting services provided by Casbin Inc.

BASIC	PREMIUM	ENTERPRISE
<b>\$ 9</b> per month	<b>\$ 25</b> per month	<b>\$ 59</b> per month
For small teams, with limited technical support (via Tickets)	For fast growing start-ups, with full technical support (8x5)	For large & medium-sized enterprise, with full technical support (8x5)
<ul style="list-style-type: none"><li>✓ 10 Applications</li><li>✓ 10 Providers</li><li>✓ 100 login requests per second</li><li>✓ 10K Users</li><li>✓ 5 Organizations</li></ul>	<ul style="list-style-type: none"><li>✓ 100 Organizations</li><li>✓ 5K login requests per second</li><li>✓ 99.9% SLA</li><li>✓ Anti-bot security protection</li><li>✓ Custom domain</li></ul>	<ul style="list-style-type: none"><li>✓ 1K Organizations</li><li>✓ 300 Providers</li><li>✓ 99.9% SLA</li><li>✓ Anti-bot security protection</li><li>✓ Custom domain</li><li>✓ UNLIMITED login requests per second</li></ul>
<a href="#">Getting started</a>	<a href="#">Getting started</a>	<a href="#">Getting started</a>

*Free 14-days trial available!*

Each [pricing list](#) belong to [application](#) and has own url that allow sign up new users with selected plan.

General user flow looks like:

- Get [pricing](#) url
- Choose [plan](#)

- Signup to application
- **Subscription** will be auto created

# Plan

Plan - describe list of application's features with own name and price.

Plan features depends on Casdoor role with set of permissions.

That allow to describe plan's features independ on naming and price. For example: plan may has diffrent prices depends on county or date.

Picture below describes relation between Plan and role.

## Plan

- Display Name
  - Price per month
- ...

## Role

permission 1  
permission 2  
...  
permission N

# Plan properties

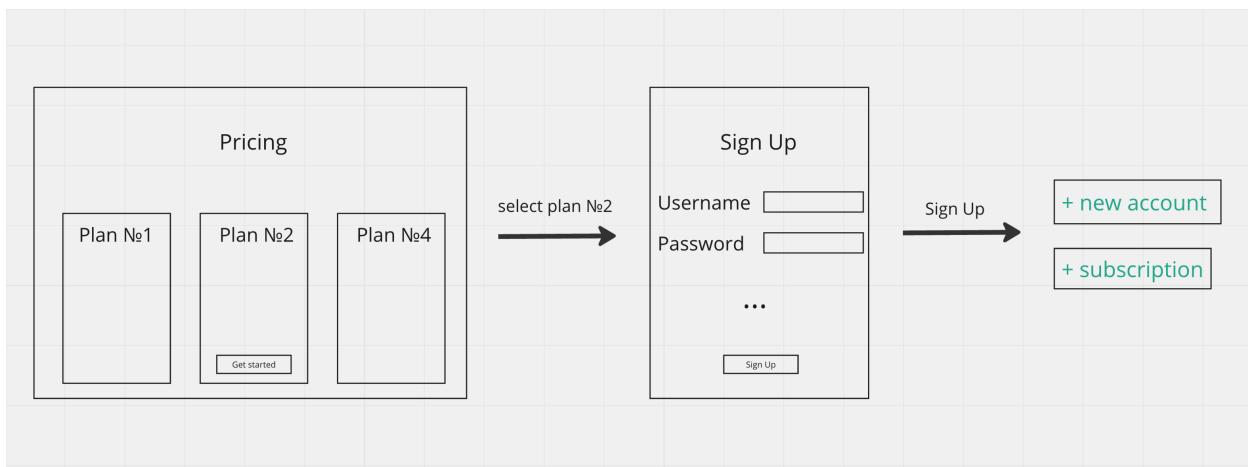
Every plan has these properties:

- Owner
- Name
- CreatedTime
- DisplayName
- IsEnabled
- PricePerMonth
- PricePerYear
- Role

# Pricing

Pricing contains one or more plan and provide ability to sing up to application with selected plan.

General flow might looks like on picture below:



## Pricing properties

Every Pricing has these properties:

- Owner
- Name
- CreatedTime
- DisplayName
- Description
- Plans - Array of plans
- IsEnabled

- `Has trial` - any payments will not be required for sign up in case of true
- `TrialDuration` - impact on subscription end days
- `Application`
- `Submitter`
- `Approver`
- `ApproveTime`
- `State`

# Subscription

**Subscription** - helps to manage user's selected plan that make easy to control application's features access.



Since each plan based on `casdoor role` you can assign plan's role to user and use enforce API for permission checking.

**Subscription** can be created in thee ways:

- Manually by admin
- After sign up from pricing page
- Via API.

## Subscription properties

Every Subscription has these properties:

- `Owner`
- `Name`
- `CreatedTime`
- `DisplayName`
- `Duration`
- `Description`
- `Plan`
- `StartDate`

- EndDate
- User
- IsEnabled
- Submitter
- Approver
- ApproveTime
- State



&gt;

用户

# 用户

## 概述

管理Casdoor的用户

## MFA / 2FA

Secure your account with MFA / 2FA

## 角色

用户角色

## 权限

用户权限

# 概述

## User properties

作为一个认证平台，Casdoor 能够管理用户。每个用户都有这些属性：

- `Owner` 用户的所有者组织
- `Name` 用户名，唯一的
- `CreatedTime` 创建时间
- `UpdatedTime`
- `Id` 每个用户的 Id 都是唯一的。
- `Type`
- `Password`
- `PasswordSalt`
- `PasswordOptions` Password complexity options
- `DisplayName` Shown in UI
- `FirstName`
- `LastName`
- `Avatar` A link to user's avatar
- `PermanentAvatar`
- `Email`
- `Phone`
- `Location`
- `Address`
- `Affiliation`

- `Title`
- `IdCardType`
- `IdCard`
- `Homepage`
- `Bio`
- `Tag`
- `Region`
- `Language`
- `Gender`
- `Birthday`
- `Education`
- `Score`
- `Karma`
- `Ranking`
- `IsDefaultAvatar`
- `IsOnline`
- `IsAdmin` Is the user the admin of his organization
- `IsGlobalAdmin` Does the user have the permission to manage the Casdoor
- `IsForbidden`
- `IsDeleted`
- `SignupApplication`
- `Hash`
- `PreHash`
- `CreatedIp`
- `LastSigninTime`
- `LastSigninIp`

- `Roles` Array of the user's roles
- `Permissions` Array of the user's permissions

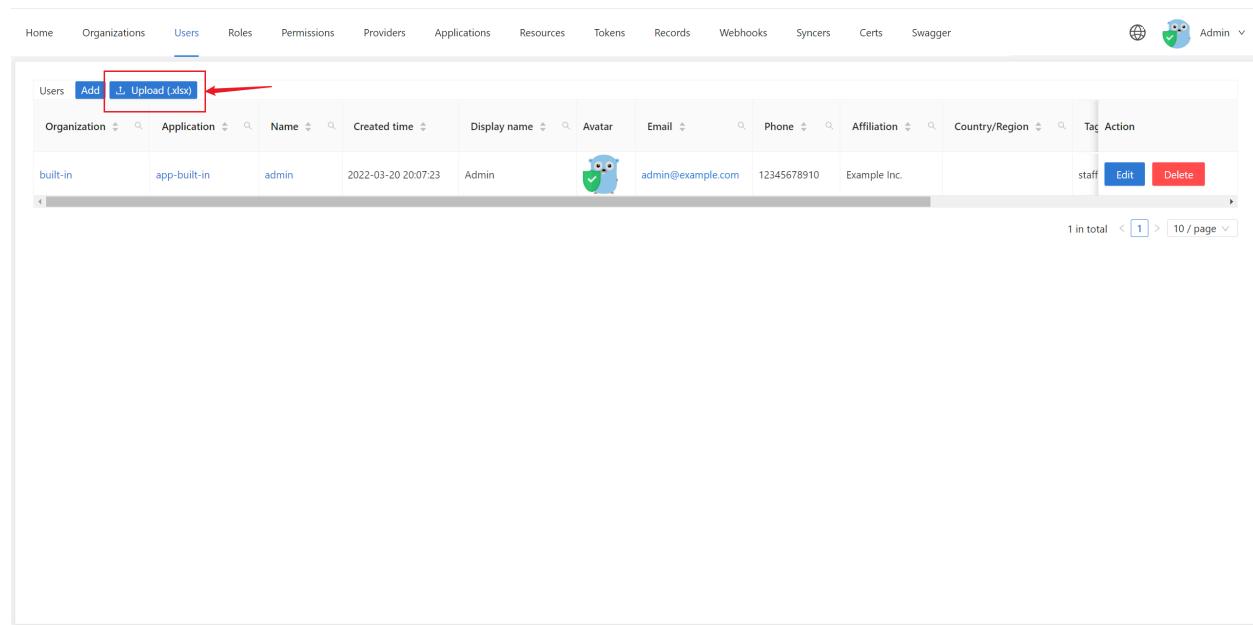
平台唯一ID:

- `Github`
- `Google`
- `QQ`
- `微信`
- `Facebook`
- `钉钉`
- `微博`
- `Gitee`
- `LinkedIn`
- `Wecom`
- `Lark`
- `Gitlab`
- `Adfs`
- `Baidu`
- `Casdoor`
- `Infoflow`
- `Apple`
- `AzureAD`
- `Slack`
- `Steam`
- `Ldap`
- `Properties` 这是一个字符串 → 字符串地图，存储所有其他属性。

# 从CSV文件导入用户

您可以通过上传用户信息的 XLSX 文件来添加新用户或更新现有的 Casdoor 用户。

在管理控制台中，转到用户并点击 **上传(.xlsx)** 按钮。



The screenshot shows the Casdoor user management interface. At the top, there is a navigation bar with links: Home, Organizations, Users, Roles, Permissions, Providers, Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, and Swagger. On the far right, there is a user icon labeled "Admin". Below the navigation bar, there is a search bar with placeholder text "Search users" and a "Clear" button. Underneath the search bar, there is a table with columns: Organization, Application, Name, Created time, Display name, Avatar, Email, Phone, Affiliation, Country/Region, Tag, and Action. A single user row is visible: built-in, app-built-in, admin, 2022-03-20 20:07:23, Admin, , admin@example.com, 12345678910, Example Inc., staff, Edit, Delete. At the bottom of the table, there is a pagination bar with the text "1 in total" and "10 / page". Above the table, there is a button labeled "Upload (.xlsx)" with a red arrow pointing to it. The entire interface is framed by a light gray border.

选择您的 XLSX 文件并点击 Open，用户将被导入。

我们提供了 [模板XLSX文件 user\\_test.xlsx](#) 在 `xlsx` 文件夹中。该模板包括5个用户测试和某些必需的用户属性的信头。

Home   Organizations   **Users**   Roles   Permissions   Providers   Applications   Users uploaded successfully, refreshing the page   Syncers   Certs   Swagger    Admin

**user\_test.xlsx**

Organization	Application	Name	Created time	Display name	Avatar	Email	Phone	Affiliation	Country/Region	Tag	Action
built-in	app-built-in	tesla	2022-03-20 20:49:03	Nikola Tesla		9v73hn@example.com	40738134827	Example Inc.	United States of America	sciencist	<button>Edit</button> <button>Delete</button>
built-in	app-built-in	gauss	2022-03-20 20:48:33	Carl Friedrich Gauss		vqdsan@example.com	98621482844	Example Inc.	Germany	mathematician	<button>Edit</button> <button>Delete</button>
built-in	app-built-in	galileleo	2022-03-20 20:47:58	Galileo Galilei		8p4f38@example.com	22596937332	Example Inc.	Italy	scientist	<button>Edit</button> <button>Delete</button>
built-in	app-built-in	euler	2022-03-20 20:47:08	Leonhard Euler		3dzw4@example.com	74409642681	Example Inc.	Switzerland	mathematician	<button>Edit</button> <button>Delete</button>
built-in	app-built-in	einstein	2022-03-20 20:46:29	Albert Einstein		z6mive@example.com	60062541396	Example Inc.	Germany	scientist	<button>Edit</button> <button>Delete</button>
built-in	app-built-in	admin	2022-03-20 20:07:23	Admin		admin@example.com	12345678910	Example Inc.		staff	<button>Edit</button> <button>Delete</button>

6 in total < 1 > | 10 / page ▾

Made with ❤ by **Casdoor**

# MFA / 2FA

## About multi-factor authentication

MFA (Multi-Factor Authentication) is a security measure that can improve the security of users and systems. It requires users to provide two or more factors of authentication to verify their identity when logging in or performing sensitive operations.

For Casdoor, the second form of authentication is a code that's sent as a text message or email. After you enable MFA, Casdoor generates an authentication code any time someone attempts to sign in your account. The only way someone can sign in your account is if they know both your password and have access to the authentication code.

## Config MFA

1. In user profile page, you can see the configuration of multi-factor authentication. If you can't see it, make sure the organization has added multi-factor authentication item in the account items table.

Managed accounts (1) Managed accounts [Add](#)

Application	Username	Password	Action
			No data

Multi-factor authentication (1) Multi-factor methods

Type : sms	<a href="#">Setup</a>
Type : email	<a href="#">Setup</a>

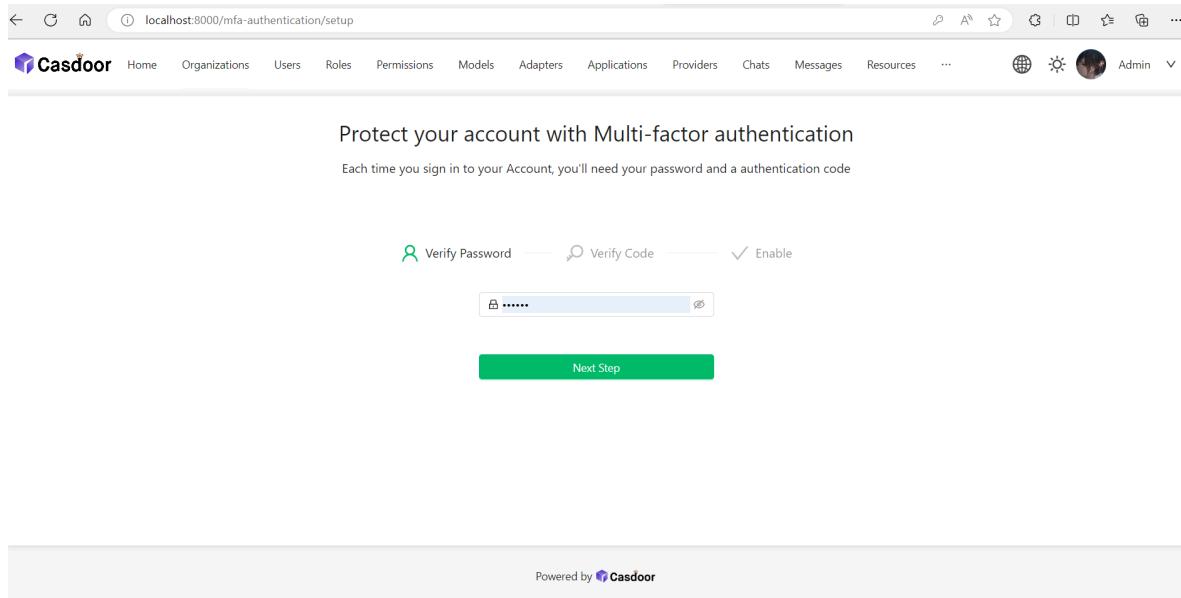
ID card (1) [Edit](#)

2. Click the "setup" button.

Multi-factor authentication (1) Multi-factor methods

Type : sms	<a href="#">Setup</a>
Type : email	<a href="#">Setup</a>

3. Type your password and click "Next Step".



## Configuring multi-factor authentication using a TOTP mobile app

A time-based one-time password (TOTP) application automatically generates an authentication code that changes after a certain period of time. We recommend using:

- [Google Authenticator](#)
- [Microsoft Authenticator](#).

### 💡 提示

To configure authentication via TOTP on multiple devices, during setup, scan the QR code using each device at the same time. If 2FA is already enabled, and you want to add another device, you must re-configure your TOTP app from user profile page.

---

## Protect your account with Multi-factor authentication

Each time you sign in to your Account, you'll need your password and a authentication code

 Verify Password —  Verify Code —  Enable



Scan the QR code with your authenticator app

Or copy the secret to your authenticator app

P757K7XT5MIO5RPZQYS0



 Passcode

Next Step

[Use email](#)    [Use SMS](#)

1. In "Verify Code" step, do one of the following:

- Scan the QR code with your mobile device's app. After scanning, the app displays a six-digit code that you can enter on Casdoor.
- If you can't scan the QR code, you can manually copy and enter the secret in your TOTP app instead.

2. The TOTP mobile application saves your account on Casdoor and generates a new authentication code every few seconds. On Casdoor, type the code into the field "Passcode" and click "Next Step".

3. Above "Enable" button, copy your recovery codes and save to your device. Save them to a secure location because your recovery codes can help you get

back into your account if you lose access.

## Protect your account with Multi-factor authentication

Each time you sign in to your Account, you'll need your password and a authentication code

 Verify Password —  Verify Code —  Enable

Please save this recovery code. Once your device cannot provide an authentication code, you can reset mfa authentication by this recovery code

ad30de29-3ce0-4e39-a97f-ceff1d503d3c

Enable

### 注意事项

One recovery code can only be used once. If you use a recovery code to sign in, it will be invalid.

## Configuring multi-factor authentication using text messages

If you have bound your mobile phone number, Casdoor will use it to send you a text message.

The screenshot shows the Casdoor interface for enabling multi-factor authentication. At the top, there's a navigation bar with links like Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Applications, Providers, Chats, Messages, and a dropdown for 'role\_test'. Below the navigation is a title 'Protect your account with Multi-factor authentication' and a subtitle 'Each time you sign in to your Account, you'll need your password and a authentication code'. There are two tabs: 'Verify Password' (selected) and 'Verify Code'. Under 'Verify Password', a red-bordered input field contains the phone number 'Your phone is 89748593889'. Below it is a 'Send Code' button. A 'Next Step' button is at the bottom, and a 'Use Email' link is also present.

If not, you need to bind your mobile phone number first.

This screenshot shows the same Casdoor interface, but the 'Verify Password' tab is now disabled, indicated by a greyed-out appearance. A message at the top says 'Please bind your phone first, the system automatically uses the phone for multi-factor authentication'. Below this, there's a country code selector set to '+86', a search icon, and a 'Phone' input field. A 'Send Code' button is available. The 'Next Step' button is visible at the bottom, along with the 'Use Email' link. The footer indicates the page is 'Powered by Casdoor'.

1. Select your country code and type your mobile phone number.
2. Check your information is correct, click "Send Code".
3. You'll receive a text message with a security code. Then type the code into the field "Enter your code" and click "Next Step".

4. Above "Enable" button, copy your recovery codes and save to your device. Save them to a secure location because your recovery codes can help you get back into your account if you lose access.

## Configuring multi-factor authentication using email

Config email as your multi-factor authentication method is similar to text messages.

1. Use your current email or type your email address and click "Send Code".
2. Then type the code into the field "Enter your code" and click "Next Step".
3. Above "Enable" button, copy your recovery codes and save to your device. Save them to a secure location because your recovery codes can help you get back into your account if you lose access.

## Changing your preferred MFA method

You can add multiple MFA methods. Only the preferred method will be used when you sign in.

If you want to set a preferred MFA method, click the "Set preferred" button.

The screenshot shows a user interface for managing multi-factor authentication methods. At the top, there is a header for "Multi-factor authentication" with a link to "Multi-factor methods". Below this, there are two entries:

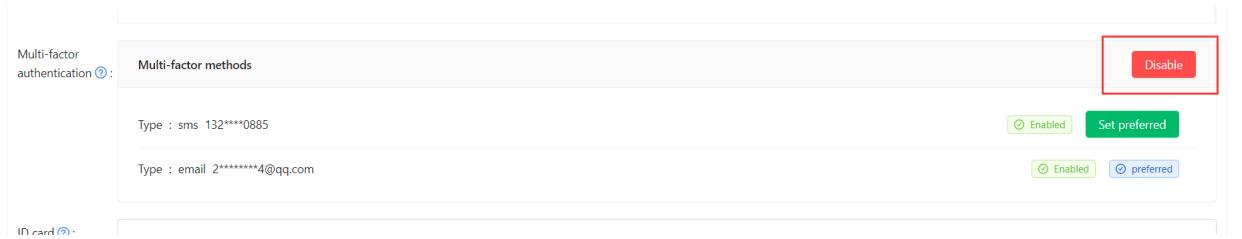
- Type : sms 132\*\*\*\*0885
- Type : email 2\*\*\*\*\*4@qq.com

For each entry, there are two buttons: "Enabled" (green) and "preferred" (blue). The "preferred" button for the email method is highlighted with a red box. At the bottom of the interface, there is a section labeled "ID card" with a placeholder box.

A "Preferred" label is displayed on your preferred method.

# Disable multi-factor authentication

If you want to disable multi-factor authentication, click the "Disable" button. All your multi-factor authentication config will be deleted.



# 角色

每个用户可能有多个角色。 您可以在用户的个人资料上看到用户的角色。

Bio [?](#) :

Tag [?](#) : staff

Signup application [?](#) : app-built-in

Roles [?](#) : [role\\_test](#) [role\\_test2](#)

Permissions [?](#) : [permission\\_test](#)

3rd-party logins [?](#) :

Is admin [?](#) :

Is global admin [?](#) :

Is forbidden [?](#) :

Is deleted [?](#) :

## 角色属性

每个角色都具有这些属性：

- 所有者
- 名称
- 创建时间
- 显示名称
- 已启用
- 用户 此角色子用户的数组
- 角色 此角色子角色的数组



# 权限

每个用户可能有多个权限。 您可以在用户个人资料上看到用户的权限。

Bio [?](#) :

Tag [?](#) : staff

Signup application [?](#) : app-built-in

Roles [?](#) : [role\\_test](#) [role\\_test2](#)

Permissions [?](#) : [permission\\_test](#)

3rd-party logins [?](#) :

Is admin [?](#) :

Is global admin [?](#) :

Is forbidden [?](#) :

Is deleted [?](#) :

## 权限属性

权限拥有这些属性

- 所有者
- 名称
- 创建时间
- 显示名称
- 已启用
- 模型
- 用户 此角色子用户的数组

- 角色 此角色子角色的数组
- 资源类型
- 资源 资源数组
- 操作 操作数组
- 效果



&gt;

同步器

# 同步器

## 概述

Casdoor同步用户

## 数据库

使用 Database 同步器同步数据库

## Keycloak

使用 Keycloak Syncer 同步 Keycloak

# 概述

作为一个认证平台，Casdoor 可以轻松地操纵存储在数据库中的用户。

## 同步器

Casdoor 在 `user` 表中存储用户。当您计划使用 Casdoor 作为认证平台时，请不要担心您的应用程序 `user` 数据迁移到 Casdoor。Casdoor 提供 **同步器** 来帮助您快速同步用户数据到 Casdoor。

指定您想要同步到 Casdoor 的数据库和用户表。同步器将在指定的间隔后同步数据。欲了解详情，请参阅 [数据库同步器](#)。

## 同步哈希值

Casdoor 使用哈希来确定如何更新用户。Casdoor 将计算表中每个用户的散列值。它是通过用户信息比如用户的密码或手机号码来生成的。

如果特定 `Id` 的用户计算的散列值与原始值相比有所改变，Casdoor 会确认哪个用户表已更新。然后，数据库将更新旧信息，实现 Casdoor 用户表和原始用户表之间的 **双向同步**。

# 数据库

## 数据库同步器

我们作为演示创建的用户表从 [模板 XLSX 文件](#) 导入。

owner	name	created_time	updated_time	id	type	password	password_salt	display_name	first_name	last_name	avatar	permanent_avatar	email
built-in	einstein	2022-03-20T20:46:29+08:00		1c57cc37-37f5-4def-9e9f-082189ef63d2	normal-user	123		Albert Einstein			<a href="https://casbin.org">https://casbin.org</a>		z6mive@
built-in	euler	2022-03-20T20:47:08+08:00		bb7831b4-0d24-4e96-b043-f8fd8d15eb	normal-user	123		Leonhard Euler			<a href="https://casbin.org">https://casbin.org</a>		3dzw4j@
built-in	galileo	2022-03-20T20:47:58+08:00		7920eb6c-f9f5-40ef-8e18-3ac99f49bd15	normal-user	123		Galileo Galilei			<a href="https://casbin.org">https://casbin.org</a>		8p4f38@
built-in	gauss	2022-03-20T20:48:33+08:00		f0c28816-2c0d-479b-b545-cb4cf96db36	normal-user	123		Carl Friedrich Gauß			<a href="https://casbin.org">https://casbin.org</a>		vqdsan@
built-in	tesla	2022-03-20T20:49:03+08:00		687c3068-fd21-4d32-b2ba-e13e8b369a	normal-user	123		Nikola Tesla			<a href="https://casbin.org">https://casbin.org</a>		9v73hn@

点击 **Syncers** 标签页并创建一个新的同步器。 填写下面所需的所有信息并保存。

Organization [?](#): built-in

Name [?](#): mysql test

Type [?](#): Database

Host [?](#): localhost

Port [?](#): 3306

User [?](#): root

Password [?](#): root

Database type [?](#): MySQL

Database [?](#): casdoorevdev

Table [?](#): user

Table primary key [?](#):

Table columns [?](#):

Table columns	Add	Column name	Column type	Casdoor column	Is hashed	Action		
		name	string	Name	<input checked="" type="checkbox"/>			
		create_time	string	CreatedTime	<input checked="" type="checkbox"/>			
		id	string	Id	<input checked="" type="checkbox"/>			
		password	string	Password	<input checked="" type="checkbox"/>			

Affiliation table [?](#):

Avatar base URL [?](#): <https://casbin.org>

Sync interval [?](#): 1000



提示

一般而言，您至少需要填写Casdoor列中的 `ID` 和 `Name` 以及其他重要信息，如 `createdTime`, `Password`, `DisplayName`.

以下是必填字段。

- `Organization`: The organization that the user will import
- `Name`: 同步器名称
- `Type`: 选择数据库
- `Host`: 原始数据库主机地址
- `Port`: 原始数据库端口
- `User`: 原始数据库用户名
- `Password`: 原始数据库密码
- `Database type`: 所有 Xorm 支持的数据库，例如: MySQL, PostgreSQL, SQL Server, Oracle, Sqlite
- `Database`: 原始数据库名称
- `Table`: The original user table name
- 表格列
- `Column name`: The original user column name
- `Column type`: The original user column type
- `Casdoor Column`: The casdoor user column name

可选项

- `Is hashed`: 是否计算哈希值. When enable "Is hashed", if the field of user in origin table updated, the syncer will sync this user. Disable "Is hashed", meaning if only the field update, the syncer need not sync the user. In short, the user does not synchronize until the fields involved in the hash calculation(enable "Is hashed") are updated.
- `Avatar base URL`: When sync users, if `Avatar base URL` is not empty and

origin user.avatar not hasPrefix "http", new user.avatar will be replaced by Avatar base URL + user.avatar.

- **Affiliation table**: It is used to sync the affiliation of user from this table in database. Because the affiliation may be code of int type in "Affiliation table", so we need to map the int to a string. See [getAffiliationMap\(\)](#) . Because Casdoor has some redundant fields to borrow, [here](#) we use `score` to map the int code to a string name.

然后您可以打开 **启用** 按钮并保存，同步器将开始工作。

Users												Action
Organization	Name	Created time	Display name	Avatar	Email	Phone	Affiliation	Country/Region				
built-in	euler	2022-04-08 10:10:45							<button>Edit</button>	<button>Delete</button>		
built-in	gauss	2022-04-08 10:10:45							<button>Edit</button>	<button>Delete</button>		
built-in	tesla	2022-04-08 10:10:45							<button>Edit</button>	<button>Delete</button>		
built-in	einstein	2022-04-08 10:10:45							<button>Edit</button>	<button>Delete</button>		
built-in	galileo	2022-04-08 10:10:45							<button>Edit</button>	<button>Delete</button>		

# Keycloak

## Keycloak 同步器

Keycloak除了能自动配置 Table 和 Table columns 外，其他与 [数据库同步器](#) 基本相同。

此外，Keycloak 同步器将会查询 credential 表，keycloak\_group 表和 user\_group\_membership 表，因为Keycloak 中的用户信息储存在多个表中。

The screenshot shows the configuration page for a Keycloak syncer. At the top, there are fields for Organization (built-in), Name (keycloak), Type (Keycloak), Host (localhost), Port (3306), User (root), Password (root), Database type (MySQL), and Database (keycloak). A red box highlights the 'Table' field containing 'user\_entity'. An arrow points from this field to a note: 'configured automatically after selecting Keycloak as syncer type'. Below this, the 'Table primary key' field is empty. At the bottom, a large red box encloses the 'Table columns' section, which lists various user attributes like ID, USERNAME, EMAIL, etc., with their corresponding column types and Casdoor column mappings.

Column name	Column type	Casdoor column	Is hashed	Action
ID	string	Id	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
USERNAME	string	Name	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
EMAIL	string	DisplayName	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
EMAIL_VERIFIED	boolean	Email	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
FIRST_NAME	string	EmailVerified	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
LAST_NAME	string	FirstName	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
CREATED_TIMESTAMP	string	LastName	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
ENABLED	boolean	CreatedTime	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>
		IsForbidden	<input checked="" type="checkbox"/>	<span>Up</span> <span>Down</span> <span>Delete</span>



&gt;

令牌

# 令牌



## 概述

Casdoor令牌简介

# 概述

Casdoor 基于 OAuth。 Tokens 是用户的 OAuth 的 token。

- Owner
- Name
- CreatedTime
- Application
- Organization
- User
- Code
- AccessToken
- ExpireIn 令牌将在数小时后过期
- Scope 授权范围
- TokenType 例如 输入 Bear

There are two options to generate a JWT Token after logging into the application:

- JWT
- JWT-Empty

The JWT option will create a token with all User fields. The JWT-Empty will create a token with all non-empty values for the user.



&gt;

Webhooks

# Webhooks



## 概述

在Casdoor添加webhooks

# 概述

## 概述

事件系统允许您构建集成，订阅 Casdoor 上的某些事件。当其中一个事件被触发时，我们将向配置的 URL 发送一个以 json 为主体的 POST 请求。应用程序解析 json 并执行 hook。事件包括注册、登录、注销、更新用户，这些用户存储在记录的操作字段中。事件系统可以用来更新用户的外部问题。



&gt;

部署

# 部署



## Nginx

使用Nginx 来反转代理您的后端Go 程序，快速启动Casdoor 服务



## k8s

在 k8s 中部署 Casdoor

# Nginx

虽然 Casdoor 是一个前后端分离的架构，但在生产环境中，后端程序仍然为前端文件提供静态文件服务。因此，您可以使用反向代理软件，如 [Nginx](#) 来代理 Casdoor 域的所有流量，并将其重定向到后端的端口。

在本章中，您将学习如何使用 Nginx 来反向代理您的后端 Go 程序，快速启动 Casdoor 服务。

## 1. 构建前端静态文件

现在假设您已经下载了 Casdoor 并完成了必要的配置。如果没有，请回到 [开始](#) 部分。

您只需要构建静态文件，例如：

[Yarn](#)    [npm](#)

```
yarn install && yarn run build
```

```
npm install && npm run build
```

## 2. 运行后端程序

```
go run main.go
```

或者先构建:

```
go build && ./main
```

## 3. 配置和运行 Nginx

```
vim /path/to/nginx/nginx.conf
```

添加服务器:

```
server {
    listen 80;
    server_name YOUR_DOMAIN_NAME;
    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_redirect off;
        proxy_pass http://127.0.0.1:8000;
    }
}
```

然后重启您的 nginx 进程, 运行:

```
nginx -s reload
```

## 4. 测试

在您最喜欢的浏览器中访问 [http://YOUR\\_DOMAIN\\_NAME](http://YOUR_DOMAIN_NAME)



# k8s

## 在 k8s 中部署 Casdoor

我们已经给出了一个将 Casdoor 部署到 k8s 中的基本示例。在 casdoor 的根目录下，有一个名为“k8s.yaml”的文件，里面包含了在 k8s 中部署 casdoor 时使用的示例最低配置、一个部署和一个服务。

首先，确保你已经修改了conf/app.conf，使 casdoor 能够成功连接数据库，并且数据库正在运行。其次，确保 k8s 能够拉取必要的镜像。

运行

```
kubectl apply -f k8s.yaml
```

很快你就可以通过命令 `kubectl get pods` 看到结果

k8s.yaml 的内容如下

```
# this is only an EXAMPLE of deploying casddor in kubernetes
# please modify this file according to your requirements
apiVersion: v1
kind: Service
metadata:
  #EDIT IT: if you don't want to run casdoor in default namespace,
  #please modify this field
  #namespace: casdoor
  name: casdoor-svc
  labels:
    app: casdoor
```

该文件只是一个示例。 比如您可以选择使用 default 以外的命名空间，使用服务类型而不是 nodeport 来暴露 casdoor，或者在 k8s 中使用 use config map 来挂载配置文件，这是 k8s 中比较推荐的方式。





LDAP

# LDAP

## 概述

Casdoor 与ldap服务器合作

## Config and Sync LDAP Users

Casdoor LDAP 配置

## LDAP 服务器

在Casdoor中连接LDAP服务器

# 概述

支持当前ldap服务器已被引入到Casdoor。 Casdoor 能够同步用户从ldap服务器到Casdoor使用它们作为用户帐户登录。 并使用 ldap服务器验证它们。 Casdoor还支持设置cron作业， 以定期自动同步用户。

## Casdoor-Ldap 同步机制详情

How Casdoor cooperates with a ldap server is described as follows:

1. Synchronization: Casdoor can connect to ldap server fetch users' information, reads all users' information (include `uidNumber`, `uid`, `cn`, `gidNumber`, `mail`, `email`, `emailAddress`, `telephoneNumber`, `mobile`, `mobileTelephoneNumber`, `registeredAddress`, `postalAddress`), and creates Casdoor accounts for each user in ldap, and stores the accounts in database.
2. Authentication: As we have seen, Casdoor doesn't fetch ldap users' passwords to Casdoor. Thus, when the account which is synchronized from ldap server tries to log in through Casdoor, instead of checking password stored in casdoor's database, Casdoor connects to ldap server and verifies whether the user's password is correct.
3. User distinguished: Casdoor uses `uid` to exclusively identify a user, thus please make sure every ldap user has a different uid.

Once a user is synchronized into Casdoor, that user's information is detached with the ldap user, which means, if you modify the user's information in Casdoor, the user's information in ldap won't be modified and vice versa (except ldap user's password, we rely on it to authenticate a user)



# Config and Sync LDAP Users

Ldap 配置属于一个组织，该组织的用户将被同步。

您应该使用全局管理员用户来修改配置。您需要在“组织”页面中输入LDAP用户同步的以下信息。

LDAP :		添加							
LDAP 服务器	服务器	基本DN	自动同步	最近同步	操作				
Example LDAP Server	example.com:389	ou=People,dc=example,dc=com	Disable			<a href="#">同步</a>	<a href="#">编辑</a>	<a href="#">删除</a>	

# Config to connect LDAP server

Edit LDAP    Save    **Save & Exit**    Sync LDAP

Organization ② :	built-in
ID ② :	691edec0-f1ab-4e23-8f9f-a824a383032f
Server name ② :	Example LDAP Server
Server host ② :	example.com
Server port ② :	389
Enable SSL ② :	<input checked="" type="checkbox"/>
Base DN ② :	ou=built-in,dc=example,dc=com
Search Filter ② :	(objectClass=posixAccount)
Filter fields ② :	<input type="checkbox"/> uid <input type="checkbox"/> Email
Admin ② :	cn=admin,dc=example,dc=com
Admin Password ② :	.....
Auto Sync ② :	0 mins

## Server Name

A friendly name is used by managers to identify different servers.

例如： LDAP 服务器示例

## Server Host

LDAP server's host or IP address.

例如: `example.com`

## Server Port

LDAP server's ports, only allow numbers.

例如: `389`

## Base DN

Casdoor uses Sub search mode by default when searching in LDAP. Base DN is the basic distinguished name of the search. Casdoor will return all users under the current base DN.

The admin account configured in casdoor should have at least read-only permissions at base DN.

例如: `ou=Example,dc=example,dc=com`

## Search filter

Casdoor uses search filter to query ldap users.

e.g: `(objectClass=posixAccount)`

## Filter fields

Filter fields are the identifier of the user in LDAP server, When you log in to Casdoor as an LDAP user. Casdoor regards the entered login username as the `uid` of LDAP user. You can also config other filed, such as `mail`, `mobile`.

The screenshot shows the Casdoor web application interface. At the top, there is a navigation bar with links like Home, Organizations, Users, Roles, Permissions, Models, Adapters, Applications, Providers, Chats, Messages, Resources, and Admin. Below the navigation bar, the main content area is titled "Edit LDAP". It contains several input fields and a save button:

- Organization: built-in
- ID: 691edec0-f1ab-4e23-8f9f-a824a383032f
- Server name: Example LDAP Server
- Server host: 1
- Server port: 389
- Enable SSL: (checkbox)
- Base DN: ou=built-in,dc=example,dc=com
- Search Filter: (objectClass=inetOrgPerson)
- Filter fields: (empty)
- Admin: cn=admin,dc=example,dc=com
- Admin Password: (redacted)

At the bottom right of the form, there are "Save" and "Sync LDAP" buttons.

## Admin

An account that can log in to the specified LDAP server.

Login with DN or ID depends on the LDAP server settings you want to connect.

e.g: cn=manager, dc=example, dc=com

## Admin Password

Password of LDAP server Admin account.

## Auto Sync

Set 0 to disable auto sync, other value means Sync every few minutes.

# Sync users

The sync table displays all users get from the LDAP server in the specific `ou`. If the users have been synced, the checkbox will be disabled. You can check the box to select users, then sync the selected users from the LDAP server.

Example LDAP Server								
	CN	UidNumber / Uid	Group Id	Email	Phone	Address		
<input type="checkbox"/>	zhan san	1000 / zsan	500					
<input type="checkbox"/>	li si	1001 / lsi	500					
<input type="checkbox"/>	a dmin	1002 / admin	500					
<input type="checkbox"/>	tom brown	1007 / jery	500					
<input type="checkbox"/>	wrie jerry	1003 / wjerry	500					
<input type="checkbox"/>	admin2	1004 / admin2	500					
<input type="checkbox"/>	yyyy	1005 / yyyy	500					

< 1 > 10 / page ▾

## ⚠ 注意事项

If the `uid` of the user in LDAP server is same as the `name` of a user existed in the organization of Casdoor, Casdoor will create a new user that the `name` include the `uid` and a random string. But the user may not be able to login, because the name of the new synced user does not exist in LDAP server. So try to avoid that.

# LDAP 服务器

许多系统，例如 [Nexus](#) 支持 [ldap](#) 身份验证。在Casdoor中，同样实现了一个简单的LDAP服务器，支持bind与search操作。

下面来介绍如何连接Casdoor中的LDAP服务器并实现简单的登录认证。

## LDAP 服务器端口号

默认情况下，LDAP服务器监听端口 [389](#)，您可以通过更改 [conf/app](#) 中的 [ldapServerPort](#) 来更改默认端口号。

## 工作原理

类似于Casdoor的ldap客户端，ldap服务器中的用户都是 [posixAccount](#) 的子类。

当服务器收到符合LDAP协议传输的一组数据时，它将解析 [cn](#) and [ou](#)，其中 [cn](#) 表示用户名，[ou](#) 表示组织名称。[dc](#) 是什么并不重要。

如果是 [bind](#) 操作，服务器将使用 Casdoor 验证用户名和密码并给予用户权限。

如果它是一个 [search](#) 操作，服务器将根据 [bind](#) 赋予客户端的权限，检查 [search](#) 是否合法，并返回相应响应。

### ① 信息

我们只支持 [Simple Authentication](#)。

## Bind

在Catdoor ldapserver中，我们识别类似于这样的 DN : cn=admin,ou=buildt-in,dc=example,dc=com

所以请将管理员用户的 DN 设置为上面的形式。然后您可以使用此 DN 绑定到 ldap 服务器，使用 Casdoor 用户的密码登录以进行验证。如果服务器验证通过，用户将被授予 Casdoor 中的权限。

## Search

bind 操作成功完成后，您可以执行 search 操作。search 与 bind 在细节上有一些区别。

- 如果您想要搜索特定的用户，比如 built-in 组织下的 Alice，您应该使用这样的 DN : ou=buildt-in,dc=example,c=com，并在过滤器字段中添加 cn=Alice
- 如果您想要在某个组织下搜索所有用户，如 built-in 下的所有用户，您应该使用这样的 DN : ou=built-in,dc=example,dc=com，并在过滤器字段中添加 cn=\*
- 如果您想要搜索所有组织的所有用户(前提是用户有足够的权限)，您应该使用这样的 DN : ou=\*,dc=example,dc=com，并在过滤器字段中添加 cn=\*。



&gt;

集成

# 集成



C++

1 个项目



Go

8 个项目



Java

17 个项目



JavaScript

1 个项目



Lua

1 个项目



PHP

4 个项目



Ruby

1 个项目



Haskell

1 个项目



Python

1 个项目



> 集成 >

C++

# C++



Nginx

Using Casdoor in Nginx

# Nginx

Enable OpenID Connect-based single-sign for applications proxied by NGINX Plus, using Casdoor as the identity provider (IdP).

This guide explains how to enable single sign-on (SSO) for applications being proxied by NGINX Plus. The solution uses OpenID Connect as the authentication mechanism, with [Casdoor](#) as the identity provider (IdP), and NGINX Plus as the relying party.

See Also: You can find more information about the NGINX Plus OpenID Connect integration in the project's GitHub repo.

## Prerequisites

The instructions assume you have the following:

- A running Casdoor server. See the Casdoor documentation for [Server Installation](#) and [Try with Docker](#).
- An NGINX Plus subscription and NGINX Plus R15 or later. For installation instructions, see the [NGINX Plus Admin Guide](#).
- The [NGINX JavaScript module](#) (njs), required for handling the interaction between NGINX Plus and the IdP. After installing NGINX Plus, install the module with the command for your operating system.

For Debian and Ubuntu:

```
sudo apt install nginx-plus-module-njs
```

For CentOS, RHEL, and Oracle Linux:

```
sudo yum install nginx-plus-module-njs
```

- The following directive included in the top-level (“main”) configuration context in `/etc/nginx/nginx.conf`, to load the NGINX JavaScript module:

```
load_module modules/ngx_http_js_module.so;
```

## Configuring Casdoor

Note: The following procedure reflects the Casdoor GUI at the time of publication, but the GUI is subject to change. Use this guide as a reference and adapt to the current Casdoor GUI as necessary.

Create a Casdoor client for NGINX Plus in the Casdoor GUI:

1. Log in to your Casdoor account <http://your-casdoor-url.com/login/> .
2. In the top navigation column, click Application. On the Application page that opens, click the Add button in the upper left corner.



3. On the Edit Application page that opens, change the value in the Name and Display name to the name of the application for which you’re enabling SSO. Here we’re using NGINX Plus.

Name  : **NGINX Plus**

Display name  : **NGINX Plus**

In the Redirect URLs field, type the URL of the NGINX Plus instance including the port number, and ending in `/_codexch` (in this guide it is [https://your-site-url.com:443/\\_codexch](https://your-site-url.com:443/_codexch)).

Redirect URLs  : **Redirect URLs** 

**Redirect URL**

 [https://my-nginx.example.com:443/\\_codexch](https://my-nginx.example.com:443/_codexch)

Notes:

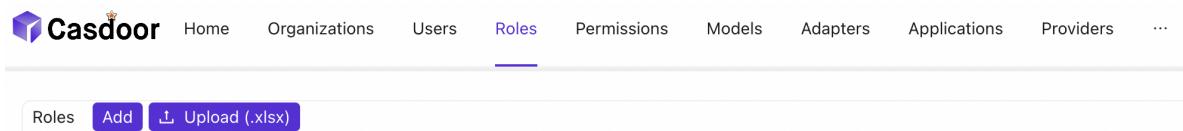
- For production, we strongly recommend that you use SSL/TLS (port 443).
- The port number is mandatory even when you’re using the default port for HTTP (80) or HTTPS (443).

4. Record the values in the Client ID and Client Secret fields. You will copy it into the NGINX Plus configuration file in [Step 4 of Configuring NGINX Plus](#).

Client ID  : 200c96d5ce5f11111111111111111111

Client secret  : 58f13a80b877e7e7e7e7e7e7e7e7e7e7e

5. Click Roles in the top navigation column, then click the Add button in the upper left corner of the page that opens.



6. On the Add page that opens, type a value in the Name and Display Name field (here it is nginx-casdoor-role) and click the Save button.

Name  : nginx-casdoor-role

Display name  : nginx-casdoor-role

7. In the top navigation column, click Users. On the Users page that opens, either click Edit to edit one of the existing users, or click the Add button in the upper left corner to create a new user.
8. On the Add page that opens, modify Name and Display Name you like (here it is user1).

Name  :

user1

Display name

user1

 :

Select NGINX Plus in Signup application.

Signup  
application  :

NGINX Plus

In the Managed accounts field, select NGINX Plus in Application and fill in the username and password.

Managed accounts  :	Managed accounts	Add
Application	Username	Password
NGINX Plus	<input type="text"/>	<input type="password"/>

9. Go back to the Roles page and click Edit on the nginx-casdoor-role row. In the opened page, in the Sub users field, select the username you just created(here it is built-in/user1)

Sub users [?](#) : built-in/user1

# Configuring NGINX Plus

Configure NGINX Plus as the OpenID Connect relying party:

1. Create a clone of the [nginx-openid-connect](#) GitHub repository.

```
git clone https://github.com/nginxinc/nginx-openid-connect
```

2. Copy these files from the clone to `/etc/nginx/conf.d`:

- `frontend.conf`
- `openid_connect.js`
- `openid_connect.server_conf`
- `openid_connect_configuration.conf`

3. Get the URLs for the authorization endpoint, token endpoint, and JSON Web Key (JWK) file from the Casdoor configuration. Run the following `curl` command in a terminal, piping the output to the indicated `python` command to output the entire configuration in an easily readable format. We've abridged the output to show only the relevant fields.

```
curl http://<casdoor-server-address>/.well-known/openid-configuration | python -m json.tool  
...
```

4. Using your preferred text editor, open `/etc/nginx/conf.d/openid_connect_configuration.conf`. Change the “default” parameter value of each of the following `map` directives to the specified value:

- `map $host $oidc_authz_endpoint` – Value of `authorization_endpoint` from [Step 3](#) (in this guide, `https://<casdoor-server-address>/login/oauth/authorize`)
- `map $host $oidc_token_endpoint` – Value of `token_endpoint` from [Step 3](#) (in this guide, `http://<casdoor-server-address>/api/login/oauth/access_token`)
- `map $host $oidc_client` – Value in the Client ID field from [Step 4 of Configuring Casdoor](#)
- `map $host $oidc_client_secret` – Value in the Client Secret field from [Step 2 of Configuring Casdoor](#)
- `map $host $oidc_hmac_key` – A unique, long, and secure phrase

5. Configure the JWK file. The procedure depends on which version of NGINX Plus you are using.

- In NGINX Plus R17 and later, NGINX Plus can read the JWK file directly from the URL reported as `jwks_uri` in [Step 3](#). Change `/etc/nginx/conf.d/frontend.conf` as follows:
  - a. Comment out (or remove) the `auth_jwt_key_file` directive.
  - b. Uncomment the `auth_jwt_key_request` directive. (Its parameter, `_jwks_uri`, refers to the value of the `$oidc_jwt_keyfile` variable, which you set in the next step.)
  - c. Change the “default” parameter of the `map $host $oidc_jwt_keyfile` directive to the value reported in the `jwks_uri` field in [Step 3](#) (in this guide, `http://<casdoor-server-address>/.well-known/jwks`).

- In NGINX Plus R16 and earlier, the JWK file must be on the local disk. (You can also use this method with NGINX Plus R17 and later if you wish.)
    - a. Copy the JSON contents from the JWK file named in the `jwks_uri` field in [Step 3](#) (in this guide, `http://<casdoor-server-address>/.well-known/jwks`) to a local file (for example, `/etc/nginx/my_casdoor_jwk.json`).
    - b. In `/etc/nginx/conf.d/openid_connect_configuration.conf`, change the “default” parameter of the `map $host $oidc_jwt_keyfile` directive to the local file path.
6. Confirm that the user named by the `user` directive in the NGINX Plus configuration (in `/etc/nginx/nginx.conf` by convention) has read permission on the JWK file.

## Testing

In a browser, enter the address of your NGINX Plus instance and try to log in using the credentials of a user mapped to the role for NGINX Plus.



# Casdoor



username, Email or phone



Password



Auto sign in

[Forgot password?](#)

Sign In

No account? [sign up now](#)

## Troubleshooting

See the [Troubleshooting](#) section at the nginx-openid-connect repository on GitHub.

# Go

## Kubernetes

Using Casdoor for authentication in Kubernetes

## OpenShift

Using Casdoor for authentication in openshift

## BookStack

在 BookStack 中使用 Casdoor 进行身份验证

## Bytebase

Using OAuth2 to connect various applications, like Bytebase

 **ELK**

Casdoor/elk-auth-casdoor概览

 **Gitea**

在 Gitea 中使用 Casdoor 进行身份验证

 **Grafana**

在 Grafana 中使用 Casdoor 进行身份验证

 **MinIO**

配置 Casdoor 作为身份提供者与 MinIO 支持

# Kubernetes

According to the [Kubernetes documentation](#), the API Server of Kubernetes can be authenticated using OpenID Connect (OIDC). This article will guide you on how to configure authentication in Kubernetes using Casdoor.

## Environment Requirements

Before starting, please make sure that you have the following environment:

- A Kubernetes cluster.
- A Casdoor application like this [demo website](#)
- kubectl command tool (optional)

 备注

Kubernetes oidc-issuer-url only accepts URLs which use the https:// prefix.  
So your Casdoor application should be deployed on an HTTPS website.

## Step 1: Creating an Casdoor App and User Account for Authentication

Go to your Casdoor and add your new application Kubernetes. Please remember the `Name`, `Organization`, `client ID`, `client Secret` and add some grant type in this APP.

Name [?](#):  Kubernetes

Display name [?](#):

Logo [?](#):  URL [?](#):  https://cdn.casbin.org/img/casdoor-logo\_1185x256.png

Preview:  Casdoor

Home [?](#):

Description [?](#):

Organization [?](#):  casbin

Client ID [?](#):  Kubernetes

Client secret [?](#):  72c65c3912aec24a9f3ec41b65a7577114ed2bae

Cert [?](#):

Grant types [?](#): Authorization Code Password ID Token Refresh Token Client Credentials Token

Next, we will add a new user to the application that we just created. Please note that the `organization` and `Signup application` used here should correspond to the APP registered earlier.

Organization <a href="#">?</a> :	casbin
ID <a href="#">?</a> :	202e02e9-9128-496a-a209-fdb336448f56
Name <a href="#">?</a> :	user_pnvm5i
Display name <a href="#">?</a> :	New User - pnvm5i
Avatar <a href="#">?</a> :	<p>Preview:</p>  <p>Upload a photo...</p>
User type <a href="#">?</a> :	normal-user
Password <a href="#">?</a> :	<a href="#">Modify password...</a>
Email <a href="#">?</a> :	pnvm5i@example.com
Phone <a href="#">?</a> :	+1 <input type="button" value="▼"/> 78005961394
Country/Region <a href="#">?</a> :	Please select country/region
Location <a href="#">?</a> :	
Affiliation <a href="#">?</a> :	Example Inc.
Title <a href="#">?</a> :	
Homepage <a href="#">?</a> :	
Bio <a href="#">?</a> :	
Tag <a href="#">?</a> :	staff
Signup application <a href="#">?</a> :	Kubernetes

# Step 2: Configure Kubernetes API Server with OIDC Authentication

To enable the OIDC plugin, at least configure the following flags on the API server:

- `--oidc-issuer-url` : URL of the provider which allows the API server to discover public signing keys.
- `--oidc-client-id` : A client id that all tokens must be issued for.

This article use of minikube for demonstration. We can configure the OIDC plugin for the minikube's API server using the following command at startup:

```
minikube start --extra-config=apiserver.oidc-issuer-
url=https://demo.casdoor.com --extra-config=apiserver.oidc-client-
id=294b09fbc17f95daf2fe
```

# Step 3: Test OIDC Authentication

## Obtain Authentication Information

Due to the lack of a frontend in kubectl, authentication can be performed by sending a POST request to the Casdoor server. Here is the code in Python which sends a POST request to the Casdoor server and retrieves the `id_token` and `refresh_token`:

```
import requests
import json
```

After executing this code, you should receive a response similar to the following:

```
{  
  "access_token": "xxx",  
  "id_token": "yyy",  
  "refresh_token": "zzz",  
  "token_type": "Bearer",  
  "expires_in": 72000,  
  "scope": ""  
}
```

Now, we can use the `id_token` that we just obtained to authenticate with Kubernetes API server.

## HTTP Request-Based Authentication

Add the token to the request header.

```
curl https://www.xxx.com -k -H "Authorization: Bearer $(id_token)"
```

- <https://www.xxx.com> is the Kubernetes API server deployment address.

## Kubectl Client-Based Authentication

### Configuration File Method

Write the following configuration to the `~/.kube/config` file. You should replace each configuration item in the configuration file above with the values you obtained earlier.

```
users:
```

Now you can directly access your API server using kubectl. Try running a test command.

```
kubectl cluster-info
```

### Command Line Argument Method

Alternatively, you can authenticate by directly adding the id\_token to the command line parameters of kubectl.

```
kubectl --token=$(id_token) cluster-info
```

# OpenShift

OpenShift supports OIDC, so we can integrate Casdoor with OpenShift. The following steps demonstrate how to integrate Casdoor with OpenShift Local using [the online demo of Casdoor](#).

## Step1. Create an Casdoor application

Add a new application in Casdoor, note following points.

- Remember the `Client ID` and `Client secret` for the next step.
- The format of the Redirect URL is `https://oauth-openshift.apps.<cluster_name>.<cluster_domain/*`, Fill it in depending your situation

Name [?](#) :

Display name [?](#) :

Logo [?](#) :   
URL [?](#) : 

Home [?](#) :

Description [?](#) :

Organization [?](#) :

Client ID [?](#) :

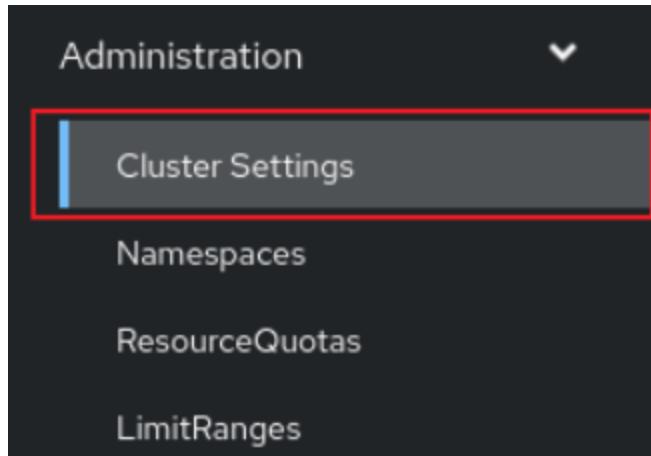
Client secret [?](#) :

Cert [?](#) :

Redirect URLs [?](#) :   
Redirect URLs [Add](#)

## Step2. Openshift Oauth Configuration

Now, login into the Openshift Console as Kubeadmin. Once you are logged In. Browse to the side menu, locate the Cluster settings



Under Global Configuration You will see Oauth

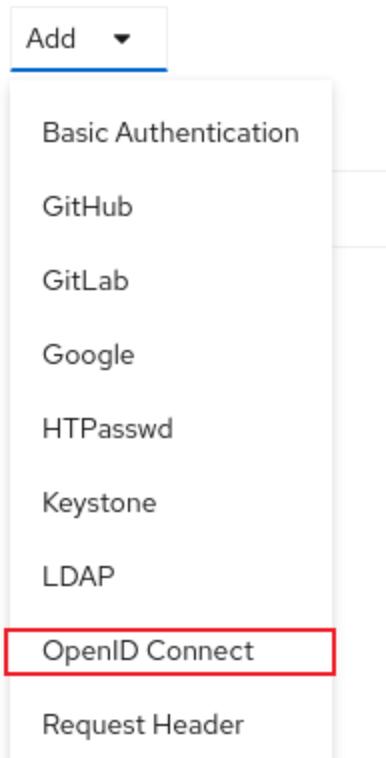
OAuth

OAuth holds cluster-wide information about OAuth. The canonical name is 'cluster'. It is used to configure the integrated OAuth server. This configuration is only honored when the top level Authentication config has type set to IntegratedOAuth. Compatibility level I: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

You will see the Identity Provider section. In ADD section, select the OpenID Connect from options.

## Identity providers

Identity providers determine ho



Configure OIDC, note following points.

- Fill in the `Client ID` and `Client Secret` remembered in the previous step.
- The Issuer URL must use https, with the form `https://<casdoor-host>`, again depending on your situation

## Add Identity Provider: OpenID Connect

Integrate with an OpenID Connect identity provider using an Authorization Code Flow.

Name \*

casdoor

Unique name of the new identity provider. This cannot be changed later.

Client ID \*

2452f2b5abb6ff131199

Client secret \*

\*\*\*\*\*

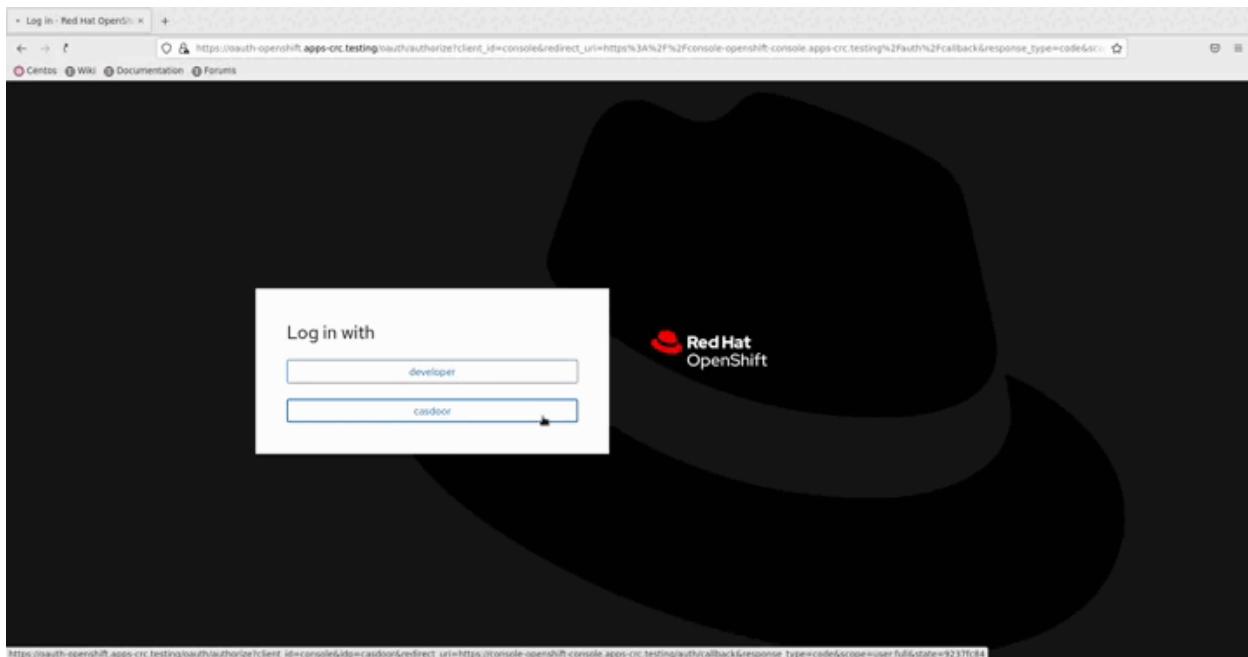
Issuer URL \*

https://demo.casdoor.com/

The URL that the OpenID provider asserts as its issuer identifier. It must use the https scheme with no URL query parameters or fragment.

## Step3. Test OIDC Authentication

Access the Openshift console in the browser. You will see casdoor (The Name you configured in the previous step). Click on the casdoor login option. You will get redirected to the Casdoor login page.



# BookStack

## 在 BookStack 中使用 Casdoor 进行身份验证

BookStack 是一个开源书和文档共享网站 以及使用 Go 语言开发的开源应用程序，帮助您更好地实现文档阅读管理。

Bookstack-casdoor 已经与 castor集成，您现在可以快速开始一个简单的配置。

### 步骤1. 创建一个Casdoor应用程序

转到您的 Casdoor 并添加您的新应用程序 BookStack。 下面是在Casdoor中创建 BookStack应用程序的示例。

Edit Application Save Save & Exit

Name ? : bookstack

Display name ? : bookstack

Logo ? : URL ? : [https://cdn.casdoor.com/logo/casdoor-logo\\_1185x256.png](https://cdn.casdoor.com/logo/casdoor-logo_1185x256.png)

Preview: 

Home ? :

Description ? :

Organization ? :

Client ID ? :

Client secret ? :

请记住 **名称**, **组织**, **客户端 ID**, 和 **客户密钥**。您将在下一步中使用它们。

## 步骤2. 配置Casdoor 登录

现在, 请移动到BookStack。查找文件: `oauth.conf.example`。

将 `oauth.conf.example` 重命名为 `oauth.conf` 和 **修改配置** 默认内容为:

```
[oauth]
```

## 步骤3. 在Casdoor中填写 重定向URL

最后一步，请回到您添加 BookStack 应用程序的页面，填写 重定向URL 请确保 重定向 URL 和 oauth.conf 文件中的 重定向URL 相同。

The screenshot shows the 'Redirect URLs' section of the Casdoor configuration interface. It includes a header with 'Redirect URLs' and a blue 'Add' button. Below is a table with one row, showing a 'Redirect URL' field containing 'http://localhost:8181/login/callback'. There is also a small 'Copy' icon next to the URL.

现在你已经完成了Casdoor的所有配置！

一旦BookStack成功部署，您可以使用Casdoor返回BookStack和体验以进行登录认证。

# Bytebase

Casdoor can use OAuth2 to connect various applications. Here we will use Bytebase as an example to show you how to use OAuth2 to connect to your applications.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`Bytebase_HOSTNAME`: Domain name or IP where Bytebase is deployed.

## Step1. Deploy Casdoor and Bytebase

Firstly, the Casdoor and Bytebase should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME = http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find a redirect url: `<CASDOOR_HOSTNAME>/oauth/callback`
3. Add your redirect url to casdoor application:

The screenshot shows the Casdoor application settings page. It includes fields for Client ID (e828fd6922f4292b979e), Client secret (bab9f6c2fad67471e1bd81e074ea192e4f46dd), Cert (cert-built-in), and Redirect URLs (Redirect URLs: <CASDOOR\_HOSTNAME>/oauth/callback). There is also a table for Redirect URLs with one entry: <CASDOOR\_HOSTNAME>/oauth/callback.

Not surprisingly, you can get two values on the application settings page: `Client ID` and `Client secret` like the picture above, we will use them in the next step.

Open your favorite browser and visit: `http://<CASDOOR_HOSTNAME>/.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

## Step3. Configure Bytebase

1. You should find sso and use OAuth 2.0

The screenshot shows the Bytebase SSO configuration page. On the left sidebar, 'SSO' is highlighted with a black arrow. The main form has 'Type' set to 'OAuth 2.0' (selected) and 'OIDC' (unchecked). Under 'Use template', 'Custom' is selected (indicated by a blue circle with a dot). In the 'Basic information' section, 'Name' is set to 'Custom'. Below it, 'Identity Provider ID' is listed as 'idp-custom-f4mw' with a note that it cannot be changed later.

2. You should config this app

The screenshot shows the Casdoor application interface. On the left, there's a sidebar with navigation items like Account, Profile, Workspace, General, Members, Projects, Subscription, Debug Log, Security & Policy, SQL Review, Risk Center, Custom Approval, Data Anonymization, Data Access Control, Audit Log, Integration, GitOps, SSO, IM, and Archive. The main area is titled 'SSO > casdoor' and contains 'Basic Information' with fields for Name (casdoor), Identity Provider ID (idp-casdoor-mels), Domain (http://101.43.192.216:8000), and Client ID (e828fd992342926979e). It also includes sections for Client secret, Auth URL, Scopes (openid profile email), Token URL, and User information URL. At the bottom, there are buttons for 'Test Connection', 'Archive this SSO', 'Discard changes', and 'Update'.

3. You can find Client Id and Client Secret in Casdoor application page.

- Token server url: `http://CASDOOR_HOSTNAME/api/login/oauth/access_token`
- Authorization server url: `http://CASDOOR_HOSTNAME/login/oauth/authorize`
- User Info server url: `http://CASDOOR_HOSTNAME/api/get-account`
- Scopes: `address phone openid profile offline_access email`

Log out of Bytebase, and test SSO.

The screenshot shows the Bytebase login page. On the left, there's a colorful illustration of a rocket ship launching from a platform, with two cartoon characters (one holding a flag) watching. On the right, the login form has a 'Bytebase' logo at the top. It includes fields for 'Email' (with placeholder 'jim@example.com') and 'Password' (with a 'Forgot your password?' link and a 'Sign in' button). Below these are links for 'New to Bytebase? Sign up' and 'Sign in with casdoor'. At the bottom, there's a language selection for 'English' and '简体中文' (Simplified Chinese), followed by a copyright notice: '© 2023 Bytebase. All rights reserved.'

# ELK

## Casdoor/elk-auth-casdoor概览

ELK (Elasticsearch、Logstash 和 Kibana) 的缺点是，这些产品原来没有认证机制。只要拥有kibana或ESurl，每个人都可以访问kibana dashboard。后来ELK 集成了一个嵌入式认证系统“Xpack”，其所有高级函数 **都不是免费的** (例如 Oauth, OIDC, LDAP, SAML)，而且只有纯粹的身份验证(设置一套账户和密码) 是免费的，这相当不方便。我们不能为每个人提供一个独特的帐户。

因此，我们已经开发了一个基于 Casdoor 免费的 elk 认证解决方案。**开源和维护中，支持大量高级功能。** Casdoor是一个基于Oauth2的中心化身份验证/单点登录平台。/OIDC, casdoor /elk-auth-casdoor 实际上是一种逆向代理，它旨在拦截所有 http 数据流到elk/kibana， 并指导尚未登录的用户登录。只要用户已登录，此逆向代理将完全透明。

如果此用户未成功通过验证，请求将会临时缓存，用户将被重定向到Casto登录页面。在用户成功登录到casdoor后，缓存请求将会还原并发送到 kibana。因此，如果一个 POST请求（或GET以外的其他请求）被拦截，也是可以的，用户不需要重新填写表格和重新发送请求。逆向代理将为你记住它。

Castor/elk-auth-casdoor 版本库的位置 <https://github.com/casdoor/elk-auth-casdoor>

## 如何使用？

0. 已安装golang环境

1. 转到 [casdoor/elk-auth-casdoor](#) 并获取代码
2. 将您的代理注册为Casdoor应用程序。
3. 修改配置

The configuration file locates in "conf/app.conf". Here is an example, and you should customize changes according to your real demands.

```
appname = .
# port on which the reverse proxy shall be run
httpport = 8080
runmode = dev
#EDIT IT IF NECESSARY. The url of this reverse proxy
pluginEndpoint="http://localhost:8080"
#EDIT IT IF NECESSARY. The url of the kibana
targetEndpoint="http://localhost:5601"
#EDIT IT. The url of casdoor
casdoorEndpoint="http://localhost:8000"
#EDIT IT. The clientID of your reverse proxy in casdoor
clientID=ceb6eb261ab20174548d
#EDIT IT. The clientSecret of your reverse proxy in casdoor
clientSecret=af928f0ef1abc1b1195ca58e0e609e9001e134f4
#EDIT IT. The application name of your reverse proxy in
casdoor
appName=ELKProxy
#EDIT IT. The organization to which your reverse proxy belongs
in casdoor
organization=built-in
```

4. visit <http://localhost:8080> (in the example above), and log in following the guidance of redirection, and you shall see kibana protected and authenticated by casdoor.
5. If everything works well, don't forget to block the visits of original kibana's

port coming from outside by configurating your firewall(or something else), so that outsiders can only visit kibana via this reverse proxy.

# Gitea

## 在 Gitea 中使用 Casdoor 进行身份验证

Gitea 是一个社区管理的轻量代码托管解决方案，写入Go。采用MIT开源协议

Gitea支持第三方身份验证，包括Oauth，这样就可以使用Casdoor进行身份验证。以下是操作教程。

### 准备：

要配置 Gitea 使用 Casdoor 作为身份识别提供者，您需要安装 Gitea 以及访问管理员帐户。

关于如何下载、安装和运行 Gitea 的更多信息，请参阅 <https://docs.gitea.io/en-us/install-from-binary/>

您需要在安装过程中创建管理员帐户。如果您已经注册，管理员将是第一个注册用户。请使用此帐户继续以下操作。

### 1. 创建一个Casdoor应用程序

像这样：

Edit Application

Name ⓘ: application\_9p7eai

Display name ⓘ: New Application - 9p7eai

Logo ⓘ: URL ⓘ: https://cdn.casbin.com/logo/logo\_1024x256.png

Preview: 

Home ⓘ: [View](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 7ceb9b7fda4a9061ec1c

Client secret ⓘ: 3416238e1edf915eac08b8fe345b2b95cdba7e04

Cert ⓘ: cert-built-in

Redirect URLs

Action	Redirect URLs	Add
<a href="#">Edit</a> <a href="#">Delete</a>	http://localhost:3000/user/oauth2/Casdoor/callback	<a href="#">Add</a>

请记住客户端ID和密码，以便下一步操作。

请不要在此步骤中填写回调url。Url取决于下一步Gitea的配置。稍后我们将返回来设置一个正确的回调url。

## 2. 配置 Gitea 使用 Casdoor

以管理员身份登录。通过右上角的下拉菜单转到“站点管理”页面。然后切换到“认证源”页面

你应该看到类似下面的内容：

Issues Pull Requests Milestones Explore

Dashboard User Accounts Organizations Repositories Webhooks Authentication Sources User Emails Configuration System Notices Monitoring

Authentication Source Management (Total: 0) Add Authentication Source

ID	Name	Type	Enabled	Updated	Created	Edit

按“添加认证源”按钮并填写类似的表单。

Issues Pull Requests Milestones Explore

Dashboard User Accounts Organizations Repositories Webhooks Authentication Sources User Emails Configuration System Notices Monitoring

Add Authentication Source

Authentication Type \* OAuth2

Authentication Name \* Casdoor

OAuth2 Provider \* OpenID Connect

Client ID (Key) \* 7ceb9b7fda4a9061ec1c

Client Secret \* 3416238e1edf915eac08b8fe345b2b95cdba7e04

Icon URL

OpenID Connect Auto Discovery URL \* http://localhost:8000/.well-known/openid-configuration

Skip local 2FA  
Leaving unset means local users with 2FA set will still have to pass 2FA to log on

Additional Scopes

请选择认证类型为“oauth2”。

请输入此认证源的名称并 **记住此名称**。此名称将在下一步骤中用于回调url。

请选择 **OpenID Connect** Oauth2 提供商。

填写上一步中记住的**客户端ID**和**客户端密码**。

在 **openid** 连接中填写自动发现URL，它应该是 `<your endpoint of casdoor>/.well-known /openid-configuration`。

按您的意愿填写其他可选配置项。然后提交它。

提交表单

### 3. 配置后台回调url

返回第2步中的应用程序编辑页面并添加以下回调url：

`<endpoint of gitea>/user/oauth2/<authentication source name>/callback`

`<authentication source name>` 是上一步Gitea认证源的名称。

### 4. 在 Gitea 上试试

退出当前管理员帐户。

您应该在登录页面中看到这一点：

The screenshot shows a top navigation bar with 'Sign In' and 'OpenID' buttons. Below is a 'Sign In' form with fields for 'Username or Email Address' and 'Password'. There is a 'Remember this Device' checkbox, a 'Sign In' button, a 'Forgot password?' link, and a 'Need an account? Register now.' link. At the bottom is an 'OpenID Connect' sign-in button.

Sign In

OpenID

Sign In

Username or Email Address \*

Password \*

Remember this Device

**Sign In**   [Forgot password?](#)

[Need an account? Register now.](#)

Sign In With OpenID Connect

按“使用openid登录”按钮，您将被重定向到casdoor登录页面。

登录后您将看到这个：

The screenshot shows a 'Complete New Account' form within a Gitea interface. It includes fields for 'Username' and 'Email Address', both marked with a red asterisk indicating they are required. A 'Complete Account' button is at the bottom.

Explore Help

Register New Account [Link to Existing Account](#)

Complete New Account

Username \*

Email Address \*

admin@example.com

**Complete Account**

按照指示并用一个新的 Gitea 帐户或现有帐户绑定下级帐户。

然后一切都将正常工作。



# Grafana

## 在 Grafana 中使用 Casdoor 进行身份验证

Grafana 支持通过 Oauth 进行认证。因此，用户在Grafana上登录变得非常容易。只有几个步骤和简单的配置就能做到这一点。

这是一个使用 Grafana 的 Casdoor 进行身份验证的教程。在您继续之前，请确保您已安装 grafana 并正在运行。

### Step 1 Create an app for Grafana in Casdoor

这是一个在 Casdoor 中创建应用程序的示例

Edit Application Save Save & Exit

Name ⓘ: application\_9p7eai

Display name ⓘ: New Application - 9p7eai

Logo ⓘ:  URL ⓘ: [https://cdn.casbin.com/logo/logo\\_1024x256.png](https://cdn.casbin.com/logo/logo_1024x256.png)

Preview:



casbin

Home ⓘ: [/](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 7ceb9b7fda4a9061ec1c

Client secret ⓘ: 3416238e1edf915eac08b8fe345b2b95cd8a7e04

Cert ⓘ: cert-built-in

Redirect URLs ⓘ: Add

Action	Redirect URL
<span>▲</span> <span>▼</span> <span>✖</span>	<a href="http://localhost:3000/login/generic_oauth">http://localhost:3000/login/generic_oauth</a>

请复制 client secret 和 client ID以便下一步操作。

Please add the callback url of Grafana. By default, Grafana's oauth callback is [/login/generic\\_oauth](/login/generic_oauth)。所以请正确地拼接这个 url。

## Step 2: Modify the configuration of Grafana

By default the configuration file for oauth locates at `conf/defaults.ini` in the workdir of Grafana.

请找到 `auth.generic_oauth` 并修改以下字段：

```
[auth.generic_oauth]
name = Casdoor
```

## About HTTPS

If you don't want HTTPS enabled for casdoor or if you deploy grafana without HTTPS enabled, please also set `tls_skip_verify_insecure = true`

## About redirectURI after Sign In With Casdoor

If the redirect uri is not right after Sign In with Casdoor in Grafana, you may want to configure [root\\_url](#)

```
[server]
http_port = 3000
# The public facing domain name used to access grafana from a
browser
domain = <your ip here>
# The full public facing url
root_url = %(protocol)s://%(domain)s:%(http_port)s/
```

related links:

1. [Grafana doc](#)
2. [Grafana defaults.ini](#)

## About Role Mapping

You may want to configure role\_attribute\_path to map your user's role to Grafana via [role\\_attribute\\_path](#)

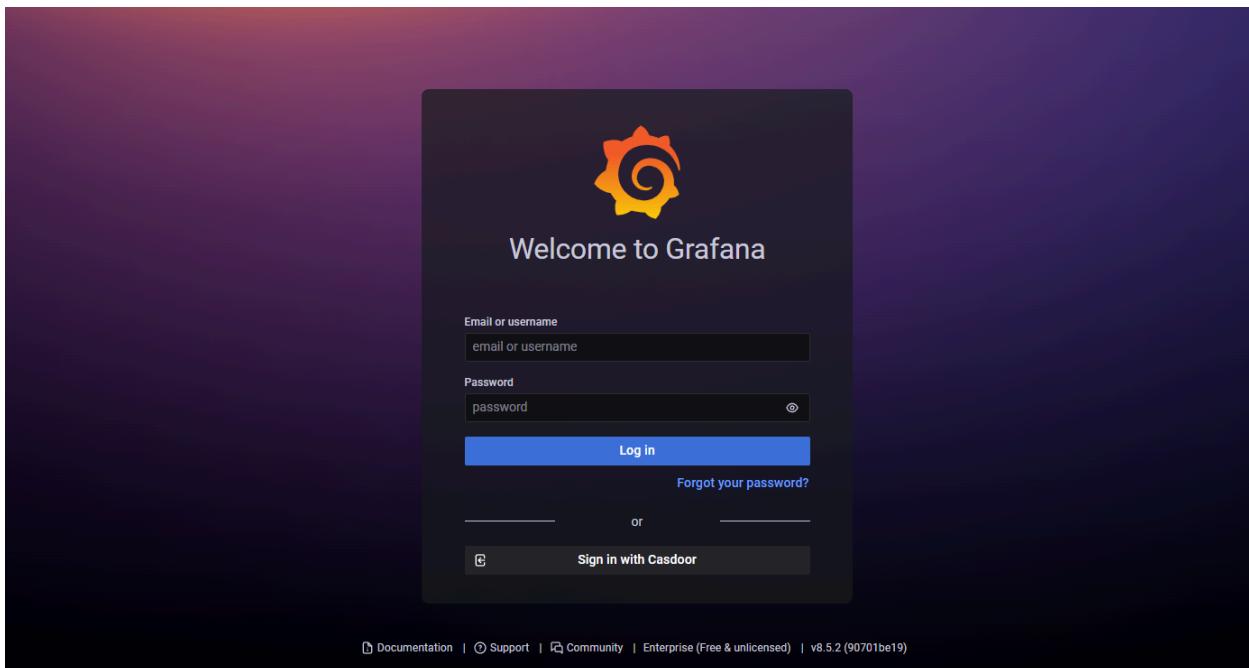
```
[auth.generic_oauth]
role_attribute_path = contains(roles[*].name, 'admin') && 'Admin'
```

the JMESPath expression after role\_attribute\_path is very important here. read grafana doc please

## 第3步：查看它是否正常运作。

Shutdown grafana and restart it.

Go to see the login page, you are supposed to see something like this



# MinIO

MinIO 支持使用OpenID Connect (OIDC) 兼容的程序提供服务进行外部身份管理。此文档介绍了通过配置Casdoor为身份提供者来支持MinIO。

## 第 1 步：部署Casdoor & MinIO

第一，应该先部署Casdoor。

您可以参考 [服务器安装](#) 的 Casdoor 官方文档。

在成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于Casdoor的登录页面。

您可以参考 [这里](#) 来部署您的 MinIO 服务器，以及[这里](#) 对于名为 `mc` 的 MinIO 客户端。

## 第 2 步：配置Casdoor应用

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加您的redirect url

Client ID <a href="#">?</a> :	24a25ea0714d92e78595	Client ID
Client secret <a href="#">?</a> :	155 [REDACTED]	Client Secret
Redirect URLs <a href="#">?</a> :	Redirect URLs	Add
	Redirect URL	Add a redirect URL for spring security
	🔗 http://localhost:8082/ui-one/login/oauth2/code/custom	

3. 添加您想要的提供商并补充其他设置。

Not surprisingly, you can get two values on the application settings page: `Client ID` and `Client secret` like the picture above, we will use them in next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

4. This step is necessary for MinIO. As MinIO needs to use a claim attribute in JWT for its policy, you should configure it in casdoor as well. Currently, casdoor uses `tag` as a workaround for configuring MinIO's policy.

Tag [?](#) : readwrite

You can find all supported policies [here](#).

## 第 3 步： 配置 MinIO

You can start a MinIO server by following commands:

```
导出MINIO_ROOT_USER=minio  
导出 MINIO_ROOT_PASSWORD=minio123  
minio server /mnt/export
```

You can use parameter `--console-address` to configure the address and port.

Then you can add a service alias by MinIO client `mc`.

```
mc 别名设置了myminio <You console address> minio minio123
```

Now, you can configure OpenID connect of MinIO. For Casdoor, the command is like the following:

```
mc admin config set myminio identity_openid  
config_url="http://CASDOOR_HOSTNAME/.well-known/openid-  
configuration" client_id=<client id> client_secret=<client secret>  
claim_name="tag"
```

You can refer to [offical document](#) for more detailed parameters.

Once successfully set, restart the MinIO instance.

```
mc 管理服务重启myminio
```

## 第4步：试用一下demo！

Now, you can open your MinIO console in the browser and click on `Login with SSO`.

You will be redirected to the casdoor user login page. Upon successful login you

will be redirected to MinIO page and logged in automatically and you should see now the buckets and objects they have access to.

### 注意事项

If you deploy frontend and backend of casdoor in different ports, the login page you are redirected to will be backend port and it will display `404 not found`. You can modify the port to the frontend one. Then you can access to casdoor login page successfully.

# Java

## Spring Boot

在Spring Boot项目中使用 Casdoor

## Spring Cloud

在Spring Cloud中使用 Casdoor

## Spring Cloud Gateway

在Spring Cloud Gateway中使用 Casdoor

## Spring 安全

2 个项目

## Jenkins Plugin

Using Casdoor plugin for your Jenkins security

## Jenkins OIDC

Using OIDC protocol as IDP to connect various applications, like Jenkins

## Jira

2 个项目

## Confluence

Using OIDC protocol as IDP to connect various applications, like Confluence

## RuoYi

在 RuoYi-Cloud 上使用 Casdoor

## Pulsar-manager

在 Pulsar-manager 中使用 Casdoor

## ShenYu

在 ShenYu 中使用 Casdoor

## ShardingSphere

在 ShardingSphere 中使用 Casdoor

## Apache IoTDB

Using Casdoor Apache IoTDB

## Apache DolphinScheduler

Using Casdoor for DolphinScheduler SSO login

 **FireZone**

Using OIDC protocol as IDP to connect various applications, like FireZone

 **CloudFoundry**

Learn how to integrate Casdoor with CloudFoundry to secure your applications

 **Thingsboard**

Learn how to integrate Casdoor with Thingsboard to secure your applications

# Spring Boot

[casdoor-spring-boot-example](#) 是关于如何在 SpringBoot 项目中使用 [casdoor-spring-boot-starter](#) 的示例。我们将向您展示以下步骤。

## 第1步：部署Casdoor

第一，应部先部署Casdoor。

您可以参考 [服务安装](#) 的 Casdoor 官方文档。请在 [生产模式](#) 中部署您的 castor 实例。

在成功部署后，您需要确保：

- Open your favorite browser and visit <http://localhost:8000>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 第2步：导入 casdoor-spring-boot-starter

您可以使用 maven 或 gradle 导入 casdoor-spring-boot-starter。

[Maven](#) [Gradle](#)

```
<!-- https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter -->
<dependency>
    <groupId>org.casbin</groupId>
    <artifactId>casdoor-spring-boot-starter</artifactId>
    <version>1.x.y</version>
</dependency>

// https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter
implementation group: 'org.casbin', name: 'casdoor-spring-boot-starter', version: '1.x.y'
```

## 第 3 步： 初始化配置

初始化需要6个参数，它们都是字符串类型。 | 名称(按顺序排列) | 必选项 | 描述 ||  
----- | --- | ----- | | endpoint | 是 |  
Casdoor 服务URL, 例如 http://localhost:8000 | | clientId | 是 |  
Application.client\_id | | clientSecret | 是 | Application.client\_secret | | certificate |  
Yes | Application.certificate | | organizationName | 是 | Application.organization | |  
applicationName | 否 | Application.name | 您可以使用 Java properties 或 YAML 文件  
来初始化，如下所示。

Properties

YML

---

```
casdoor.endpoint = http://localhost:8000
casdoor.clientId = <client-id>
casdoor.clientSecret = <client-secret>
casdoor.certificate = <certificate>
casdoor.organizationName = built-in
```

```
casdoor:  
  endpoint: http://localhost:8000  
  client-id: <client-id>  
  client-secret: <client-secret>  
  certificate: <certificate>  
  organization-name: built-in  
  application-name: app-built-in
```

### ⚠ 注意事项

您应该用您自己的 Casdoor 实例替换配置，特别是 `clientId`, `clientSecret` 和 `jwtPublicKey`。

## 第 4 步：重定向到登录页面

当您需要访问您的应用程序的身份验证时，您可以发送目标 url 并重定向到 Casdoor 提供的登录页面。Please be sure that you have added the callback url (e.g. <http://localhost:8080/login>) in application configuration in advance.

```
@Resource  
private CasdoorAuthService casdoorAuthService;  
  
@RequestMapping("toLogin")  
public String toLogin() {  
    return "redirect:" +  
        casdoorAuthService.getSigninUrl("http://localhost:8080/login");  
}
```

# 第 5 步： 获取令牌并解析

在 Casdoor 验证通过后，它将被重定向到您的应用程序，并带有 code 和状态。

您可以获取 code 并调用 `getOAuthToken` 方法，然后解析出 jwt 令牌。

`Casdoor User` 包含了由Casdoor提供的有关用户的基本信息。您可以使用它作为关键词在您的应用程序中设置会话。

```
@RequestMapping("login")
public String login(String code, String state, HttpServletRequest
request) {
    String token = "";
    CasdoorUser user = null;
    try {
        token = casdoorAuthService.getOAuthToken(code, state);
        user = casdoorAuthService.parseJwtToken(token);
    } catch (CasdoorAuthException e) {
        e.printStackTrace();
    }
    HttpSession session = request.getSession();
    session.setAttribute("casdoorUser", user);
    return "redirect:/";
}
```

## Service

这两种情况的示例如下所示：

- CasdoorAuthService

- `String token = casdoorAuthService.getOAuthToken(code, "app-`

- ```
built-in");
```
- ```
CasdoorUser casdoorUser =
casdoorAuthService.parseJwtToken(token);
```
- CasdoorUserService
  - ```
CasdoorUser casdoorUser = casdoorUserService.getUser("admin");
```
  - ```
CasdoorUser casdoorUser =
casdoorUserService.getUserByEmail("admin@example.com");
```
  - ```
CasdoorUser[] casdoorUsers = casdoorUserService.getUsers();
```
  - ```
CasdoorUser[] casdoorUsers =
casdoorUserService.getSortedUsers("created_time", 5);
```
  - ```
int count = casdoorUserService.getUserCount("0");
```
  - ```
CasdoorResponse response = casdoorUserService.addUser(user);
```
  - ```
CasdoorResponse response =
casdoorUserService.updateUser(user);
```
  - ```
CasdoorResponse response =
casdoorUserService.deleteUser(user);
```
- CasdoorEmailService
  - ```
CasdoorResponse response = casdoorEmailService.sendEmail(title,
content, sender, receiver);
```
- CasdoorSmsService
  - ```
CasdoorResponse response =
casdoorSmsService.sendSms(randomCode(), receiver);
```
- CasdoorResourceService
  - ```
CasdoorResponse response =
casdoorResourceService.uploadResource(user, tag, parent,
fullFilePath, file);
```
  - ```
CasdoorResponse response =
casdoorResourceService.deleteResource(file.getName());
```

# 更多内容

您可以探索以下项目/文件来了解更多关于Java与Casdoor一体化的信息。

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)

# Spring Cloud

在Spring Cloud微型服务体系下，一般由网关进行认证。请参阅 [casdoor-springcloud-gateway-example](#)。

如果您想要在单个服务中使用Casdoor，您可以参考 [casdoor-spring-boot](#)示例。

无论是网关还是在某个服务负责认证，都需要使用 [casdoor-spring-boot-starter](#)

## 更多内容

您可以查阅以下项目/文件来了解更多关于Java中集成Casdoor信息。

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)
- [casdoor-springcloud-gateway-example](#)

# Spring Cloud Gateway

[casdoor-springcloud-gateway-example](#) 是如何在 Spring Cloud Gateway 中使用 [casdoor-spring-boot-starter](#) 作为一个 OAuth2 插件。我们将向您展示以下步骤。

## 第1步 部署Casdoor

首先，应当部署Casdoor。

您可以参考 [Server Installation](#) 的 Casdoor 官方文档。请在 **生产模式** 中部署您的 Casdoor 实例。

在成功部署后，您需要确认：

- Open your favorite browser and visit <http://localhost:8000>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 第2步：初始化一个Spring Cloud Gateway 项目

您可以直接使用此示例的代码或结合您自己的业务代码。

我们需要一个网关服务和至少一个业务服务。

在这个示例中，`casdoor-gateway` 作为网关服务，`casdoor-api` 作为业务服务。

## 第3步：引入依赖

将 `casdoor-spring-boot-starter` 添加到Spring Cloud Gateway项目。

对于Apache Maven:

```
/casdoor-gateway/pom.xml
```

```
<!-- https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter -->
<dependency>
    <groupId>org.casbin</groupId>
    <artifactId>casdoor-spring-boot-starter</artifactId>
    <version>1.x.y</version>
</dependency>
```

对于Gradle:

```
// https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter
implementation group: 'org.casbin', name: 'casdoor-spring-boot-starter', version: '1.x.y'
```

## 第4步：配置属性

初始化需要 6 个参数，它们都是字符串类型：

名称(按顺序排列)	是否必须	描述
endpoint	是	Casdoor 服务URL, 例如 <code>http://localhost:8000</code>
clientId	是	Application.client_id
clientSecret	是	Application.client_secret
certificate	是	Application.certificate
organizationName	是	Application.organization
applicationName	否	Application.name

您可以使用 Java properties 或 YAML 文件来初始化，如下所示。

对于 properties:

```
casdoor.endpoint=http://localhost:8000
casdoor.clientId=<client-id>
casdoor.clientSecret=<client-secret>
casdoor.certificate=<certificate>
casdoor.organizationName=built-in
casdoor.applicationName=app-built-in
```

对于yaml:

```
casdoor:
  endpoint: http://localhost:8000
```

此外，还需要配置网关路由。对于yaml：

```
spring:
  application:
    name: casdoor-gateway
  cloud:
    gateway:
      routes:
        - id: api-route
          uri: http://localhost:9091
          predicates:
            - Path=/api/**
```

## 第5步：添加CasdoorAuthFilter

将 GlobalFilter 的一个实现类添加到网关中做身份验证，例如此示例中的 CasdoorAuthFilter。

如果身份验证失败，则返回到401，前端跳转到统一登录界面。

```
@Component
public class CasdoorAuthFilter implements GlobalFilter, Ordered {

  private static final Logger LOGGER =
LoggerFactory.getLogger(CasdoorAuthFilter.class);

  @Override public int getOrder() {
    return 0;
  }

  @Override public Mono<Void> filter(ServerWebExchange exchange,
GatewayFilterChain chain) {
    return exchange.getSession().flatMap(webSession -> {
      CasdoorUser user =
```

# 第6步：使用Casdoor提供的相关Service

现在提供 5 种Service: `CasdoorAuthService`, `CasdoorUserService`, `Casdoore-mailService`, `CasdoorSmsService` 和 `CasdoorResourceService`

您可以按照下面示例在Gateway项目中创建它们。

```
@Resource  
private CasdoorAuthService casdoorAuthService;
```

当您需要访问您的应用程序的身份验证时，您可以发送目标 url 并重定向到 Casdoor 提供的登录页面。

Please be sure that you have added the callback url (e.g. <http://localhost:9090/callback>) in application configuration in advance.

```
@RequestMapping("login")  
public Mono<String> login() {  
    return Mono.just("redirect:" +  
        casdoorAuthService.getSignInUrl("http://localhost:9090/callback"));  
}
```

在 Casdoor 验证通过后，它将被重定向到您的应用程序，并带有 code 和状态。

您可以获取 code 并调用 `getAuthToken` 方法，然后解析出 jwt 令牌。

`CasdoorUser` 包含了 Casdoor 提供的用户基本信息，您可以将其作为关键字在您的应用程序中设置会话。

```
@RequestMapping("callback")
```

API示例如下：

- CasdoorAuthService
  - `String token = casdoorAuthService.getOAuthToken(code, "app-built-in");`
  - `CasdoorUser casdoorUser = casdoorAuthService.parseJwtToken(token);`
- CasdoorUserService
  - `CasdoorUser casdoorUser = casdoorUserService.getUser("admin");`
  - `CasdoorUser casdoorUser = casdoorUserService.getUserByEmail("admin@example.com");`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getUsers();`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getSortedUsers("created_time", 5);`
  - `int count = casdoorUserService.getUserCount("0");`
  - `CasdoorResponse response = casdoorUserService.addUser(user);`
  - `CasdoorResponse response = casdoorUserService.updateUser(user);`
  - `CasdoorResponse response = casdoorUserService.deleteUser(user);`
- CasdoorEmailService
  - `CasdoorResponse response = casdoorEmailService.sendEmail(title, content, sender, receiver);`
- CasdoorSmsService
  - `CasdoorResponse response = casdoorSmsService.sendSms(randomCode(), receiver);`
- CasdoorResourceService
  - `CasdoorResponse response = casdoorResourceService.uploadResource(user, tag, parent,`

```
fullFilePath, file);  
◦ CasdoorResponse response =  
casdoorResourceService.deleteResource(file.getName());
```

## 第7步：重新启动项目

After start, open your favorite browser and visit <http://localhost:9090>, then click any button which can request resources from casdoor-api.



# Casdoor

Get Resource

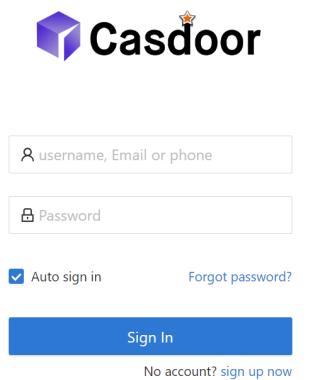
Update Resource

将触发网关认证逻辑。由于您没有登录，您将跳转到登录界面。点击登录按钮。



[Login](#)

随后您可以看到Cassdoor统一的登录平台。



登录成功后，它将跳转到主界面。然后您可以点击任意资源相关按钮。



# Casdoor

[Get Resource](#)

[Update Resource](#)

"success get resource1"

## 更多内容

您可以探索以下项目/文件来了解更多关于Java与Casdoor一体化的信息。

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)
- [casdoor-springcloud-gateway-example](#)

# Spring 安全

## Spring Security OAuth

这里我们将使用Spring Security 作为示例，向您展示如何将 OIDC 链接到您的应用程序。

## Spring Security Filter

基于Spring Security Filter，如何使用OIDC连接您的应用程序。

# Spring Security OAuth

Casdoor 可以使用 OIDC 协议作为IDP 连接各种应用程序。 这里我们将使用Spring Security作为示例，向您展示如何使用 OIDC 链接到您的应用程序。

## 步骤1. 部署Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 第2步： 配置Casdoor应用程序

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加您的重定向url (您可以在下一节中看到更多关于如何获取重定向url的细节)

Client ID ? : 24a25ea0714d92e78595 Client ID

Client secret ? : 155... Client Secret

Redirect URLs ? :  
Redirect URLs Add  
Redirect URL Add a redirect URL for spring security  
🔗 http://localhost:8082/ui-one/login/oauth2/code/custom

3. 添加您想要的提供商并补充其他设置。

不出意外的话，您会在应用程序设置页面看到： Client ID 和 Client secret 就像上面的图片一样。我们将在下一步中使用它们。您将在下一步中使用它们。

打开你喜欢的浏览器，访问：[http://CASDOOR\\_HOSTNAME/.well-known/openid-configuration](http://CASDOOR_HOSTNAME/.well-known/openid-configuration)，你会看到 Casdoor 的 OIDC 配置。

## 步骤3. 配置Spring Security

Spring Security 完全支持 OIDC。

您可以自定义Spring Security OAuth2 客户端的设置：

### ⚠ 注意事项

您应该用您自己的 Casdoor 实例替换配置，特别是 <Client ID> 等。

[application.yml](#) [application.properties](#)

```
spring:  
  security:  
    oauth2:  
      client:
```

```
spring.security.oauth2.client.registration.casdoor.client-id=<Client ID>
spring.security.oauth2.client.registration.casdoor.client-
secret=<Client Secret>
spring.security.oauth2.client.registration.casdoor.scope=<Scope>
spring.security.oauth2.client.registration.casdoor.authorization-grant-
type=authorization_code
spring.security.oauth2.client.registration.casdoor.redirect-
uri=<Redirect URL>

spring.security.oauth2.client.provider.casdoor.authorization-
uri=http://CASDOOR_HOSTNAME:7001/login/oauth/authorize
spring.security.oauth2.client.provider.casdoor.token-
uri=http://CASDOOR_HOSTNAME:8000/api/login/oauth/access_token
spring.security.oauth2.client.provider.casdoor.user-info-
uri=http://CASDOOR_HOSTNAME:8000/api/get-account
spring.security.oauth2.client.provider.casdoor.user-name-attribute=name
```

### ⚠ 注意事项

对于Spring Security的默认情况，<Redirect URL> 应该等于 http://<Your Spring Boot Application Endpoint>/<Servlet Prefix if it is configured>/login/oauth2/code/code。例如，对于下面的演示来说，重定向URL应该是 http://localhost:8080/login/oauth2/code/code/custom。您也应该在 casdoor 应用程序中配置它。

您也可以通过 ClientRegistration 在您的代码中自定义设置。您可以在[这里](#)找到映射

## 步骤4. 从一个 Demo 开始

1. 我们可以创建 Spring Boot 应用程序。
2. We can add a configuration which protects all endpoints except / 和 /login\*\* for users to log in.

```

@EnableWebSecurity
public class UiSecurityConfig extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.authorizeRequests().antMatchers("/", "/login**").permitAll().anyRequest().authenticated().and()
            .oauth2Login();

    }
}

```

3. We can add a naive page for user to log in.

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Spring OAuth Client Thymeleaf - 1</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/
bootstrap.min.css" />
</head>
<body>
    <nav
        class="navbar navbar-expand-lg navbar-light bg-light shadow-
sm p-3 mb-5">
        <a class="navbar-brand" th:href="@{/foos/}">Spring OAuth
Client
            Thymeleaf - 1</a>
    </nav>
    <div class="container">
        <label>Welcome ! </label> <br /> <a th:href="@{/foos/}"
            class="btn btn-primary">Login</a>
    </div>
</body>
</html>

```

When user clicks the `login` button, he will be redirected to `casdoor`.

4. Next, we can define our protected resources. We can export an endpoint called `/foos` and a web page for display.

### Data Model

```
public class FooModel {  
    private Long id;  
    private String name;  
  
    public FooModel(Long id, String name) {  
        super();  
        this.id = id;  
        this.name = name;  
    }  
    public Long getId() {  
        return id;  
    }  
    public void setId(Long id) {  
        this.id = id;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

### Controller

```
@Controller  
public class FooClientController {  
    @GetMapping("/foos")  
    public String getFoos(Model model) {  
        List<FooModel> foos = new ArrayList<>();  
        foos.add(new FooModel(1L, "a"));  
        foos.add(new FooModel(2L, "b"));  
        foos.add(new FooModel(3L, "c"));  
    }  
}
```

## Web page

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Spring OAuth Client Thymeleaf - 1</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/
bootstrap.min.css" />
</head>
<body>
    <nav
        class="navbar navbar-expand-lg navbar-light bg-light shadow-
sm p-3 mb-5">
        <a class="navbar-brand" th:href="@{/foos/}">Spring OAuth
Client
            Thymeleaf -1</a>
        <ul class="navbar-nav ml-auto">
            <li class="navbar-text">Hi, <span
sec:authentication="name">preferred_username</span>&ampnbsp&ampnbsp&ampnbsp;
            </li>
        </ul>
    </nav>
    <div class="container">
        <h1>All Foos:</h1>
        <table class="table table-bordered table-striped">
            <thead>
                <tr>
                    <td>ID</td>
                    <td>Name</td>
                </tr>
            </thead>
            <tbody>
                <tr th:if="${foos.empty}">
                    <td colspan="4">No foos</td>
                </tr>
                <tr th:each="foo : ${foos}">
                    <td><span th:text="${foo.id}"> ID </span></td>
                    <td><span th:text="${foo.name}"> Name

```

### ⚠ 注意事项

All the web page template should be put under `resources/templates`.

## 步骤5. 试用一下demo!

Firstly, you can try to open your favorite browser and directly visit `/foos`. It will automatically redirect to casdoor's login page. You can log in here or from the root page.

If you visit your root page,

Spring OAuth Client Thymeleaf - 1

Welcome !  
[Login](#)

Click the `login` button and the page will redirect to casdoor's login page.



username, Email or phone

Password

Auto sign in      [Forgot password?](#)

[Sign In](#)

[Sign in with code](#)    No account? [sign up now](#)

After you log in, the page will redirect to </foos>.

Spring OAuth Client Thymeleaf -1

Hi, [Logout](#)

Your Username

### All Foos:

ID	Name
1	a
2	b
3	c

# Spring Security Filter

Casdoor 可以使用 OIDC 协议作为IDP 连接各种应用程序。下面我们将使用spring security中的过滤器来集成casdoor，并向您展示如何使用oidc连接到应用程序。

## 步骤1. 部署Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 第2步： 配置Casdoor应用程序

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加您的重定向url (您可以在下一节中看到更多关于如何获取重定向url的细节)

Name [?](#) : application\_a6ftas → your application name

Display name [?](#) : New Application - a6ftas

Logo [?](#) : URL [?](#) : [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home [?](#) :

Description [?](#) :

Organization [?](#) : organization\_carg1b → your organization name

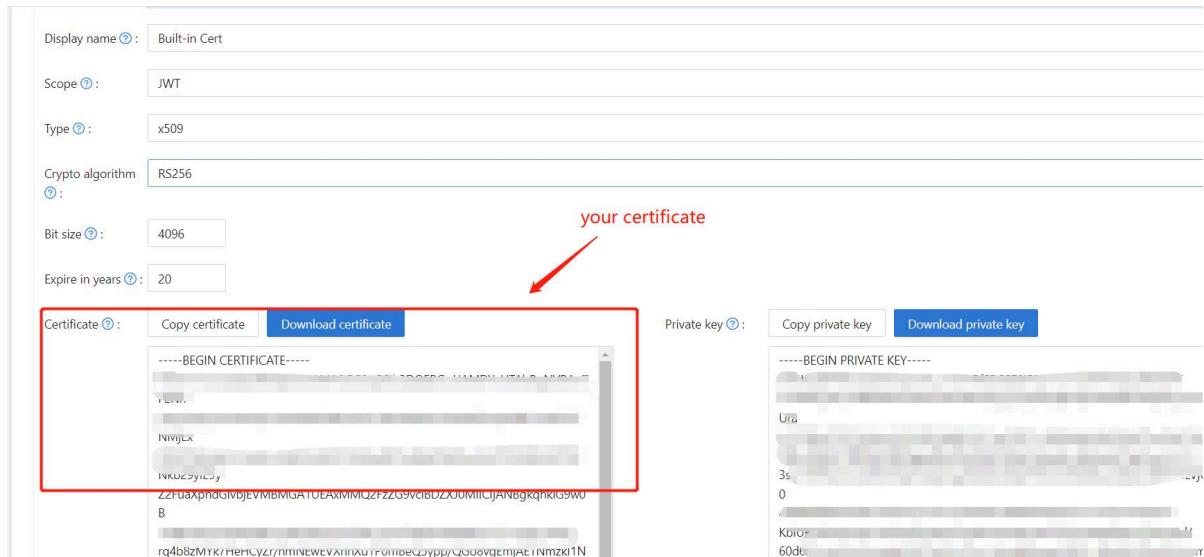
Client ID [?](#) : 3ed7314825ecf955cb19 → your client id

Client secret [?](#) : ee9314ea228 → your client secret

Cert [?](#) : cert-built-in

Redirect URLs [?](#) : Redirect URLs Add  
Redirect URL → your redirect url  
<http://localhost:3000/callback>

3. 在证书编辑页面上，您可以看到您的 [证书](#)。



#### 4. 添加您想要的提供商并补充其他设置。

不出意外的话，您会在应用程序设置页面看到： 应用程序名称，组织名称，重定向URL，客户端ID，客户密钥，认证. 如上所述，我们将在下一步中使用它们。

打开你喜欢的浏览器，访问：[http://CASDOOR\\_HOSTNAME/.well-known/openid-configuration](http://CASDOOR_HOSTNAME/.well-known/openid-configuration)，你会看到 Casdoor 的 OIDC 配置。

## 步骤3. 配置Spring Security

您可以自定义spring security filter 的设置来处理标记：

### ⚠ 注意事项

您应该用您自己的 Casdoor 实例替换配置，特别是 <Client ID> 等。

```
server:
  port: 8080
casdoor:
  endpoint: http://CASDOOR_HOSTNAME:8000
  client-id: <Client ID>
  client-secret: <Client Secret>
```

### ⚠ 注意事项

对于前端应用程序来说，`<FRONTEND_HOSTNAME>` 的默认值是 `localhost:3000`。

例如，对于下面的演示来说，重定向URL应该是 `http://localhost:3000/callback`。

您也应该在 `casdoor` 应用程序中配置它。

## 步骤4. 配置前端

您需要安装 `casdoor-javascript-sdk` 并配置 `SDK`。

1. 安装 `casdoor-javascript-sdk`。

```
npm i casdoor-javascript-sdk
# or
yarn add casdoor-javascript-sdk
```

2. 设置 `SDK`。

```
import Sdk from "casdoor-javascript-sdk";

// Serverurl is the URL where spring security is deployed
export const ServerUrl = "http://BACKEND_HOSTNAME:8080";

const sdkConfig = {
  serverUrl: "http://CASDOOR_HOSTNAME:8000",
  clientId: "<your client id>",
  appName: "<your application name>",
  organizationName: "<your organization name>",
  redirectPath: "/callback",
};

export const CasdoorSDK = new Sdk(sdkConfig);
```

# 步骤5. 从一个 Demo 开始

1. 我们可以创建 Spring Boot 应用程序。
2. We can add some configurations to handle JWT.

```
@EnableWebSecurity
public class SecurityConfig {

    private final JwtTokenFilter jwtTokenFilter;

    public SecurityConfig(JwtTokenFilter jwtTokenFilter) {
        this.jwtTokenFilter = jwtTokenFilter;
    }

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        // enable CORS and disable CSRF
        http = http.cors(corsConfig -> corsConfig
            .configurationSource(configurationSource()))
            .csrf().disable();

        // set session management to stateless
        http = http
            .sessionManagement()

        .sessionCreationPolicy(SessionCreationPolicy.STATELESS)
            .and();

        // set permissions on endpoints
        http.authorizeHttpRequests(authorize -> authorize
            .mvcMatchers("/api/redirect-url", "/api/signin").permitAll()
            .mvcMatchers("/api/**").authenticated()
        );

        // set unauthorized requests exception handler
    }
}
```

3. We can add a simple JWT filter to intercept requests that need to verify tokens.

```
@Component
public class JwtTokenFilter extends OncePerRequestFilter {

    private final CasdoorAuthService casdoorAuthService;

    public JwtTokenFilter(CasdoorAuthService casdoorAuthService) {
        this.casdoorAuthService = casdoorAuthService;
    }

    @Override
    protected void doFilterInternal(HttpServletRequest request,
                                    HttpServletResponse response,
                                    FilterChain chain)
        throws ServletException, IOException {
        // get authorization header and validate
        final String header =
request.getHeader(HttpHeaders.AUTHORIZATION);
        if (!StringUtils.hasText(header) ||
!header.startsWith("Bearer ")) {
            chain.doFilter(request, response);
            return;
        }

        // get jwt token and validate
        final String token = header.split(" ")[1].trim();

        // get user identity and set it on the spring security
        context
        UserDetails userDetails = null;
        try {
            CasdoorUser casdoorUser =
casdoorAuthService.parseJwtToken(token);
            userDetails = new CustomUserDetails(casdoorUser);
        } catch (CasdoorAuthException exception) {
            logger.error("casdoor auth exception", exception);
            chain.doFilter(request, response);
            return;
        }
    }
}
```

When the user accesses the interface requiring authentication, `JwtTokenFilter` will obtain the token from the request header `Authorization` and verify it.

4. Next, we need to define a `Controller` to handle that when the user login to the `casdoor`, it will be redirected to the server and carry the `code` and `state`. The server needs to verify the user's identity from the `casdoor` and obtain the `token` through these two parameters.

```
@RestController
public class UserController {

    private static final Logger logger =
LoggerFactory.getLogger(UserController.class);

    private final CasdoorAuthService casdoorAuthService;

    // ...

    @PostMapping("/api/signin")
    public Result signin(@RequestParam("code") String code,
@RequestParam("state") String state) {
        try {
            String token = casdoorAuthService.getOAuthToken(code,
state);
            return Result.success(token);
        } catch (CasdoorAuthException exception) {
            logger.error("casdoor auth exception", exception);
            return Result.failure(exception.getMessage());
        }
    }

    // ...
}
```

## 步骤6. 试用一下demo!

First, you can try to access the frontend application through the browser. If you have not

logged in, it will display a login button. Click the login button, and you will be redirected to the `casdoor` login page.

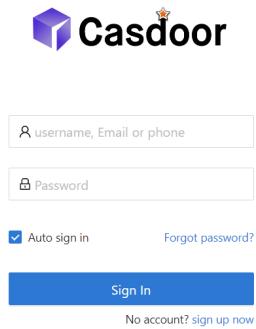
If you visit your root page,

---

`Casdoor Login`

Click the `Casdoor Login` button and the page will redirect to casdoor's login page.

---



After you log in, the page will redirect to `/`.



New User - rtsbx4

[Logout](#)

# Jenkins Plugin

Casdoor provides a plugin for users to login Jenkins. Here we will show you how to use Casdoor plugin for your Jenkins security.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`JENKINS_HOSTNAME`: Domain name or IP where Jenkins is deployed.

## Step1. Deploy Casdoor and Jenkins

Firstly, the [Casdoor](#) and Jenkins should be deployed.

After a successful deployment, you need to ensure:

1. Set Jenkins URL(Manage Jenkins → Configure System → Jenkins Location) to `JENKINS_HOSTNAME`.

The screenshot shows the Jenkins 'Dashboard' > 'configuration' page. In the 'Jenkins Location' section, the 'Jenkins URL' field contains the value `http://10.144.125.123:6780`, which is highlighted with a red box. Below it, the 'JENKINS\_HOSTNAME' field contains the value `address not configured yet <nobody@nowhere>`. In the 'Global properties' section, there is a checkbox for 'Environment variables'. At the bottom, there are 'Save' and 'Apply' buttons.

2. Casdoor can be logged in and used normally.
3. Set Casdoor's `origin` value (conf/app.conf) to `CASDOOR_HOSTNAME`.

```
conf > ⚙ app.conf
  8  dbName = casdoor
  9  redisEndpoint =
10  defaultStorageProvider =
11  isCloudIntranet = false
12  authState = "casdoor"
13  httpProxy = "127.0.0.1:10808"
14  verificationCodeTimeout = 10
15  initScore = 2000
16  logPostOnly = true
17  origin = "http://10.144.1.2:8000"|
                                CASDOOR_HOSTNAME
```

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url: `http://JENKINS_HOSTNAME/securityRealm/finishLogin`

The screenshot shows the Casdoor application settings page. It includes fields for Client ID (bbd0bd66696e504dec59), Client secret (d2de01b01...110b47465c), and Redirect URLs (http://10.144.125.123:6780/securityRealm/finishLogin).

Description	Casdoor for Jenkins	
Organization	built-in	
Client ID	bbd0bd66696e504dec59	Client ID
Client secret	d2de01b01...110b47465c	Client secret
Redirect URLs	Redirect URLs <a href="#">Add</a> Redirect URL <a href="http://10.144.125.123:6780/securityRealm/finishLogin">http://10.144.125.123:6780/securityRealm/finishLogin</a> <a href="#">Add a redirect url for Jenkins</a> <b>JENKINS_HOSTNAME</b>	

3. Add provider you want and supplement other settings.

Not surprisingly, you can get two values on the application settings page: `Client`

`ID` and `Client secret` like the picture above, we will use them in next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

## Step3. Configure Jenkins

Now, you can install Casdoor plugin from the market or by uploading its `.jar` file.

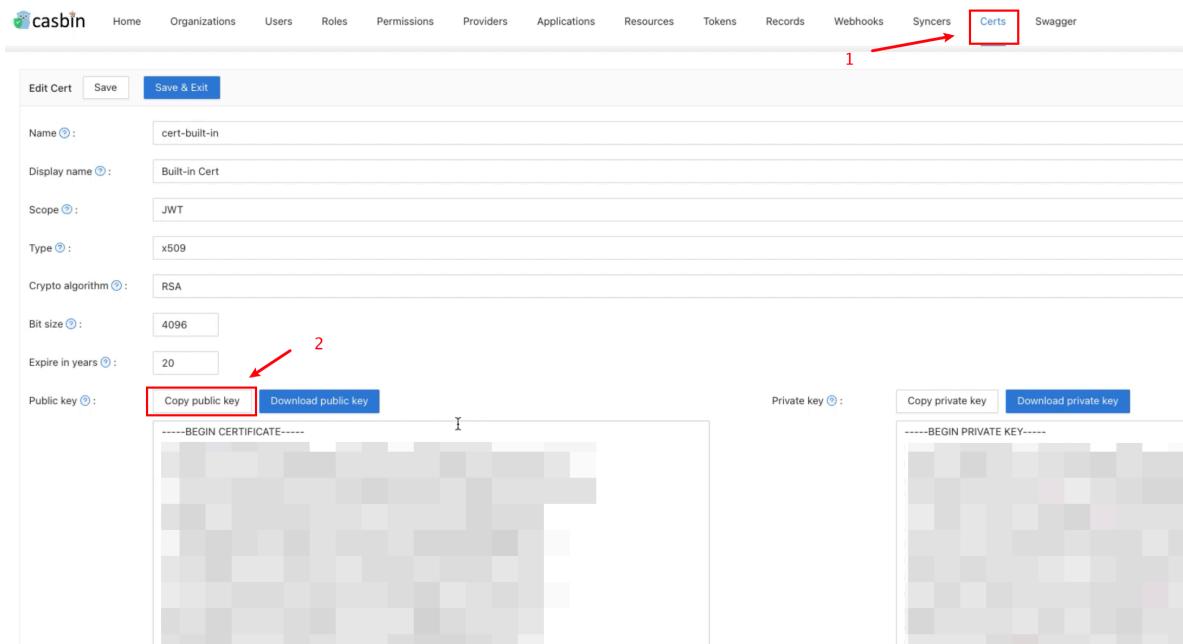
After completing the installation, go to Manage Jenkins → Configure Global Security.

Suggestion: Back up the Jenkins `config.xml` file, and use it to recover in case of setup errors.

The screenshot shows the Jenkins 'Configure Global Security' page. Under the 'Authentication' section, there is a checkbox for 'Disable remember me'. In the 'Security Realm' section, a radio button is selected for 'Casdoor Authentication Plugin'. The 'Casdoor Endpoint' field is filled with a placeholder URL. Below it, error messages indicate that 'Casdoor Endpoint is required.' and 'Client Id is required.' The 'Client ID' and 'Client Secret' fields are empty, with corresponding error messages: 'Client Id is required.' and 'Client Secret is required.'. The 'JWT Public Key' field is also empty, with the error message 'Jwt Public Key is required.' The 'Organization Name' and 'Application Name' fields are empty. At the bottom, there are 'Save' and 'Apply' buttons, and an 'Advanced...' link.

1. In Security Realm, select "Casdoor Authentication Plugin".
2. In Casdoor Endpoint, specify the `CASDOOR_HOSTNAME` noted above.

3. In Client ID, specify the `Client ID` noted above.
4. In Client secret, specify the `Client secret` noted above.
5. In JWT Public Key, specify the public key used to validate JWT token. You can find the public key in Casdoor by clicking `Cert` at the top. After clicking `edit` your application, you can copy your public key in the following page.



6. Organization Name and Application Name is optional. You can specify your organization and application to verify users in other organizations and applications. If they are empty, the plugin will use the default organization and application.
7. In the Authorization section, check “Logged-in users can do anything”. Disable “Allow anonymous read access”.
8. Click `save`.

Now, Jenkins will automatically redirect you to Casdoor for authentication.

# Jenkins OIDC

Casdoor can use OIDC protocol as IDP to connect various applications. Here we will use Jenkins as an example to show you how to use OIDC to connect to your applications.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`JENKINS_HOSTNAME`: Domain name or IP where Jenkins is deployed.

## Step1. Deploy Casdoor and Jenkins

Firstly, the [Casdoor](#) and [Jenkins](#) should be deployed.

After a successful deployment, you need to ensure:

1. Set Jenkins URL([Manage Jenkins](#) → [Configure System](#) → Jenkins Location) to `JENKINS_HOSTNAME`.

Dashboard > configuration

**Jenkins Location**

Jenkins URL  
 ?

JENKINS\_HOSTNAME  
 ?

**Serve resource files from another domain**

Resource Root URL  
 ?

Without a resource root URL, resources will be served from the Jenkins URL with Content-Security-Policy set.

**Global properties**

Environment variables

**Save** **Apply**

2. Casdoor can be logged in and used normally.
3. Set Casdoor's `origin` value (conf/app.conf) to `CASDOOR_HOSTNAME`.

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000" | CASDOOR_HOSTNAME
```

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url: `http://JENKINS_HOSTNAME/securityRealm/finishLogin`

Description [?](#): Casdoor for Jenkins

Organization [?](#): built-in

Client ID [?](#): bbd0bd66696e504dec59 Client ID

Client secret [?](#): d2de01b01 [REDACTED] 110b47465c Client secret

Redirect URLs [?](#):

Redirect URLs	Add
Redirect URL	<a href="http://10.144.125.123:6780/securityRealm/finishLogin">http://10.144.125.123:6780/securityRealm/finishLogin</a> <span style="color: red;">Add a redirect url for Jenkins</span>
	JENKINS_HOSTNAME

3. Add provider you want and supplement other settings.

Not surprisingly, you can get two values on the application settings page: Client ID and Client secret like the picture above, we will use them in the next step.

Open your favorite browser and visit: [http://CASDOOR\\_HOSTNAME/.well-known/openid-configuration](http://CASDOOR_HOSTNAME/.well-known/openid-configuration), you will see the OIDC configure of Casdoor.

## Step3. Configure Jenkins

First, we need to install [OpenId Connect Authentication](#), Jenkins does not natively support OIDC.

After completing the installation, go to [Manage Jenkins → Configure Global Security](#).

The screenshot shows the Jenkins 'Manage Jenkins' interface. On the left, there's a sidebar with links like '用户列表' (User List), '构建历史' (Build History), 'Manage Jenkins' (which is highlighted with a red border), and 'My Views'. Below that are sections for '构建队列' (Build Queue) and '构建执行状态' (Build Execution Status). The main area is titled 'System Configuration' and contains several management options: 'Configure System' (global settings), 'Global Tool Configuration' (tools and installers), 'Manage Nodes and Clouds' (node management), 'Manage Plugins' (plugin management), 'Configure Global Security' (highlighted with a red border), 'Manage Credentials' (credential management), and 'Configure Credential Providers'.

### 💡 提示

Back up the Jenkins `config.xml` file, and use it to recover in case of setup errors.

1. In Access Control, Security Realm select `Login with Openid Connect`.
2. In Client ID, specify the `Client ID` noted above.
3. In Client secret, specify the `Client secret` noted above.
4. In Configuration mode, select `Automatic configuration` and fill in `http://CASDOOR_HOSTNAME/.well-known/openid-configuration` into Well-known configuration endpoint.

**Security Realm**

- Delegate to servlet container ?
- Jenkins' own user database ?
- Login with Openid Connect Select this  ?

**Client id**

bbd0bd66696e504dec59	Input your Client ID
----------------------	----------------------

**Client secret**

 Concealed	Input your Client secret	<span style="color: blue;">Change Password</span>
---	--------------------------	---

**Configuration mode**

- Automatic configuration ?
- Well-known configuration endpoint ?
- Manual configuration ?

`http://[10.144.1.2:8000]/well-known/openid-configuration`

**CASDOOR\_HOSTNAME**

If your casdoor is deployed locally, you may need to select **Manual configuration** and input some information:

- Token server url: `http://[CASDOOR_HOSTNAME]/api/login/oauth/access_token`
- Authorization server url: `http://[CASDOOR_HOSTNAME]/login/oauth/authorize`
- UserInfo server url: `http://[CASDOOR_HOSTNAME]/api/get-account`
- Scopes: `address phone openid profile offline_access email`

**Configuration mode**

- Automatic configuration ?
- Manual configuration ?

**Token server url**

http://[10.144.1.2:8000]/api/login/oauth/access_token
---

**CASDOOR\_HOSTNAME**

**Authorization server url**

http://[10.144.1.2:8000]/login/oauth/authorize
--

**Userinfo server url**

http://[10.144.1.2:8000]/api/get-account
--

**Scopes**

address phone openid profile offline_access email
---

## 5. Click Advanced setting, fill in the following:

- In User name field, specify `name`
- In Full name field, specify `displayName`

- In Email field, specify `email`

User name field name	<code>name</code>
Full name field name	<code>displayName</code>
Email field name	<code>email</code>
Groups field name <small>?</small>	<code></code>
Token Field Key To Check <small>?</small>	<code></code>

6. In the Authorization section, check “Logged-in users can do anything”. Disable “Allow anonymous read access”. You can configure more complex authorization later, for now check if OpenID actually works.

Log out of Jenkins, it should now redirect you to Casdoor for authentication.



Auto sign in
[Forgot password?](#)

[Sign In](#)

[Sign in with code](#)    [No account? sign up now](#)

# Jira

## Via Built-in SSO

Using OIDC protocol as IDP to connect various applications, like Jira

## Via miniOrange Plugin

使用 OIDC 协议作为IDP 连接各种应用程序，如Jira

# Via Built-in SSO

This is a free method to connect casdoor, but your website must use https;

Casdoor can use OIDC protocol as IDP to connect various applications. Here is a [Jira tutorial](#).

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`Jira_HOSTNAME`: Domain name or IP where Jira is deployed.

## Step1. Deploy Casdoor and Jira

Firstly, the [Casdoor](#) and [Jira](#) should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME = http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find Authentication methods:

The screenshot shows the Jira Software Administration interface. The left sidebar has sections like Applications, Projects, Issues, Manage apps, User management, and System (which is selected). The main content area is titled 'Authentication methods' and contains sections for 'Make authentication safer', 'Login options', and 'Authentication on API calls'. A red box highlights the 'System' tab in the top navigation bar, and another red box highlights the 'Authentication methods' section in the sidebar.

### 3. Add a Configure and choose OpenId Connection signle sign-on in Authentication method

#### Add new configuration

Name \*

Use a unique name for this configuration.

Authentication method

OpenID Connect single sign-on



Users log in using OpenID Connect

### 4. Find the redirect url:

Give these URLs to your identity provider

Redirect URL

<https://test.v2tl.com/plugins/servlet/oidc/callback>



Location where the client is sent to after successful account authentication.

## 5. Add a redirect url:

The screenshot shows the Casdoor application settings page. It includes fields for Client ID (642ec5d6779a2f0e879d), Client secret (26cb47985c47ae3844580536ce2f59872969e109), Cert (cert-built-in), and a Redirect URLs section. The Redirect URLs section contains an 'Add' button and a table with one row: https://test.v2tl.com/plugins/servlet/oidc/callback. There are also Action buttons for sorting and deleting.

Not surprisingly, you can get two values on the application settings page: `Client ID` and `Client secret` like the picture above, we will use them in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

## Step3. Configure Jira

1. We need continue to config our Configure in jira

## Edit existing configuration

Name \*

casdoor

Use a unique name for this configuration.

Authentication method

OpenID Connect single sign-on

Users log in using OpenID Connect

### OpenID Connect settings

Issuer URL \*

https://demo.casdoor.com

your casdoor url

The complete URL of the OpenID Provider. Needs to be unique.

Client ID \*

642ec5d6779a2f0e879d

application client ID

The client identifier, as registered with the OpenID Provider.

Client secret \*

.....

application client secret [Change](#)

Client secret is used in conjunction with the Client ID to authenticate the client application against the OpenID Provider.

Username mapping \*

\${preferred\_username}

Used to map IdP claims to the username, e.g. \${sub}

Additional scopes

phone ✕ email ✕ address ✕ profile ✕

✖ ▾

The default scope is 'openid'. Add more scopes if needed to obtain the username claim.

Redirect URL  
 Copy it to casdoor

Location where the client is sent to after successful account authentication.

Initiate login URL

URL used for OpenID Provider-initiated login.

**Additional settings**  
The authorization, token, and user info endpoints will be filled automatically if your Identity provider offers this option. If not, you will be asked to provide this information.

Fill the data automatically from my chosen identity provider.

**JIT provisioning**  
Just-in-time user provisioning allows users to be created and updated automatically when they log in through SSO to Atlassian Data Center applications. [Learn more](#).

Create users on login to the application

**OpenID Connect behaviour**

Remember user logins  
If checked, successful login history will be saved and users will be logged in automatically without the need for reauthentication.

**Login page settings**  
Decide if the IdP should be visible on login page and customize what the user will see on the button.

Show IdP on the login page

Login button text \*

The text is shown to the user on the login page. Remaining characters: 33.

Save configuration Cancel

2. You can configure more complex authorization later, for now check if OpenID actually works.

⚠ You have temporary access to administrative functions. [Drop access](#) if you no longer require it. For more information, refer to the [documentation](#).



Dashboards ▼ Projects ▼ Issues ▼ Boards ▼ Plans ▼ Create

Search



## Administration

Search Jira admin

Applications Projects Issues Manage apps User management Latest upgrade report System

General configuration

[Find more admin tools](#)

Jira mobile app

SYSTEM SUPPORT

System info

Instrumentation

Monitoring

Database monitoring

Integrity checker

Logging and profiling

Scheduler details

Troubleshooting and support tools

Clean up

Audit log

Clustering

SECURITY

Project roles

Global permissions

### Authentication methods

Add configuration

Manage how users authenticate. Save authentication configurations using SAML, OpenID Connect, or Crowd as the identity provider. [Learn more about using multiple identity providers.](#)

#### ⚠ Make authentication safer

Authenticating with username and password is less secure than through single sign-on. Now that you've configured the latter, consider disabling product login form and basic authentication.

Communicate this change to your users.

[How to disable](#) - Dismiss

#### Login options

Name	Type	Last updated	Show on login page	Actions
Username and password	Product login form	Never	<input checked="" type="checkbox"/>	<span style="font-size: small;">...</span>
casdoor	OpenID Connect	26 April 2023 7:20 PM	<input checked="" type="checkbox"/>	<span style="font-size: small;">...</span>

#### Authentication on API calls



Allow basic authentication on API calls.

You can use personal access tokens as a safer alternative method of authentication. See [Using personal access tokens](#).

# Via miniOrange Plugin

This is a tutorial on using [miniOrange](#) to connect casdoor and jira

[Casdoor](#) 可以使用 OIDC 协议作为IDP 连接各种应用程序。 Here is a [Jira](#) tutorial.

以下是配置中的一些专有名词：

`CASDOOR_HOSTNAME`: 私有部署的Casdoor域名或IP。

`Jira_HOSTNAME`: 部署 Jira 的域名或 IP。

## 步骤1. 部署 Casdoor 和 Jira

首先，应该部署[Casdoor](#) 和 [Jira](#)。

成功部署后，您需要确保：

1. 设置 Jira URL(Plans → Administration → System → 常规配置) 为

The screenshot shows the Jira administration interface under the 'System' tab. On the left sidebar, there are sections for General configuration, Find more admin tools, Jira mobile app, SYSTEM SUPPORT, System info, Instrumentation, Monitoring, Database monitoring, Integrity checker, Logging and profiling, Scheduler details, and Troubleshooting and support tools. The main content area is titled 'General configuration' and contains a 'Settings' section. Under 'General Settings', there are fields for 'Title' (set to 'JIRA'), 'Mode' (set to 'Private'), 'Maximum Authentication Attempts Allowed' (set to '3'), 'CAPTCHA on signup' (set to 'OFF'), and 'Base URL' (set to 'http://localhost:8080'). A red arrow points from the text 'Jira\_HOSTNAME' in the previous step to the 'Base URL' field. Below the 'Base URL' field, there are fields for 'Email from' (set to '\${fullname} (Jira)') and 'Introduction'. At the bottom, there is a 'Internationalization' section.

2. Casdoor 可以正常登录使用。
3. 您可以设置 `CASDOOR_HOSTNAME = http://localhost:8000`。在 `prod` 模

式下部署Casdoor。详见 [生产模式](#)。

## 步骤2. Configure Casdoor application and Jira

1. 创建或使用现有的 Casdoor 应用程序。
2. You should install a [miniOrange](#) app to support OAuth. You can find this app in Plans->Administration->Find new apps->search

The screenshot shows the Atlassian Marketplace for JIRA interface. A red arrow points to the 'Manage apps' tab at the top. Another red arrow points to the 'Find new apps' button. A third red arrow points to the search bar containing 'OAuth'. The search results list the 'mO Jira OAuth SSO, Jira OpenID Connect SSO, Jira OIDC SSO' app by miniOrange. This app has a 5-star rating and 607 installations. It is categorized under ADMIN TOOLS, INTEGRATIONS, JIRA SERVICE DESK, JIRA SOFTWARE, SECURITY, and UTILITIES. A red arrow points to the app name. Below the app listing, a note states: 'Login to Jira & Service Desk using OAuth2.0/OpenID Connect (OIDC) compliant applications like Google apps, AWS Cognito, Azure AD, Keycloak, GitHub, GitLab, Discord, Facebook, Microsoft, Meetup and custom apps. Best OAuth SSO App In Market!'

3. 将 [选定的应用程序](#) 设置为自定义 OpenId
4. 查找重定向url:

The screenshot shows the miniOrange OAuth Configuration page. A red arrow points to the 'OAuth/OIDC Configurations' section. Within this section, a red arrow points to the 'Callback URL' field, which contains the value 'http://localhost:8080/plugins/servlet/oauth/callback'.

5. 添加您的重定向url:

## 6. 您应该配置此应用

Selected Application: **Custom Openid**

Provider ID: **5c881c25-2e02-42c9-af06-0a71e0beb516**

Custom App Name: casdoor

Client Id:<sup>\*</sup> 514e09591ee5554b16fe

Client Secret:<sup>\*</sup> e7f05b14a68fb23e526f08515aefb73bbab7814a

Scope:<sup>\*</sup> openid email profile address phone offline\_access

Authorize Endpoint:<sup>\*</sup> http://localhost:8000/login/oauth/authorize

Access Token Endpoint:<sup>\*</sup> http://localhost:8000/api/login/oauth/access\_token

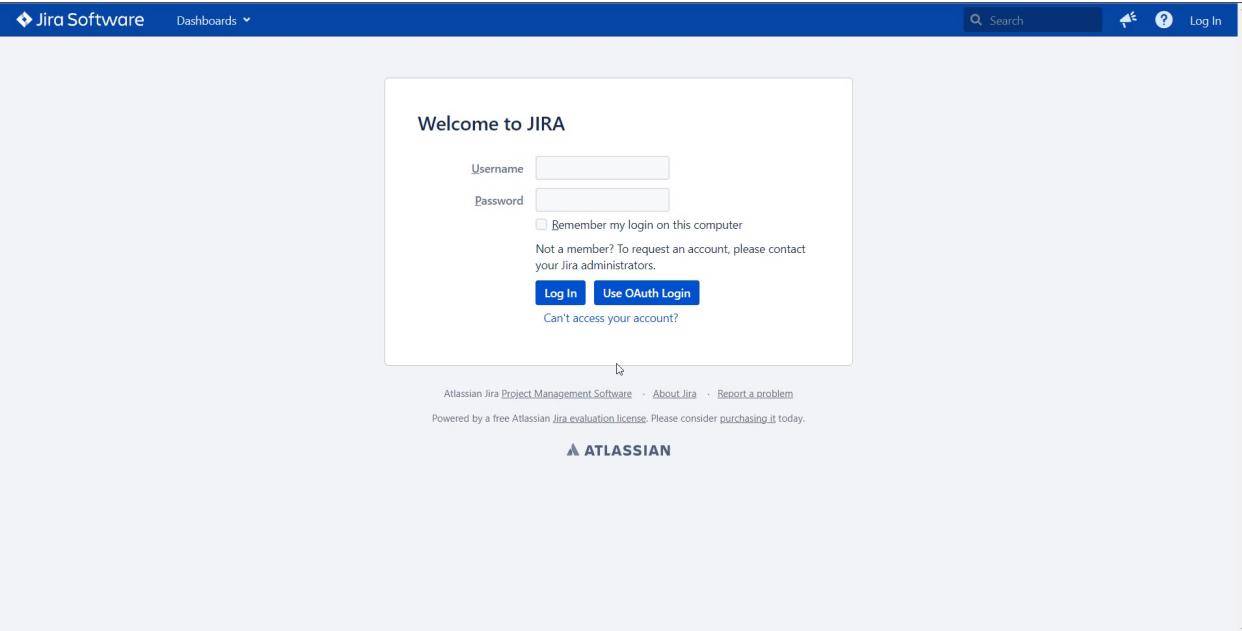
Logout Endpoint: Enter the Logout Endpoint URL

Save      Test Configuration

- Token server url: `http://CASDOOR_HOSTNAME/api/login/oauth/access_token`
- Authorization server url: `http://CASDOOR_HOSTNAME/login/oauth/authorize`
- User Info server url: `http://CASDOOR_HOSTNAME/api/get-account`
- Scopes: address phone openid profile offline\_access email

打开你喜欢的浏览器，访问: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`，你会看到 Casdoor 的 OIDC 配置。

退出Jira并测试SSO。



# Confluence

Casdoor can use OIDC protocol as IDP to connect various applications. Here we will use [Confluence](#) as an example to show you how to use OIDC to connect to your applications.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`Confluence_HOSTNAME`: Domain name or IP where Confluence is deployed.

## Step1. Deploy Casdoor and Confluence

Firstly, the [Casdoor](#) and [Confluence](#) should be deployed.

After a successful deployment, you need to ensure:

1. Set Confluence URL to `Confluence_HOSTNAME`.

The screenshot shows the 'General Configuration' section of the Confluence administration interface. On the left, a sidebar lists various configuration options. The 'General Configuration' option is highlighted with a blue box and has a black arrow pointing to it from the left. On the right, the main content area is titled 'General Configuration' and contains a 'Site Configuration' section. Under 'Site Configuration', there is a 'Server Base URL' field set to 'http://localhost:8090'. A black arrow points to this field from the right side of the image.

2. Casdoor can be logged in and used normally.
3. You can set CASDOOR\_HOSTNAME = `http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find a redirect url:

The screenshot shows the 'OAuth/OIDC Configurations' page. On the left, a sidebar has a 'OAuth/OIDC Configurations' option highlighted with a blue box. On the right, the main content area is titled 'OAuth/OIDC Configurations'. It shows a 'Callback URL' field containing 'http://localhost:8090/plugins/servlet/oauth/callback'. A black arrow points to this field from the right side of the image.

3. Add a redirect url:

The screenshot shows the 'OAuth/OIDC Configurations' page. On the left, a sidebar has a 'OAuth/OIDC Configurations' option highlighted with a blue box. On the right, the main content area shows several configuration fields: 'Client ID' (014ae4bd048734ca2dea), 'Client secret' (f26a4115725867b7bb7b668c81e1f8f7fae1544d), 'Cert' (cert-built-in), and 'Redirect URLs'. The 'Redirect URLs' section includes a 'Redirect URLs' field with an 'Add' button and a 'Redirect URL' field containing 'http://localhost:8090/plugins/servlet/oauth/callback'. Black arrows point to each of these four fields from the right side of the image.

4. Add provider you want and supplement other settings.

Not surprisingly, you can get two values on the application settings page: `Client ID` and `Client secret` like the picture above, we will use them in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

## Step3. Configure Confluence

1. You should install a `miniOrange` app to support OAuth. You can find this app in

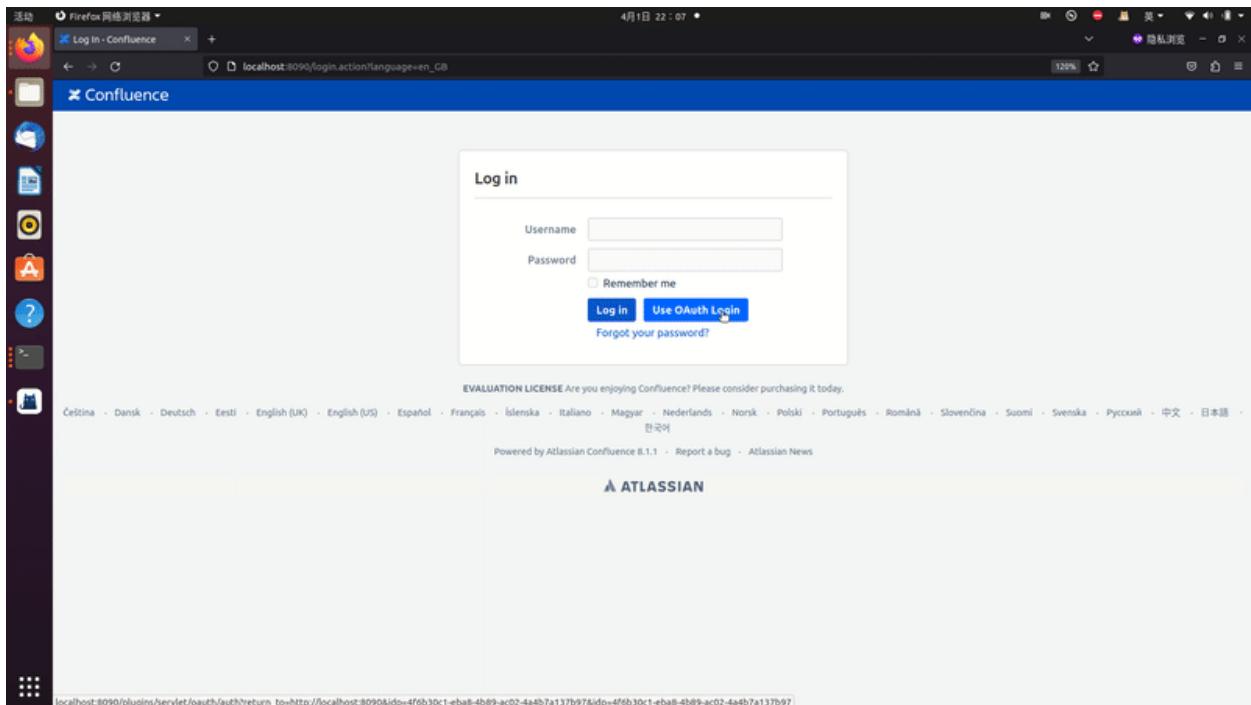
The screenshot shows the Atlassian Marketplace interface. On the left, there's a sidebar with various configuration options like 'Backup Administration', 'External Gadgets', and 'Find new apps'. Below the sidebar, there's a section titled 'ATLASSIAN MARKETPLACE' with a 'Find new apps' button highlighted by a black arrow. The main area is titled 'Find new apps' and shows search results for 'oauth'. A search bar contains 'oauth'. Below it are filters for '搜索结果', 'All categories', and 'All paid & free'. The first result is 'mO Confluence OAuth SSO, Confluence OpenID Connect/OIDC SSO' by miniOrange. This listing includes a star rating of 4.0 (40 reviews), 409 installations, and a '立即购买' (Buy Now) button. Below this is another app listing: 'Table Filter and Charts for Confluence' by Stiltsoft Europe OÜ, with a star rating of 3.47 (347 reviews), 15,179 installations, and a '立即购买' (Buy Now) button. The 'Find new apps' button in the sidebar is also highlighted by a black arrow.

2. You should config this app

Selected Application:	<b>Custom OpenId</b>	<b>Import Details</b>	<b>Setup Guide</b>
Provider ID:	<b>4f6b30c1-eba8-4b89-ac02-4a4b7a137b97</b>		
Custom App Name:	* Casdoor SSO		
Client Id:	* 014ae4bd048734ca2dea		
Client Secret:	* f26a4115725867b7bb7b668c81e1f8f7fae1544d		
Scope:	* openid profile email		
Authorize Endpoint:	* <a href="https://door.casdoor.com/login/oauth/authorize">https://door.casdoor.com/login/oauth/authorize</a>		
Access Token Endpoint:	* <a href="https://door.casdoor.com/api/login/oauth/access_token">https://door.casdoor.com/api/login/oauth/access_token</a>		
Logout Endpoint:	Enter the Logout Endpoint URL <small>Enter the Logout endpoint of your OAuth/OpenID Provider. Leave blank if Logout endpoint not supported by provider. e.g. If Keycloak Logout endpoint is configured with {hostname}/auth/realm/{realm-name}/{protocol}/openid-connect/logout URL then on!</small>		

3. Set **Selected Application** to Custom OpenId
  4. You can find Client Id and Client Secret in Casdoor application page.
- **Token server url**: `http://CASDOOR_HOSTNAME/api/login/oauth/access_token`
  - **Authorization server url**: `http://CASDOOR_HOSTNAME/login/oauth/authorize`
  - **UserInfo server url**: `http://CASDOOR_HOSTNAME/api/get-account`
  - **Scopes**: `address phone openid profile offline_access email`
1. You can configure more complex authorization later, for now check if OpenID actually works.

Log out of Confluence, and test SSO.



# RuoYi

Casdoor可以轻易连接到 RuoYi-cloud。

## 步骤1. 部署Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

在成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 步骤2. 配置Casdoor

如何配置casdoor可以参考 [casdoor](#)(配置casdoor和开发最好使用不同的浏览器)。

您还应该配置一个组织、应用程序和同步器。 您也可以参考 [Casdoor](#)。

需要注意的一些要点：

1. 修改同步器表格中的列

Table columns	Add	Column name	Column type	Casdoor column	Is hashed	Action
user_id	integer	Id	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
dept_id	integer	Affiliation	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
user_name	string	Name	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
nick_name	string	DisplayName	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
user_type	string	Type	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
email	string	Email	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
phonenumber	string	Phone	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
sex	string	Gender	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
avatar	string	Avatar	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
password	string	Password	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
de_flag	string	IsDeleted	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
login_ip	string	CreatedIp	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
create_time	string	CreatedTime	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
password	string	Password	✓	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

## 2. 修改组织中的密码类型

Password type [?](#):

## 3. 您也应该打开软删除。

# 步骤3. 前端改造

## 3.1 跳转到casdoor的登录页

我们可以前端的sdk，这里我们以vue-sdk为例。在加入vue-sdk后，您可以通过getSignInUrl() 获得casdoor登录页面

您可以以您喜欢的方式链接到它，并且您可以删除一些不再使用的ruoyi-cloud的代码，如账户和密码EL输入框。

## 3.2 接收casdoor返回的code和state

在我们通过casdoor成功登录后，casdoor会把code和state返回给我们创建的页面。我们可以调用create函数获取code和state。

```
created() {
  let url = window.location.href // get url
  let u = new URL(url);
  this.loginForm.code = u.searchParams.get('code') // get code and state
  this.loginForm.state = u.searchParams.get('state')
```

对于RuoYi-Cloud，我们只是将发送的账户和密码更改为code和state。因此，相对于原始登录，它只是改变发送到后端的内容。

## 步骤4. 后端改造

### 4.1 接收前端返回的code和state

```
@PostMapping("login")
public R<?> callback(@RequestBody CodeBody code) { //we should define a
CodeBody entity which have code and state
    String token = casdoorAuthService.getOAuthToken(code.getCode(),
code.getState());
    CasdoorUser casdoorUser = casdoorAuthService.parseJwtToken(token);
    if(casdoorUser.getName()!=null){
        String casdoorUserName = casdoorUser.getName();

        if(sysLoginService.getUserByCasdoorName(casdoorUserName)==null){ //if
database haven't this user
            // add this user into database
            sysLoginService.casdoorRegister(casdoorUserName);
        }
    }
    LoginUser userInfo =
    sysLoginService.casdoorLogin(casdoorUser.getName()); //get this user's
information by database
    return R.ok(tokenService.createToken(userInfo));
}
```

在这种方法中，我们使用 casdoor-SpringBoot-sdk，并对 RuoYi-Cloud 方法稍做修改。

例如，RuoYi-Cloud原本的register使用账户和密码注册，我将register改成了casdoorRegister。

我还添加了一个getUserByCasdoornname这样的方法来检查账户是否存在，并将基于账户和密码用户切换成了当前用户。

这很简单，因为我们只需要删除检查密码的部分。

# 步骤5. 总结

## 5.1 前端

- 我们需要删除原始的登录和注册功能。
- 我们还需要接受code和state并发送到后端。

## 5.2 后端

RuoYi 后端具有极好的登录和注册功能。 我们只需要稍做改变，因此非常方便。

# 步骤6. 详细步骤

1. 部署和配置Casdoor。 部署并配置casdoor。我们必须注意组织的密码类型必须为bcrypt，因为 RuoYi-Cloud的密码类型是bcrypt。
2. 我们应该使用casdoor的同步器将数据库用户中的用户信息复制到您的casdoor组织。 此步骤可以将原始帐户导入到casdoor。
3. 在我们部署了casdoor之后，我们应当改造前端。 我们应该关闭 RuoYi 的验证码功能

```
// checkcode switch  
captchaEnabled: false,  
// register switch  
register: true,
```

Note that RuoYi-Cloud captcha needs to be close in nacos again. Note that RuoYi-Cloud open registration function requires changing sys.account.registerUser to true.

4. We should add button jump to casdoor and change data's loginForm

```
</div><div class="button">  
    <a href="http://localhost:7001/login/oauth/authorize?client_id=d509b6b3edc8a3d4cce9&response_type=code&redirect_uri=http%3A%2F%2Flocalhost%3D8080%2Fcasdoor">casdoor</a>
```

```
        loginForm: {
            code: '',
            state: ''
        },
    }
```

Here I write url, you can get url by casdoor-vue-sdk or casdoor-SpringBoot-sdk.

5. 由于我们不使用原先的登录功能，我们应该删除关于 cookie 和验证码的方法。

所以创建新的函数：

```
created() {
    let url = window.document.location.href//get url
    let u = new URL(url);
    this.loginForm.code = u.searchParams.get('code')//get code and state
    this.loginForm.state = u.searchParams.get('state')
    if(this.loginForm.code!=null&&this.loginForm.state!=null){//if code and state is null, execute handleLogin
        this.handleLogin()
    }
}
```

6. In fact, we just need to change the parameter we send to back-end and delete the function we don't need, we don't need to change anything else.

```
handleLogin() {
    console.log("进入handleLogin")
    this.$store.dispatch("Login", this.loginForm).then(() => {
        this.$router.push({ path: this.redirect || "/" }).catch(()=> {});
    }).catch(() => {
        this.loading = false;
        if (this.captchaEnabled) {
            this.getCode();
            console.log(this.getCode)
        }
    });
}
```

```
  Login({ commit }, userInfo) {
    const code = userInfo.code
    const state = userInfo.state
    return new Promise((resolve, reject) => {
      login(code, state).then(res => {
        console.log("LOGIN")
        let data = res.data
        setToken(data.access_token)
        commit('SET_TOKEN', data.access_token)
        setExpiresIn(data.expires_in)
        commit('SET_EXPIRES_IN', data.expires_in)
        resolve()
      }).catch(error => {
        reject(error)
      })
    })
  },

```

```
export function login(code, state) {
  return request({
    url: '/auth/login',
    headers: {
      isToken: false
    },
    method: 'post',
    data: {code, state}
  })
}
```

## 7. 在后端导入依赖

pom.xml

```
<dependency>
  <groupId>org.casbin</groupId>
  <artifactId>casdoor-spring-boot-starter</artifactId>
  <version>1.2.0</version>
</dependency>
```

您还需要在代码上配置casdoor。

8. 回调函数被定义为重定向函数。我对sysLoginService 中的一些方法做了小修改 删除检查密码步骤，因为我们不需要它。

```
@PostMapping("login")
public R<?> callback(@RequestBody CodeBody code) {//we should define
    a CodeBody entity which have code and state
    String token = casdoorAuthService.getOAuthToken(code.getCode(),
code.getState());
    CasdoorUser casdoorUser =
casdoorAuthService.parseJwtToken(token);
    if(casdoorUser.getName()!=null){
        String casdoorUserName = casdoorUser.getName();

        if(sysLoginService.getUserByCasdoorName(casdoorUserName)==null){//if
database haven't this user
            // add this user into database
            sysLoginService.casdoorRegister(casdoorUserName);
        }
    }
    LoginUser userInfo =
sysLoginService.casdoorLogin(casdoorUser.getName());//get this
user's information by database
    return R.ok(tokenService.createToken(userInfo));
}
```

9. SysLoginService的新方法

```
public LoginUser casdoorLogin(String username){
    // execute user
    R<LoginUser> userResult =
remoteUserService.getUserInfo(username, SecurityConstants.INNER);
    if (R.FAIL == userResult.getCode())
    {
        throw new ServiceException(userResult.getMsg());
    }
}
```

```
public String getUserByCasdoorName(String casdoorUsername){  
    R<LoginUser> userResult =  
    remoteUserService.getUserInfo(casdoorUsername,  
    SecurityConstants.INNER);  
    if (StringUtils.isNull(userResult) ||  
    StringUtils.isNull(userResult.getData()))  
    {  
        //if this user is not in Ruoyi-Cloud database and casdoor  
        have this user, we should create this user in database  
        return null;  
    }  
    String username =  
    userResult.getData().getSysUser().getUserName();  
    return username;  
}
```

```
public void casdoorRegister(String username){  
    if (StringUtils.IsAnyBlank(username))  
    {  
        throw new ServiceException("User must fill in");  
    }  
    SysUser sysUser = new SysUser();  
    sysUser.setUserName(username);  
    sysUser.setNickName(username);  
    R<?> registerResult =  
    remoteUserService.registerUserInfo(sysUser, SecurityConstants.INNER);  
    System.out.println(registerResult);  
    if (R.FAIL == registerResult.getCode())  
    {  
        throw new ServiceException(registerResult.getMsg());  
    }  
    recordLogService.recordLoginInfor(username, Constants.REGISTER,  
    "register successfully");  
}
```

# Pulsar-manager

Casdoor可以轻易连接到Pulsar-manager。

因为 Pulsar-manager中已经添加了连接casdoor 的代码，所以我们只需要在后端的 application.yml中做一些配置，并在前端打开一个开关。

## 步骤1. 部署Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

在成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 步骤2. 配置Casdoor

如何配置casdoor可以参考 [casdoor](#)(配置casdoor和开发最好使用不同的浏览器)。

您还应该配置一个组织和一个应用程序。 您也可以参考 [Casdoor](#)。

## 步骤2.1 创建一个组织

Edit Organization Save Save & Exit

Name ⓘ: pulsar

Display name ⓘ: pulsar

Favicon ⓘ: URL ⓘ: <https://cdn.casbin.org/img/favicon.png>

Preview: 

Website URL ⓘ: <http://localhost:9527/#/login?redirect=%2F>

Password type ⓘ: plain

Password salt ⓘ:

Phone prefix ⓘ: + 86

## 步骤2.2 创建一个应用程序

Name ⓘ: app-pulsar

Display name ⓘ: app-pulsar

Logo ⓘ: URL ⓘ: [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home ⓘ: [/](#)

Description ⓘ:

Organization ⓘ: pulsar

Client ID ⓘ: 6ba06c1e1a30929fdda

Client secret ⓘ: df926bf913225ebbae9af7ba8d41fe19507eb079

Cert ⓘ: cert-built-in

Redirect URLs ⓘ: Add

Redirect URLs	Action
<a href="http://localhost:9527/callback">http://localhost:9527/callback</a>	  

## 步骤3. 打开 Pulsar-Manager 前端中的开关。

打开此开关，将代码和状态发送到后端。

这个开关在pulsar-manager/front-end/src/router/index.js的第80行

```
- // mode: 'history', // require service support
+ mode: 'history', // require service support
```

## 步骤4. 后端配置

您需要在pulsar-manager/src/main/resources/application.properties的154行加入casdoor的配置

```
casdoor.endpoint = http://localhost:8000
casdoor.clientId = <client id in previous step>
casdoor.clientSecret = <client Secret in previous step>
casdoor.certificate=<client certificate in previous step>
casdoor.organizationName=pulsar
casdoor.applicationName=app-pulsar
```

# ShenYu

ShenYu 可通过插件使用 casdoor。

## 步骤1. 部署Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 步骤2. 配置Casdoor应用程序

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加您的重定向url

Name : app-test → application name

Display name : app-test

Logo : 

Preview:

Home :  → organization name

Description :

Organization : built-in → organization name

Client ID : 6e3a84154e73d1fb156a → client id

Client secret : 84209d412a338a42b789c05a3446e623cb7262d → client secret

Cert : cert-built-in

Redirect URLs : → redirect url

Action	Redirect URL
	http://localhost:9195/http/hi
	http://localhost:9195/http/hello

3. 在证书编辑页面上，您可以看到您的 **证书**。

Certificate : Copy certificate Download certificate

```
-----BEGIN CERTIFICATE-----
MIIE+TCCAUgGAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTABBgNVBAoTFENh
c2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENlcnQwHhcNMjEx
MDE1MDgxMTUyWhcNNDExMDE1MDgxMTUyWjA2M0RwGwYDVQQKExRDYXNkb29yIEx9
Z2FuaxphdGlvbjEVMBMGAGUEAxMMQ2FzZG9vcj8DZXj0MIIcjANBgkqhkiG9w0B
AQEFAAOCAg8AMIICCgKCAgEAisInpb5E1/ymf0f1RfSDSSE8I7y+lw+RjIj74e5ej
rq4b8zMMyk7HeHCyZr/hmNewEVXnhXu1P0mBeQ5ypp/QGo8vgEmjAETNmzkl1NjOQ
CjCYwUrasO/f/Mnl1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07
uvFMCJe5W8+0rKErZCKTR8+9VB3janeBz/zQePFvh79bfZate/hLirPK0Go9P1g
OvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRS8b/wUjNCUBD
PTSLVjc04WIIIf6Nkfx0Z7KvmbPstSj+btvqcsvRAGtvsB9h62Kptjs1Yn7GAuo
I3qt/4zoKbiURYxkQJXlvwCQsEftUuk5ew5zuPSIDRLoLByQTLbx0jLAFNfW3g/
pzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4Gu
hRsnlsmkmUDeylZ9aBL9oj11YEQfM2JZEq+RvtUx+wB4y8K/tD1bcY+lfnG5rBpw
IDpS262boq4SRsvb3Z7bB0w4ZxvOfj/1VLoRftPbLifobhfr/AeZMHpiKOXvfz4
yE+hqzi6wdF0VR9xYc/RbSAf7323OsjYnjjEglnUtRohnRgCpjlk/Mt2Kt84Kb0
wn8CAwEAAaMQMA4wDAYDVR0TAQH/BAlwADANBgkqhkiG9w0BAQsFAAOCAgEAn2lf
DKkLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgic4/LSdzuf4Awe6ve
C06lVdWSlis8UPUPdjmt2uMPSNjwLxG3QsrimMURNlwFLTfRem/heJe0Zgur9J1M
8haawdSdjH2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCvl
2nR42Fybap3O/g2JXMhNNRowZmNjgpsF7XVENCSuFO1jTywLaqjuXCg54lL7XVLG
omKNNNcc8h1FCelj/nbbGMhodnFWKDTsJcbNmcOPNHo6ixzqMy/Hqc+mWv7maAG
Jtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPnOBLql
2DJ1zaBmjIGolvb7XNVKcUfdXYw85ZTQ5b9cl4e+6bmyWqQltlwt+Ati/uFEV
Xzcj70B4IALX6xau1kLepV9O1GERizYrz5P9NJNA7Ko05AVMp9w0DQTkt+LbXnZE
HHnWKy8xHQKF9sR7YBPGLs/Ac6tviv5ua15Ogj/8dLRZ/veyFfGo2yZsl+hKVU5
nCCJHBcAyFnm1hdvdwEdH33jDBjNB6ciotJZrf/3VYalWSalADosHAgMWfxUWP+h
8XKXmzlxuHbTMQYtZPDgsps5aK+S4Q9wb8RRAYo=
-----END CERTIFICATE-----
```

## 步骤3. 在 shenyu 中使用casdoor插件

### 1. 在 shenyu 中配置casdoor插件

## Plugin

X

\* Plugin: casdoor

### casdoor Configuration

\* application-name: app-test

\* certificate: -----BEGIN CERTIFICATE-----\nMIIE+TCCAuGgAwI

\* client\_id: 6e3a84154e73d1fb156a

\* client\_secrect: a4209d412a33a842b7a9c05a3446e623cbb7262d

\* casdoor endpoint: http://localhost:8000

\* organization-name: test

\* Role: Authentication

\* Sort: 40

Status:

Cancel

Sure

注意：由于shenyu 仅有单行输入框，所以我们需要在每行证书中添加 \n

Certificate  :

[Copy certificate](#)

[Download certificate](#)

```
-----BEGIN CERTIFICATE-----\nMIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENh\n\nc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENIcnQwHhcNMjEx\n\nc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENIcnQwHhcNMjEx\nMDE1MDgxMTUyWhcNNDEzMDE1MDgxMTUyWjA2MR0wGwYDVQQKExRDYXNkb29yIE9y\nZ2FuaXphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vcIBDZXJ0MIICljANBqkqhkiG9w0B\nAQEFAOCAG8AMIICgKCAgEAsInpb5E1/yM0f1RfSDSSE8IR7y+lw+RJjl74e5ej\nrq4b8zMYk7HeHCyZr/hmNEwEVXnhXu1P0mB8eQ5yp/PGo8vgEmjaETNmzkl1NjOQ\nCjCYwUrasO/f/Mnl1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07\nuvFMCje5W8+0rKErZCKTR8+9VB3janeBz//zQePFVh79bFZate/hLirPK0Go9P1g\nOvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRSBb/wUjNCUBD\nPTSLVjC04WIISf6Nkfx0Z7KvmbPstSj+btvcqsvRAGtvdsB9h62Kptjs1Yn7GAuo\nl3qt/4zoKbiURYxkQJXlvwCQsEftUuk5ew5zuPSIDRLoLByQTLbx0jqLAFnfW3g/\npzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMcdfx4Gu\nhRsnLsmkmUDeylZ9aBL9oj11YEQfM2JZEq+RVtUx+wB4y8K/tD1bcY+lfnG5rBpw\nIDpS262boq4SRVb3Z7b0w4ZxvOfj/1VLoRftjPbLlf0bhfr/AeZMHplKOXvfz4\nyE+hqzi68wdF0VR9xYc/RbSAf7323OsjYnjEgInUtRohnRgCpjlk/Mt2Kt84Kb0\nwn8CAwEAQMA4wDAYDVR0TAQH/BAIwADANBqkqhkiG9w0BAQsFAAOCAgEAn2If\nDKkLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/LSdzuf4Awe6ve\nC06IVdWSlis8UPUPdjmT2uMPSNjwLxG3QsrimMURNwFILTfRem/heJe0Zgur9J1M\n8haawdSdjH2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCvl\n2nR42Fyap3O/g2JXMhNNROwZmNjgpsF7XVENCSuFO1jTywLaqjuXCg54IL7XVLG\nomKNNNcc8h1FCeKj/nnbGMhodnFWKDTsJcbNmcOPNHo6ixzqMy/Hqc+mWYv7maAG\nJtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPnOBLql\n2DJ1zaBmjIGjolvb7XNVKcUfDXYw85TZQ5b9cl4e+6bmyWqQltlw+Ati/uFEV\nXzCj70B4IALX6xau1kLEpV9O1GERizYRz5P9NJNA7KoO5AVMp9w0DQTkt+LbXnZE\nHHnWKy8xHQKZF9sR7YBPGLs/Ac6tviv5ua15OgJ/8dLRZ/veyFfGo2yZsl+hKVU5\nnCCJHBcAyFnm1hdvdwEdH33jDBjNB6ciotJZrf/3VYalWSaiADosHAgMWfXuWP+h\n8XKXmzlxuHbTMQYtZPDgspS5aK+S4Q9wb8RRAYo=\n-----END CERTIFICATE-----
```

 here not need add \n

你可以复制并粘贴到Shenyu 后台配置的证书。

**您不需要将其保存到casdoor证书编辑页面，因为只需复制即可。**

## 2. 配置shenyu casdoor的插件

The screenshot shows the Apache Shenyu Gateway Management System interface. On the left, there is a sidebar with a dark theme containing the following sections: Change Mode, PluginList (Mock, Cache, Authentication, Sign, Jwt, Casdoor), and a central content area for RulesList. The RulesList section has tabs for SelectorList and RulesList, with the latter being active. It includes search fields for RuleName and Operation, and buttons for Query and Add. A table displays rule configurations, with one row highlighted: RuleName /http/ (Open), Operation Modify\_Delete, and UpdateTime 2022-09-28 10:50:02.729. Below the table are navigation buttons (1, <, >, 12 / page).

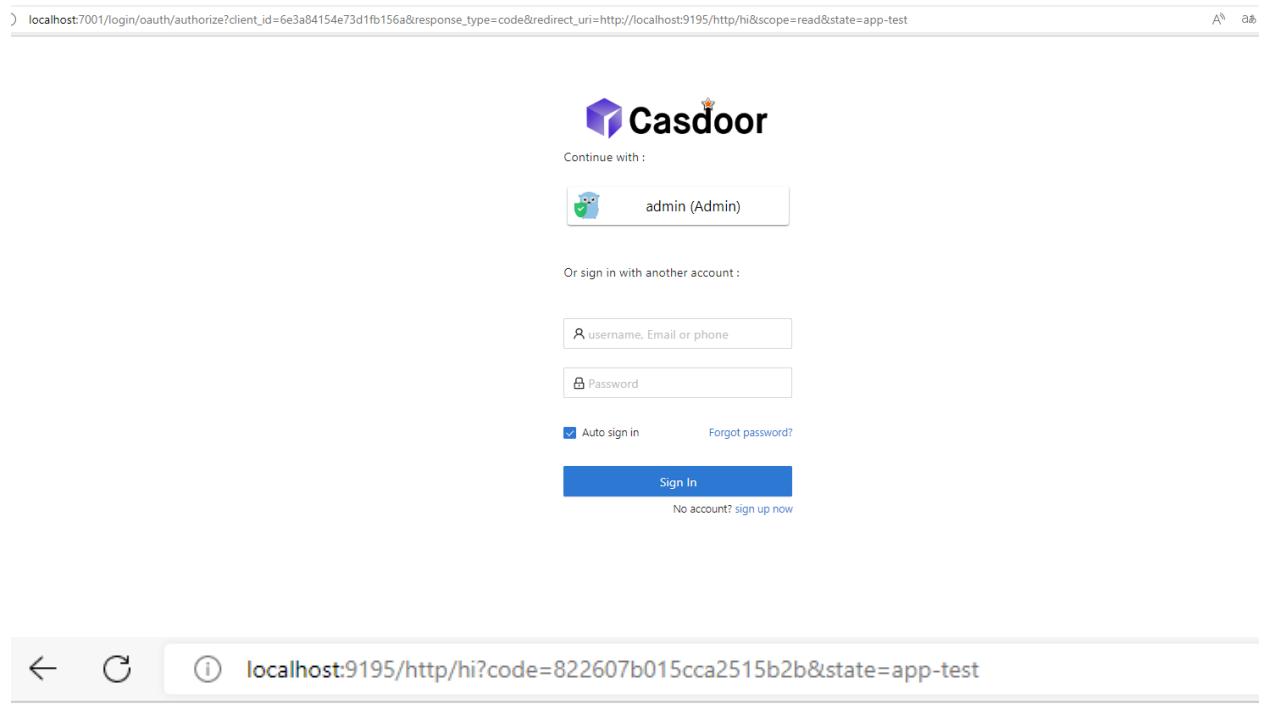
您可以配置您需要使用casdoor配置

## 3. 使用Casdoor提供的相关服务

### 3.1 直接访问Web

The screenshot shows a browser window with the address bar containing "localhost:9195/http/hi". The page content displays a JSON error response: {"code":401, "message":"Illegal authorization"}

### 3.2像这样登录casdoor



The screenshot shows the Casdoor login interface. At the top, there is a URL bar with the address `localhost:7001/login/oauth/authorize?client_id=6e3a84154e73d1fb156a&response_type=code&redirect_uri=http://localhost:9195/http/hi&scope=read&state=app-test`. Below the URL bar is the Casdoor logo and the text "Continue with:". A button labeled "admin (Admin)" is shown, which has a small owl icon next to it. Below this, there is a section for "Or sign in with another account:" containing two input fields: one for "username, Email or phone" and one for "Password". There are also links for "Auto sign in" (with a checked checkbox) and "Forgot password?". At the bottom of the form is a large blue "Sign In" button. Below the "Sign In" button, there is a link "No account? sign up now".

localhost:9195/http/hi?code=822607b015cca2515b2b&state=app-test

hi! null! I'm Shenyu-Gateway System. Welcome!

### 3.3 Headers中的Carry token, 你也可以访问

The screenshot shows the Postman application interface. At the top, it displays a GET request to `http://localhost:9195/http/hi`. Below the URL, there are tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The Headers tab is currently selected, showing the following configuration:

Key	Description
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.29.2
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Authorization	eyJhbGciOiJSUzI1NjIiLmtpZCI6ImNlcnQtYn... <span style="color:red">Your token</span>

Below the headers, there are tabs for Body, Cookies (1), Headers (9), and Test Results. The Body tab is selected, showing the response body:

```
1 hi! null! I'm Shenyu-Gateway System. Welcome!
```

At the bottom right, there are buttons for Save Response, a copy icon, and a search icon.

### 3.4 它也可以保存 Headers 的用户名、id 和组织，以便下次使用。

# ShardingSphere

shardingsphere-elasticsearch-ui已经集成了Casdoor。配置后我们可以使用它。

## 步骤1. 部署 Casdoor

首先，部署Casdoor。

您可以参考Casdoor 官方文档 [Server Installation](#)。

成功部署后，您需要确保：

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- 输入 `admin` 和 `123` 测试登录功能正常工作。

然后您可以通过以下步骤在自己的应用程序中快速实现基于 Casdoor 的登录页面。

## 步骤2. 配置ShardingSphere 中的 cassdoor 应用程序并配置

1. 创建或使用现有的 Casdoor 应用程序

Name ⓘ: ShardingSphere

Display name ⓘ: ShardingSphere

Logo ⓘ: URL ⓘ: https://cdn.casbin.org/img/casdoor-logo\_1185x256.png

Preview: 

Home ⓘ:

Description ⓘ:

Organization ⓘ: ShardingSphere

Client ID ⓘ: 3ed79fa530645fb3653

Client secret ⓘ: 54633c82b7796a4332c6976864c6c16bc3b05556

Cert ⓘ: cert-built-in

Redirect URLs ⓘ:

Redirect URL	Action
http://localhost:8080	[Edit] [Delete]

重定向URL 取决于您需要重定向的网址。他选择的数据将在下次使用。

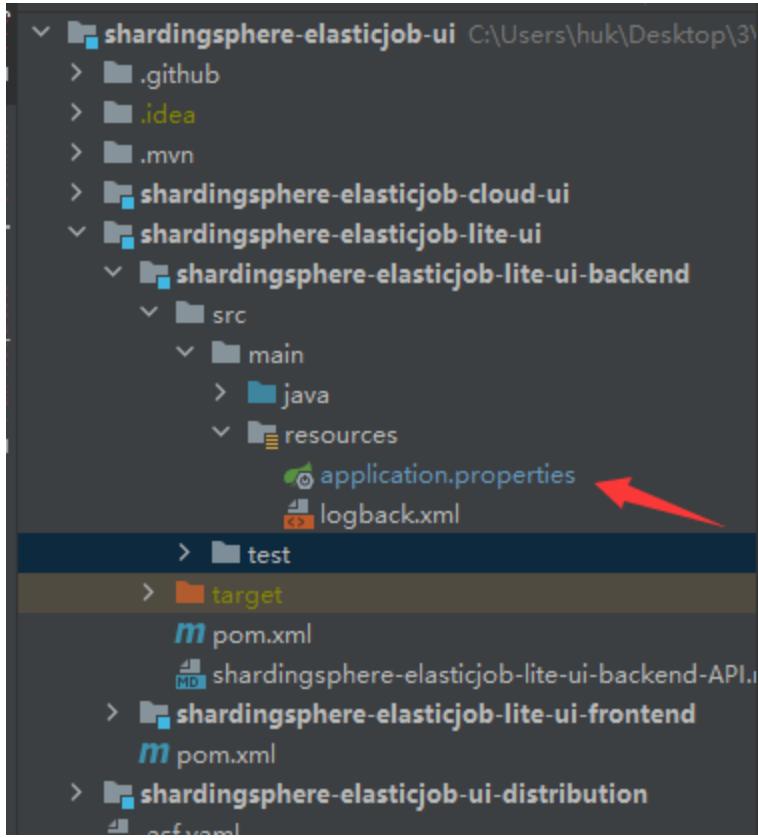
2. 在证书编辑页面上，您可以看到您的 **证书**。

Certificate [?](#) [Copy certificate](#) [Download certificate](#) Private

-----BEGIN CERTIFICATE-----  
MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENlcnQwHhcNMjExMDE1MDgxMTUyWhcNNExMDE1MDgxMTUyWjA2MR0wGwYDVQQKExRDYXNkb29yIE9yZ2FuaXphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vciBDZXJ0MIICljANBqkqhkiG9w0BAAQFAAACAg8AMIICCgKCAgEAsInpb5E1ym0f1RfSDSSE8IR7y+lw+RJi74e5ejrq4b8zMk7HeHCyZr/hmNewEVXnhXu1P0mBeQ5ypp/QGo8vgEmjAETNmzk1NjOQCjCYwUrasO/f/MnI1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07uvFMCJe5W8+0rKErZCKTR8+9VB3janeBz//zQePFVh79bFZate/hLirPK0Go9P1gOvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRSBb/wJNCUBDPTSLVjC04WIISf6Nkfx0Z7KvmbPstSj+btcqsvRAGtvdsB9h62Kptjs1Yn7GAuoI3qt/4zoKbiURYxkJXlvwCQsEftUuk5ew5zuPSIDRLoLByQTLbx0JqLAFNfW3gpzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4GuhRsnLsmkmUDeylZ9aBL9oj11YEQfm2JZEq+RVtUx+wB4y8K/tD1bcY+IfnG5rBpwIDps262boq4SRsvb3Z7b80w4ZxvOfj/1VLorftjPbLif0bhfr/AeZMHplKOXvf4yE+hqzi68wdF0VR9xYc/RbSAf7323OsjYnjjEgInUtRohnRgCpjlk/Mt2Kt84Kb0wn8CAwEAaMQMA4wDAYDVR0TAQH/BAIwADANBgkqhkiG9w0BAQsFAAACAgEAn2IfDKKLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/LSdzuf4Awe6veC06IVdWSlis8UPUPdmjT2uMPSNjwLxG3QsrimMURNwFILTfRem/heJe0Zgur9J1M8haawdSdJh2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCvI2nR42Fybap3O/g2JXMhNNROwZmNjgpsF7XVENCSuFO1jTywLaqjuXCg54IL7XVLGomKNNNcc8h1FCeKj/nnbGMhodnFWKDTsJcbNmcoOPNho6ixzqMy/Hqc+mWYv7maAGJtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPnOBLql2DJ1zaBmjIGjolvb7XNVKcUFDXYw85TZQ5b9cl4e+6bmyWqQltlw+Ati/uFEVXzCj70B4IALX6xau1kLEpV9O1GERizYRz5P9NJNA7KoO5AVMp9w0DQTkt+LbXnZEHHnWKy8xHQKZF9sR7YBPGls/Ac6tviv5Ua15OgJ/8dLRZ/veyFfGo2yZsl+hKVU5nCCJHBcAyFnm1hdvdwEdH33jDBjNB6ciotJZrf/3VYalWSalADosHAgMWfXuWP+h8XKKXmzlxuHbTMQYtZPDgspS5aK+S4Q9wb8RRAYo=-----END CERTIFICATE-----

### 3. 配置在 ShardingSphere 中的应用程序

首先，我们需要找到应用程序以及我们需要配置的属性。



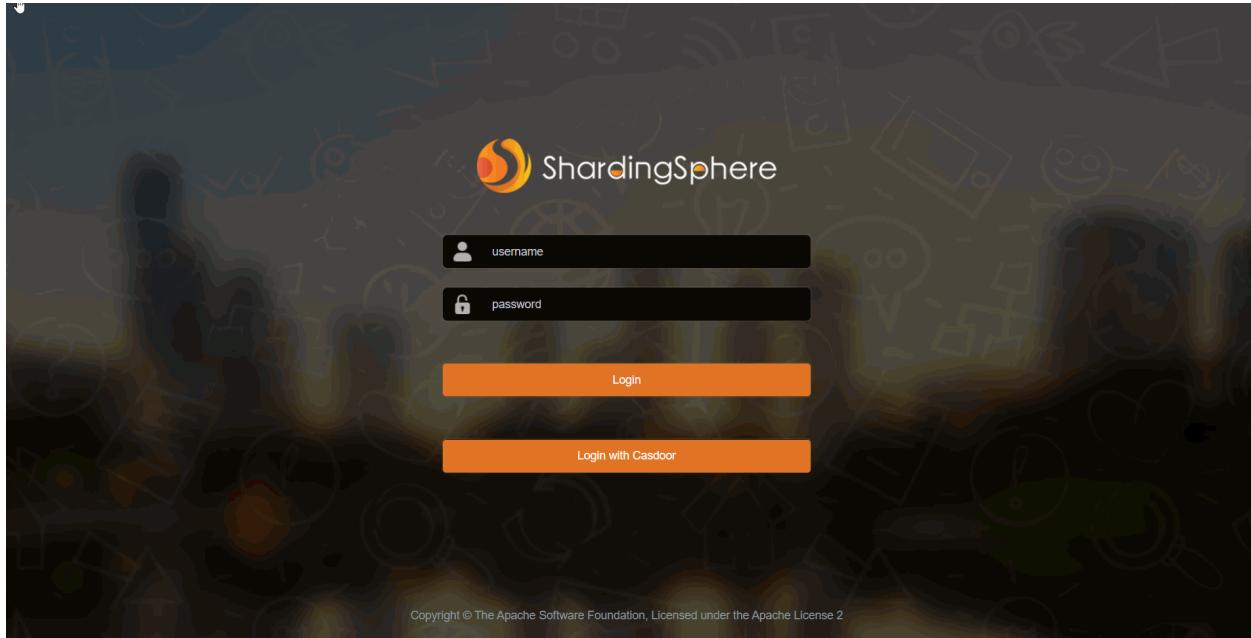
其次，我们需要复制Casdoor应用程序中的数据并将其粘贴到应用中。

```

casdoor.endpoint=http://localhost:7001
casdoor.client-id=3ed79fa530645fb3653
casdoor.client-secret=54633c82b7796a4332c6976864c6c16bc3b05556
casdoor.certificate=\n\
-----BEGIN CERTIFICATE-----\n\
MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENh\n\
c2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxYDXNkb29yIEEnlcnQwHhcNMjEx\n\
MDE1MDgxMTUyWhcNNDExMDE1MDgxMTUyWjA2MR0wGwYDVQQKExRDYXNkb29yIE9y\n\
Z2FuaXphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vcibDZXJ0MIICijANBqkghkiG9w0B\n\
AQEFAAOCAg8AMIICCgKCACgEAIsInpb5E1/ym0f1RfSDSSE8IR7y+lw+RjI74e5ej\n\
rq4b8zMYk7HeHCyZr/hmNewEVXnhXu1P0mBeQ5yp/QGo8vgEmjAETNmzkI1Nj0Q\n\
CjCYwUras0/f/MnI1C0j13vx6mV1kJHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07\n\
uvFMCJe5W8+0rKErZCKTR8+9VB3janeBz//zQePFVh79bFZate/hLirPK0Go9P1g\n\
0vwIoC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDfE7mtTstRSBb/wUjNCUBD\n\
PTSLVjC04Wllsf6Nkfx0Z7KvmbPstsj+btvcsrvAGtvdsB9h62Kptjs1Yn7GAuo\n\
I3qt/4zoKbiURYxkQJXIvwCQsEftUuk5ew5zuPSlDRLoLByQTLbx0JqLAFNfW3g/\n\
pzSDjqd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4Gu\n\
hRsnLsmkmUDeyIZ9aBL9oj11YEQfM2JZEeq+RvtUx+wB4y8K/tD1bcY+IfnG5rBpw\n\
IDpS262boq4SRSvb3Z7b0w4Zxv0fJ/1VLoRftjPbLIf0bhfr/AeZMHpIK0Xvfz4\n\
yE+hqzi68wdF0VR9xYc/RbSAf73230sjYnjjEgInUtRohnRgCpjIk/Mt2Kt84Kb0\n\
wn8CAwEEAaMQMA4wDAYDVR0TAQH/BAIwADANBqkghkiG9w0BAQsFAAOCAgEAn2lf\n\
DKkLX+F1vKR0/5gJ+PLr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/LSdzuf4Awe6ve\n\
C06LVdWSIis8UPUPdjmT2uMPSNjwLxG3QsrilmMURNwFLLTfRem/heJe0Zgur9J1M\n\
8●awdSdJjH2RgmFoDeE2r8NVRfhbR8KnC01ddTJKuS1N0/irHz21W4jt4rxzCvl\n\
2nR42Fybap30/g2JXMhNNR0wZmNjqpsF7XVENCSuF01jTywLaqjuXCg54IL7XVLG\n\
omKNNNcc8h1FCeKj/nnbGMhodnFWKDTsJcbNmcOPNHo6ixzqMy/Hqc+mWYv7maAG\n\
Jtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKft0PZgtH979XC4mdf0WPn0BLql\n\
2DJ1zaBmjigJolvb7XNVKcUfDXYw85TZQ5b9cLI4e+6bmyWqQItlw+Ati/uFEV\n\
XzCj70B4lALXxau1kLEpV901GERizYRz5P9NJNA7Ko05AVMp9w0DQTkt+LbXnZE\n\
HHnWKy8xHQKZF9sR7YBPGls/Ac6tviv5Ua150gJ/8dLRZ/veyFfGo2yZsI+hKVU5\n\
nCCJHBcAyFnm1hdvdwEdH33jDBjNB6ciotJZrf/3VYaIWSalADosHAgMWfXuWP+h\n\
8XKXmzlxuHbTMQYtZPDqspS5aK+S4Q9wb8RRAYo=\n\
-----END CERTIFICATE-----\n
casdoor.organization-name=ShardingSphere
casdoor.application-name=ShardingSphere

```

## 4. 测试



# Apache IoTDB

Casdoor can simply connect to [Apache IoTDB](#).

Because the code for connecting casdoor has been added in [Apache IoTDB Web Workbench](#), we just need to configure application.yml in back-end and open front switch.

## Step1. Deploy Casdoor

Firstly, the Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#).

After a successful deployment, you need to ensure:

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- Input `admin` and `123` to test login functionality is working fine.

Then you can quickly implement a Casdoor-based login page in your own app with the following steps.

## Step2. Configure Casdoor

Configure casdoor can refer to [casdoor](#)(Configure casdoor's browser better not use one browser as your develop browser).

You also should configure an organization and an application. You also can refer to [casdoor](#).

## 2.1 you should create an organization

Name ⓘ: IoTDB

Display name ⓘ: IoTDB

Favicon ⓘ: URL ⓘ: <https://cdn.casbin.org/img/favicon.png>

Preview: 

Website URL ⓘ: <https://door.casdoor.com>

Password type ⓘ: plain

Password salt ⓘ:

## 2.2 you should create an application

Name ⓘ: app\_IoTDB

Display name ⓘ: app\_IoTDB

Logo ⓘ: URL ⓘ: [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home ⓘ: [/](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 6f561b83d119be3e1e3c

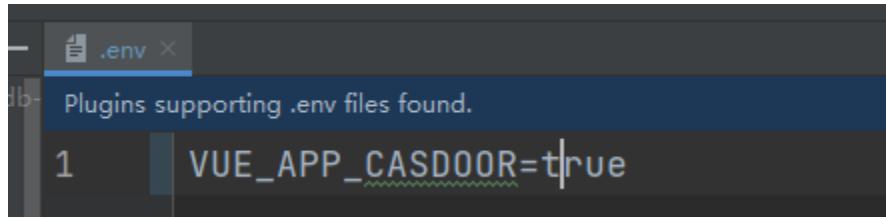
Client secret ⓘ: 242082e823b31a9b0d3a0a4a5a80cd5e415c75f

Cert ⓘ: cert-built-in

## Step3. Open Apache IoTDB Web Workbench front-end switch

Open this switch to make code and state send to back-end.

This switch in the iotdb-web-workbench/fronted/.env



A screenshot of a code editor showing a file named '.env'. The file contains the line 'VUE\_APP\_CASDOOR=true'. A status bar at the top says 'Plugins supporting .env files found.'

## Step4. Configure back-end code

You should configure casdoor's Configuration in the iotdb-web-workbench/backend/src/main/resources/application.properties

```
casdoor.endpoint = http://localhost:8000
casdoor.clientId = <client id in previous step>
casdoor.clientSecret = <client Secret in previous step>
casdoor.certificate=<client certificate in previous step>
casdoor.organizationName=IoTDB
casdoor.applicationName=app-IoTDB
```

# Result

The image is a composite of two screenshots. On the left, there is a photograph of a factory conveyor belt system, showing several large, dark cylindrical components and a bright yellow rectangular object being processed. On the right, there is a screenshot of the IoTDB WorkBench login page. The page has a header with the IoTDB logo and "WorkBench". Below the header, it says "Welcome To IoTDB WorkBench". There are two input fields: one for "Account" with the placeholder "Please Input Account" and one for "Password" with placeholder "\*\*\*\*\*". To the right of the password field is a "Forgot Password?" link. At the bottom of the page are two buttons: a teal "Sign In" button and a green "Sign In With Casdoor" button with a small circular icon.

# Apache DolphinScheduler

Casdoor is one of the supported login method for [Apache DolphinScheduler](#).

## Step1. Deploy Casdoor

Firstly, the Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#).

After a successful deployment, you need to ensure:

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, you will see the login page of Casdoor.
- Input admin and 123 to test login functionality is working fine.

Then you can quickly implement a Casdoor based login page in your own app with the following steps.

## Step2. Configure Casdoor Application

1. Create or use an existing Casdoor application.
2. Add Your redirect url (You can see more details about how to get redirect url in the next section)

Name [?](#) : application\_a6ftas → your application name

Display name [?](#) : New Application - a6ftas

Logo [?](#) : URL [?](#) : [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home [?](#) :

Description [?](#) :

Organization [?](#) : organization\_carg1b → your organization name

Client ID [?](#) : 3ed7314825ecf955cb19 → your client id

Client secret [?](#) : ee9314ea228... → your client secret

Cert [?](#) : cert-built-in

Redirect URLs [?](#) :

Redirect URLs	Add
Redirect URL	<a href="http://localhost:3000/callback">http://localhost:3000/callback</a> → your redirect url

### 3. Add provider you want and supplement other settings.

Not surprisingly, you can get two values on the application settings page: Client ID and Client secret like the picture above. We will use them in next step.

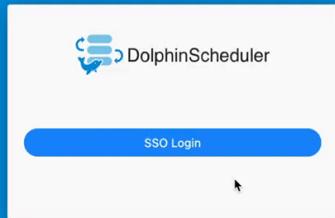
Open your favorite browser and visit: [http://CASDOOR\\_HOSTNAME/.well-known/openid-configuration](http://CASDOOR_HOSTNAME/.well-known/openid-configuration), you will see the OIDC configure of Casdoor.

# Step3. Configure Dolphinscheduler

| dolphinscheduler-api/src/main/resources/application.yaml

```
security:
  authentication:
    # Authentication types (supported types:
    # PASSWORD, LDAP, CASDOOR_SSO)
    type: CASDOOR_SSO
  casdoor:
    # Your Casdoor server url
    endpoint:
    client-id:
    client-secret:
      # The certificate may be multi-line, you can use `|-` for ease
      certificate:
        # Your organization name added in Casdoor
        organization-name:
        # Your application name added in Casdoor
        application-name:
        # Dolphinscheduler login url
        redirect-url: http://localhost:5173/login
```

Now, Dolphinschduler will automatically redirect you to Casdoor for authentication.



# FireZone

Casdoor can use OIDC protocol as IDP to connect various applications. Here we will use [FireZone](#) as an example to show you how to use OIDC to connect to your applications.

## Step 1. Deploy Casdoor and FireZone

Firstly, the Casdoor and FireZone should be deployed.

After a successful deployment, you need to ensure:

1. Set FireZone URL(Sigin → Security → Add OpenID Connect Provider) to FIREZONE\_HOSTNAME.

The screenshot shows the Firezone configuration interface. On the left is a sidebar with navigation links: Configuration (Users, Devices, Rules), Settings (Defaults, Account, Customization, Security), and Diagnostics (WAN Connectivity). The main area is titled "Site Settings" with the sub-section "Site Defaults". Under "Allowed IPs", the value is "172.21.0.0/16,172.16.0.0/16". Under "DNS Servers", the value is "172.16.250.155". Under "Endpoint", the value is "localhost:8080", which is highlighted with a red arrow and labeled "FIREZONE\_HOSTNAME". Other settings shown include "Persistent Keepalive" (0), "MTU" (1280), and a note about WireGuard interface MTU.

2. Casdoor can be logged in and used normally.
3. `CASDOOR_HOSTNAME`: <http://localhost:8000>. If you deploy Casdoor using default `app.conf`.

## Step 2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url:

For example, the Configid in the FireZone Provider is TEST, so the redirect URL should be `http://[FIREZONE_HOST]/auth/oidc/[PROVIDER_CONFIG_ID]/callback/`

Home ?:

Description ?:

Organization ?:

Client ID ?:

Client secret ?:

Cert ?:

Redirect URLs ?:  
    
  

Open your favorite browser and visit: [http://\[CASDOOR\\_HOSTNAME\]/.well-known/openid-configuration](http://[CASDOOR_HOSTNAME]/.well-known/openid-configuration), you will see the OIDC configure of Casdoor.

### 3. Configure FireZone, Security → Add OpenID Connect Provider

OIDC Config

Config ID  
TEST

Label  
TEST

Scope  
openid email profile

Response type  
code

Client ID  
0159c45127541d48e433

Client secret  
add1be9982640e048fcf46770d75674b918484af

Discovery Document URI  
http://localhost:8000/.well-known/openid-configuration

Auto create users

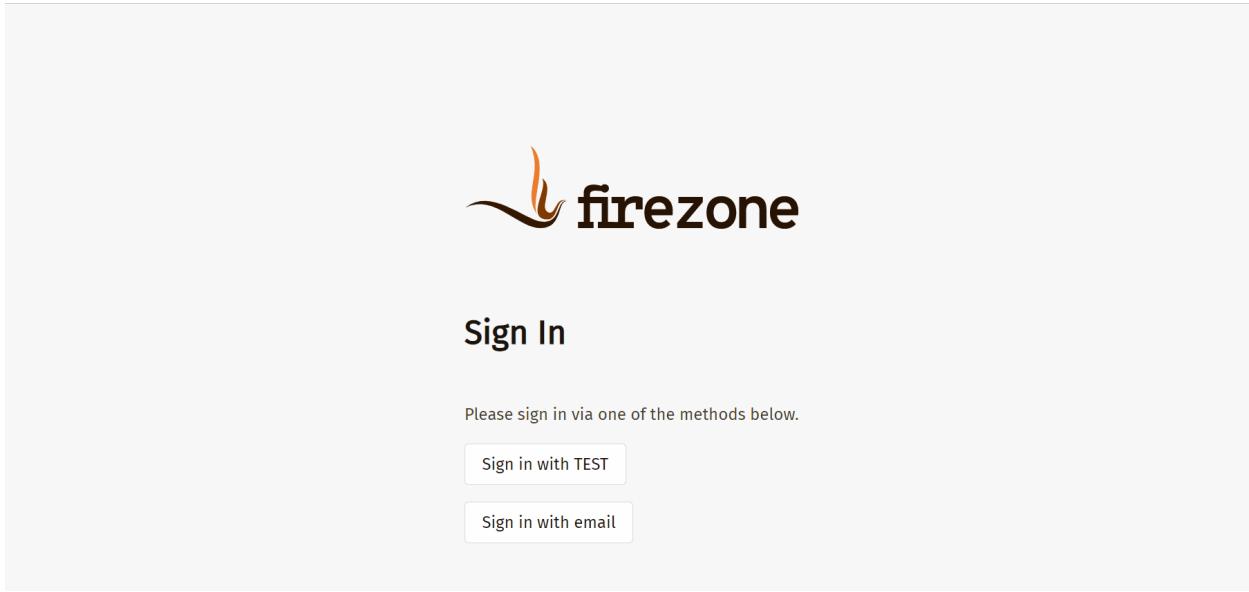
Save

- Discovery Document URI: FireZone Provider Discovery Document URI should be https://[CASDOOR\_HOST]/.well-known/openid-configuration
- Scopes: openid email profile
- ConfigID: ConfigID should be the PROVIDER\_COONFIG\_ID of the redirect

URL and should correspond to casdoor redirect URL

- `Auto create users`: Successful login will automatically create a user

## Log out of FireZone, and test SSO



# CloudFoundry

Before the integration, we need to deploy Casdoor locally.

Then we can quickly implement a Casdoor-based login page in our own app with the following steps.

## Step1. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url: `http://CASDOOR_HOSTNAME/login`

Redirect URLs	Action
http://localhost:8080/login	[Delete]

3. Copy the client ID, we will need it in the following steps.

## Step2. Add user in Casdoor

Now you have the application, but not a user. That means you need to create a user and assign the role.

Go to the “Users” page and click on “Add user” in the top right corner. That opens a new page where you can add the new user.

Save the user after adding a username and adding the organisation CloudFoundry(other details are optional).

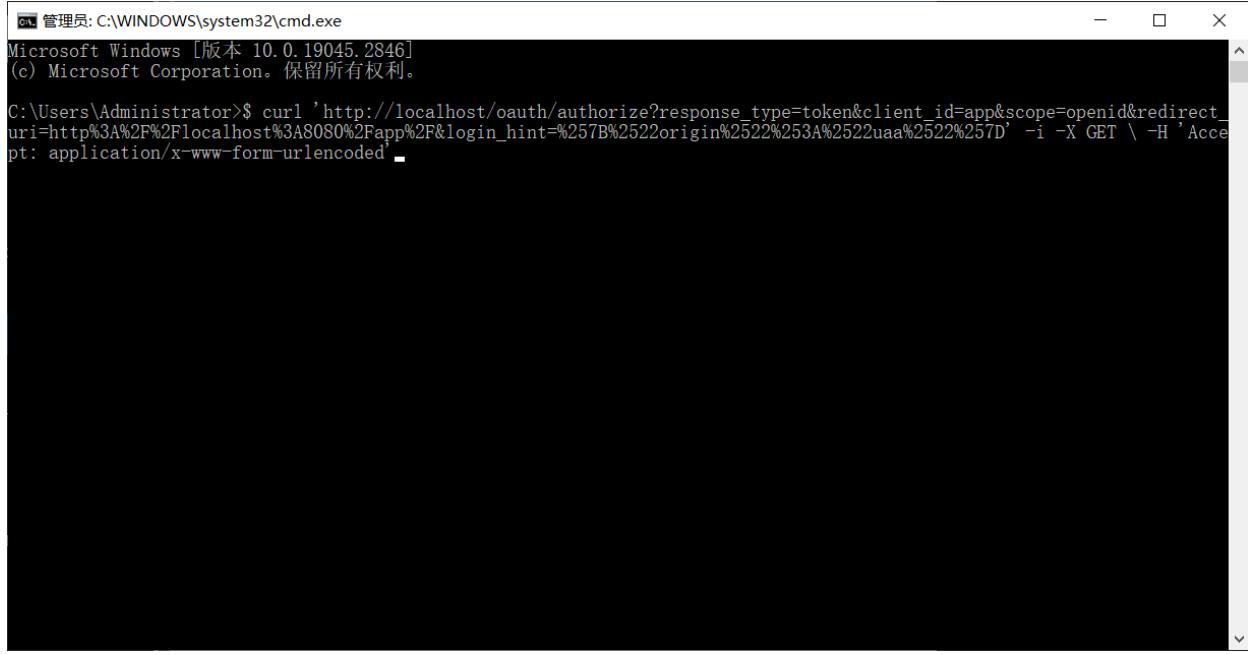
Now you need to set up a password for your user, which you can do by clicking manage your password.

Choose a password for your user and confirm it.

## Step3. Build CloudFoundry App

Start the CloudFoundry by follow.

- `$git clone git://github.com/cloudfoundry/uaa.git`
- `$ cd uaa`
- `./gradlew run`

A screenshot of a Windows Command Prompt window titled "管理员: C:\WINDOWS\system32\cmd.exe". The window shows the following command being run:  
C:\Users\Administrator>\$ curl 'http://localhost/oauth/authorize?response\_type=token&client\_id=app&scope=openid&redirect\_uri=http%3A%2F%2Flocalhost%3A8080%2Fapp%2F&login\_hint=%257B%2522origin%2522%253A%2522uaa%2522%257D' -i -X GET \ -H 'Accept: application/x-www-form-urlencoded'  
The command is intended to fetch an OAuth token from a local server using the Casdoor library.

## Step4. Integrate Casdoor

Now open another command line and input :

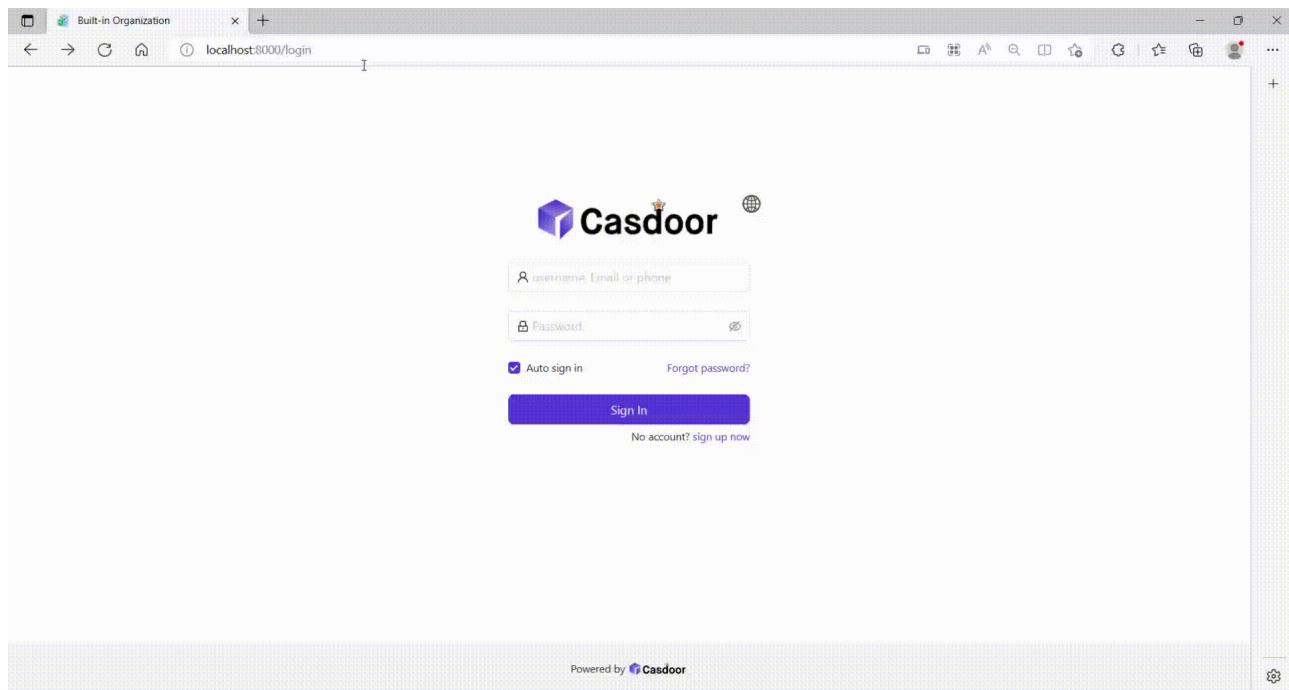
```
curl '<http://localhost/oauth/
authorize?response_type=token&client_id=app&scope=openid&redirect_uri=http%3A%2F%2Flocalhost%3A8080%2Fapp%2F>' 
-i -X GET \
-H 'Accept: application/x-www-form-urlencoded'
```

we have already got the client\_id and redirect\_uri before, we input these parameters.

Parameter	Type	Constraints	Description
response_type	String	Required	Space-delimited list of response types. Here, token , i.e. an access token
client_id	String	Required	a unique string representing the registration information provided by the client
scope	String	Optional	requested scopes, space-delimited
redirect_uri	String	Optional	redirection URI to which the authorization server will send the user-agent back once access is granted (or denied), optional if pre-registered by the client

execute the command, we can get the result below, which means that we have integrated Casdoor with CloudFoundry successfully.

```
HTTP/1.1 302 Found
Content-Security-Policy: script-src 'self'
Strict-Transport-Security: max-age=31536000
Set-Cookie: X-Uaa-CsrF=09mMqMDhcwHGLMufn84YA1; Path=/; Max-Age=86400; Expires=Fri, 5 May 2023 14:53:54 GMT; HttpOnly; SameSite=Lax
Cache-Control: no-store
Content-Language: en
X-XSS-Protection: 1; mode=block
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
Location: http://localhost:8080/app/#token_type=bearer&access_token=eyJhbGciOiJIUzI1NiIsImprdsI6Imh0dHBzO18vbG9jYWxob3N0OjgwODAvdWFhL3Rva
```



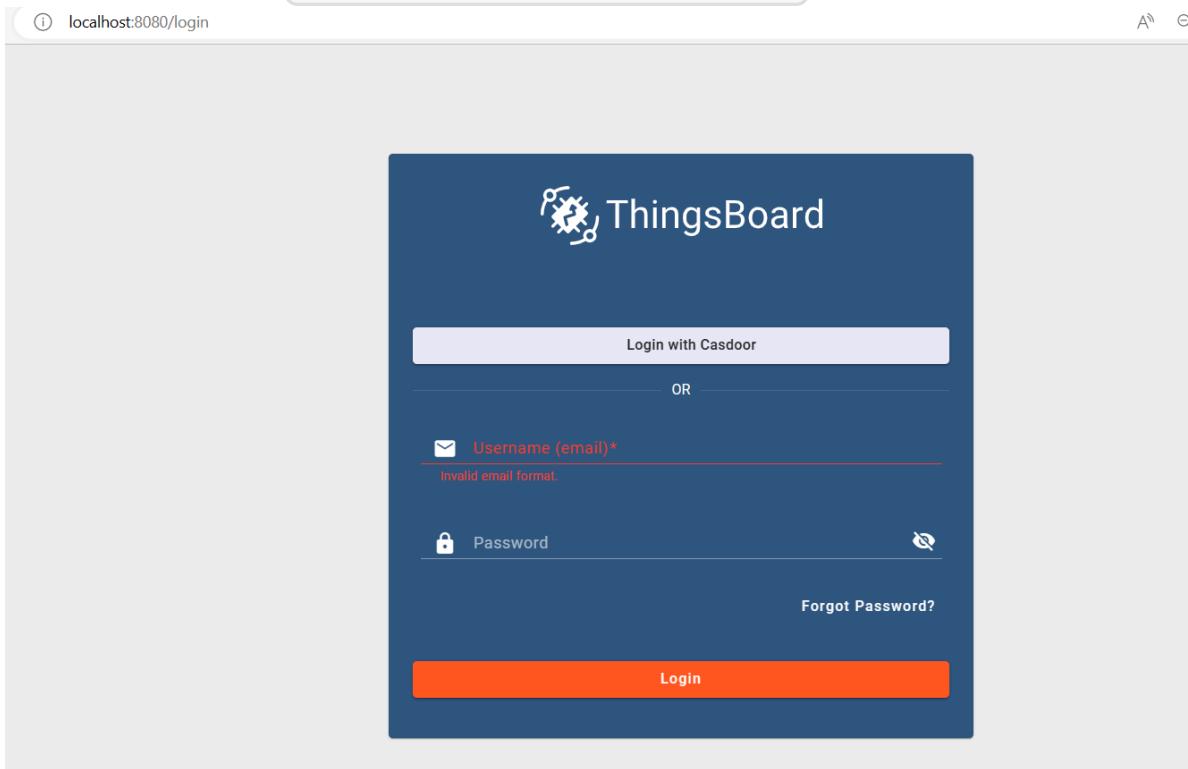
# Thingsboard

Before the integration, we need to deploy Casdoor locally.

Then we can quickly implement a Casdoor-based login page in our own app with the following steps.

## Step1. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url: `http://CASDOOR_HOSTNAME/login`



3. Copy the client ID and client secret, we will need it in the following steps.

## Step2. Add user in Casdoor

Now you have the application, but not a user. That means you need to create a user and assign the role.

Go to the "Users" page and click on "Add user" in the top right corner. That opens a new page where you can add the new user.

Save the user after adding a username and adding the organisation Thingsboard(other details are optional).

Now you need to set up a password for your user, which you can do by clicking manage your password.

Choose a password for your user and confirm it.

## Step3. Prerequisites and Build Thingsboard App

First of all, Thingsboard only support Java 11 (OpenJDK)

You can download in the following link:

[JDK Download Page](#)

Start the Thingsboard by follow.(Take Windows system as example)

- \$Download and extract the package. [Download the package](#)
- \$We can configure Thingsboard in \thingsboard\conf\thingsboard.yml as you like, configure the Kafka、PostgreSQL and others.

- \$Run “install.bat –loadDemo” in command line in Thingsboard folder to install and add demo data

```
C:\Program Files (x86)\thingsboard>install.bat --loadDemo  
Detecting Java version installed.  
CurrentVersion 110  
Java 11 found!  
Installing thingsboard ...  
...  
ThingsBoard installed successfully!
```

- \$Input "net start thingsboard" in command line, we will get  
The ThingsBoard Server Application service is starting.  
The ThingsBoard Server Application service was started successfully.
- 

## Step4. Integrate Casdoor

Now open <http://localhost:8080/> and login in admin account:

account: [sysadmin@thingsboard.org](mailto:sysadmin@thingsboard.org) / password: sysadmin

After login in successfully, we click the oath2 button on the left bottom in the page.

The screenshot shows the ThingsBoard Home dashboard. On the left, a sidebar menu includes: Home, Tenants, Tenant profiles, Resources (with a dropdown arrow), Notification center, Settings, Security (with a dropdown arrow), General, Two-factor authentication, and OAuth2.

The main dashboard area has a title "Home". It features several cards:

- Tenants**: Shows 2 tenants with a "+" button.
- Tenant profiles**: Shows 2 tenant profiles with a "+" button.
- CPU**: Shows 15% usage of 8 cores.
- Devices**: Shows 9 devices.
- Assets**: Shows 2 assets.
- Users**: Shows 8 users.
- Customers**: Shows 3 customers.
- Realtime - last h**: A chart showing CPU usage over time, ranging from 0% to 100%.
- Documentation**: Includes links to Getting started, Tenant profiles, API, and Widgets Library.
- Configured features**: Includes Email, SMS, Slack, OAuth 2, and 2FA.
- Transport messages**: Shows history of transport messages from May 02 to May 05.

Fill in the blank like this:

## Providers

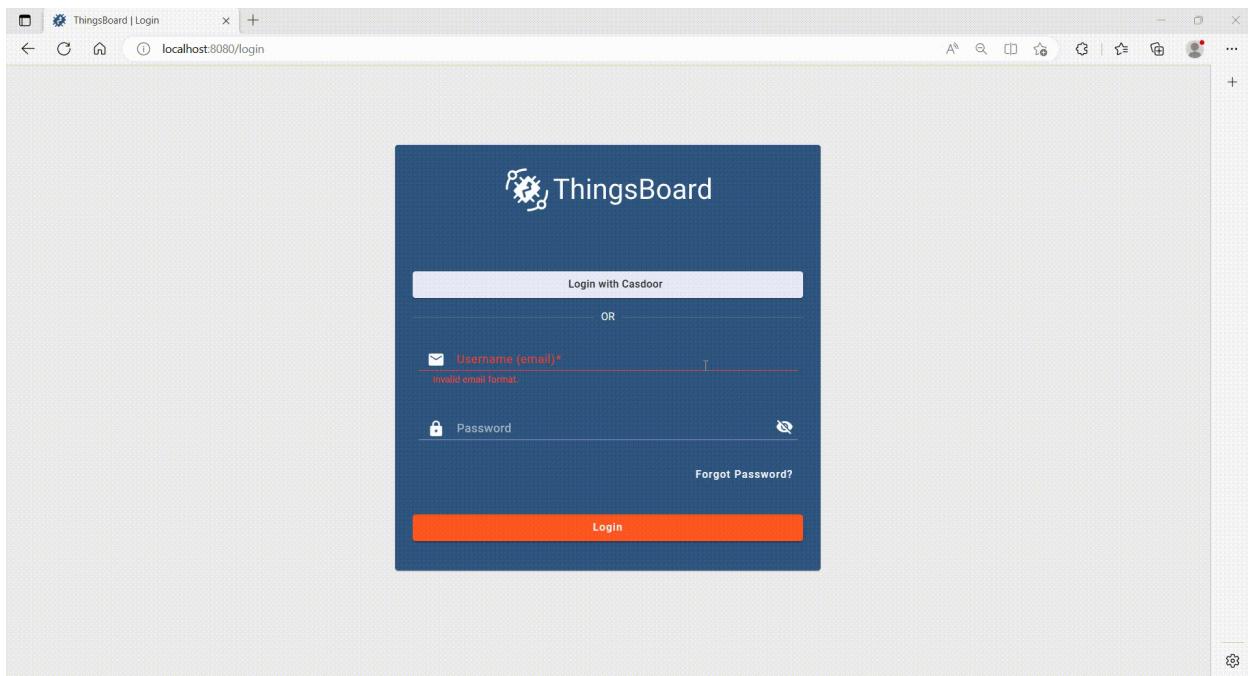
Custom

Login provider*	Custom	Allowed platforms	Web, Android, iOS
Client ID*	e324f9a3f55e1adac4ef	Client secret*	28b3f98c1f55c1cc57f74b9b1a68b5d2e79
General		Mapper	
Access token URI*	http://localhost:8000/api/login/oauth/access_token	Authorization URI*	http://localhost:8000/login/oauth/authorize
JSON Web Key URI	http://localhost:8000/.well-known/jwks	User info URI	http://localhost:8000/api/userinfo
Client authentication method*	POST		
Provider label*	Casdoor	Login button icon	
<input checked="" type="checkbox"/> Allow user creation			

We can get these values in this link: [OIDC discovery URL](#)

```
{  
  "issuer": "https://door.casdoor.com",  
  "authorization_endpoint": "https://door.casdoor.com/login/oauth/authorize",  
  "token_endpoint": "https://door.casdoor.com/api/login/oauth/access_token",  
  "userinfo_endpoint": "https://door.casdoor.com/api/userinfo",  
  "jwks_uri": "https://door.casdoor.com/.well-known/jwks",  
  "introspection_endpoint": "https://door.casdoor.com/api/login/oauth/introspect",  
  "response_types_supported": ["code"]}
```

After filling these blanks, we successfully integrate Casdoor with Thingsboard, when we login <http://localhost:8080/>, we can get this:



# JavaScript

## 微信小程序

在 微信小程序中使用 Casdoor

# 微信小程序

## ① 信息

Casdoor支持1.41.0版后的微信小程序

## 介绍

由于微信小程序不支持标准的 OAuth，所以它不能跳转到自主机的 Casdoor 网页进行登录。因此，将Casdoor 用于微信小程序的过程与普通方案的过程有所不同。

这份文件将讨论如何通过Casdoor访问微信小程序。您可以在 GitHub 上找到示例：[casdoor-wechat-miniprogram-example](#)。详细信息可参见微信小程序 [登录文档](#)。

以下是配置中的一些专有名词：

`CASDOOR_HOSTNAME`：私有部署的Casdoor域名或IP。例如：

`https://door.casbin.com.`

## 第一步：部署Casdoor

首先，您应先部署 [Casdoor](#)。

成功部署后，您需要确认：

1. Casdoor 可以正常登录使用。
2. 将 Casdoor 的 `origin` 值 (`conf/app.conf`) 设置为 `CASDOOR_HOSTNAME`。

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000"| CASDOOR_HOSTNAME
```

## 第二步： 配置Casdoor应用程序

1. 在 casdoor 创建一个wechat idp, 并填写微信小程序开发平台给您的 APPID 和 APPSECRET :

New Provider [Save](#) [Save & Exit](#) [Cancel](#)

Name [?](#) : provider\_Mini Program

Display name [?](#) : Mini Program

Category [?](#) : OAuth

Type [?](#) : WeChat Mini Program

Client ID [?](#) : \*\*\*

Client secret [?](#) : \*\*\*

Provider URL [?](#) : <https://github.com/organizations/xxx/settings/applications/1234567>

[Save](#) [Save & Exit](#) [Cancel](#)

2. 创建或使用现有的 Casdoor 应用程序。
3. 将上面添加的 idp 添加到您想要使用的应用程序中。

#### ❗ TIPS

为方便起见，casdoor 将在应用程序中读取第一个WeChat类型 idp 默认是微信小程序 idp 。

因此，如果你想要在这个应用中使用微信小程序，请不要在一个应用中添加 WeChat 类型 idp 。

# 第三步： 编写微信小程序

WeChat Mini 方案提供一个内部登录的API并获取代码。 您需要做的只是将此代码发送到Casdoor。 Casdoor 将使用此代码从 WeChat 服务器获取一些信息(例如OpenID、SessionKey 等)。

下面代码展示如何完成上述过程：

```
// login in mini program
wx.login({
  success: res => {
    // this is your login code you need to send to casdoor
    console.log(res.code)

    wx.request({
      url: `${CASDOOR_HOSTNAME}/api/login/oauth/access_token`,
      method: "POST",
      data: {
        "tag": "wechat_miniprogram", // required
        "client_id": "6825f4f0af45554c8952",
        "code": res.code,
        "username": this.data.userInfo.nickName, // update user
        profile, when you login.
        "avatar": this.data.userInfo.avatarUrl,
      },
      header: {
        "content-type": "application/x-www-form-urlencoded",
      },
      success: res => {
        console.log(res)
        this.globalData.accessToken = res.data.access_token // get
        casdoor's accessToken
      }
    })
  }
})
```

值得一提的是，`tag` 参数是强制性的，您需要让用户了解这是来自微信小程序的请求。

上述代码在用户登录时传入微信小程序用户名和头像uri。您也可以通过这两个参数而不首先传递它们。然后在登录成功并获得访问令牌后将它们传递到 casdoor：

```
WX.getUserProfile({
  desc: 'share your info to casdoor',
  success: (res) => {
    this.setData({
      userInfo: res.userInfo,
      hasUserInfo: true
    })
    console.log(app.globalData.accessToken)
    wx.request({
      url: `${CASDOOR_HOSTNAME}/api/update-user`, // casdoor uri
      method: "POST",
      data: {
        "owner": "test",
        "name": "wechat-oGk3T5tIiMFo3SazC075f0HEiE7Q",
        "displayName": this.data.userInfo.nickName,
        "avatar": this.data.userInfo.avatarUrl
      },
      header: {
        "Authorization": "Bearer " + app.globalData.accessToken,
        // Bearer token
        "content-type": "application/json"
      },
      success: (res) => {
        console.log(res)
      }
    })
  }
})
```

此外，您可以使用 `accessToken` 作为任何Casdoor操作的访问令牌

 TIPS

目前 Casdoor 无法将现有账户绑定到微信小程序用户。在 Catdoor 从WeChat 获取 openID , 如果此 id 不存在。将创建一个新用户, 如果它存在, 将使用旧用户。



> 集成 >

Lua

# Lua



APISIX

在 APISIX 中使用 Casdoor

# APISIX

目前有两种方法可以使用 Casdoor 通过 APISIX 插件连接到 APISIX，并保护 APISIX 背后的 API：使用 APISIX 的 Casdoor 插件或使用 APISIX 的 OIDC 插件。

## 通过 APISIX 的 Casdoor 插件连接 Casdoor

这个名为 authz-casdoor 的插件可以保护 APISIX 背后的 api，强制每个请求都经过身份验证，而无需修改 api 代码。

### 如何启用它？

您需要在创建路由时指定此插件，并给出所有必需的字段。参见下面的示例。

```
curl "http://127.0.0.1:9180/apisix/admin/routes/1" -H "X-API-KEY: edd1c9f034335f136f87ad84b625c8f1" -X PUT -d '  
{}  
  "methods": ["GET"],  
  "uri": "/anything/*",  
  "plugins": {  
    "authz-casdoor": {  
      "endpoint_addr": "http://localhost:8000",  
      "callback_url": "http://localhost:9080/anything/callback",  
      "client_id": "7ceb9b7fda4a9061ec1c",  
      "client_secret": "3416238e1edf915eac08b8fe345b2b95cdba7e04"  
    }  
  },  
  "upstream": {
```

在本例中，我们使用apisix的管理API创建了指向“`httpbin.org:80`”的路由“`/anything/*`”，并启用了“authz casdoor”。此路由现在受到 Casdoor 的身份验证保护。

## 属性

名称	类型	申请标准	默认	有效期	描述
<code>endpoint_addr</code>	字符串	必填			Casdoor的url。
<code>client_id</code>	字符串	必填			Casdoor 中的 client id。
<code>client_secret</code>	字符串	必填			Casdoor 中的 client secret。
<code>callback_url</code>	字符串	必填			用于接收状态和代码的回调 url。

`endpoint_addr` 和 `callback_url` 不应以“`/`”结尾

在“authz casdoor”插件的配置中，我们可以看到四个参数。

第一个是“`callback_url`”。这是OAuth2中的回调url。It should be emphasized that this callback url must belong to the "uri" you specified for the route, for example, in this example, <http://localhost:9080/anything/callback> obviously belongs to “`/anything/*`”。只有通过这种方式，插件才能拦截并利用对 `callback_url` 的访问（这样插件才能在 OAuth2 中获得代码和状态）。`Callback_url` 的逻辑完全由插件实现，因此无需修改服务器来实现此回调。

第二个参数“endpoint\_addr”显然是Casdoor的url。第三个和第四个参数是“client\_id”和“client\_secret”，您可以在注册应用程序时从Casdoor获取这些参数。

## 它是如何工作的？

Suppose a new user who has never visited this route before is going to visit it (<http://localhost:9080/anything/d?param1=foo&param2=bar>), considering that "authz-casdoor" is enabled, this visit would be processed by "authz-casdoor" plugin first. 然后检查会话，若此用户尚未通过身份验证后，将拦截访问。用户想要继续访问原始url后，他将被重定向到Casdoor的登录页面。

成功登录用户名和密码(或他使用的任何方法)，Casdoor 会将此用户重定向到“callback\_url”，带有GET 参数“code”和“state”。因为插件知道“callback\_url”，当这次拦截访问“callback\_url”时，Oauth2中的“授权代码流”逻辑将被触发，这意味着此插件将请求访问令牌以确认此用户是否真的登录。确认后，此插件将会将此用户重定向到原始用户想访问的URL，这是我们先前保存的。登录状态也将保持在会话。

Next time this user want to visit url behind this route (for example, <http://localhost:9080/anything/d>), after discovering that this user has been authenticated previously, this plugin won't redirect this user anymore so that this user can visit whatever he wants under this route without being interfered.

## 通过 APISIX 的 OIDC 插件连接 Casdoor

Casdoor 可以使用 OIDC 协议链接到 APISIX，这份文档将向您展示如何处理相关问题。

以下是配置中的一些专有名词：

`CASDOOR_HOSTNAME`: 私有部署的Casdoor域名或IP。

`APISIX_HOSTNAME`: 部署 APISIX 的域名或 IP。

## 步骤1. 部署Casdoor和APISIX

Firstly, the Casdoor and APISIX should be deployed.

在成功部署后，您需要确保：

1. 可以登录并正常使用Casdoor。
2. 将Casdoor的 origin 值 (conf/app.conf) 设置为 CASDOOR\_HOSTNAME。

```
conf > ⚙ app.conf
  8  dbName = casdoor
  9  redisEndpoint =
10  defaultStorageProvider =
11  isCloudIntranet = false
12  authState = "casdoor"
13  httpProxy = "127.0.0.1:10808"
14  verificationCodeTimeout = 10
15  initScore = 2000
16  logPostOnly = true
17  origin = "http://10.144.1.2:8000"|
                                CASDOOR_HOSTNAME
```

## 步骤2. 配置Casdoor应用程序

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加一个重定向网址： http://APISIX\_HOSTNAME/REDIRECTWHATYOUWANT 并更改 REDIRECTWHATYOUWANT 到你需要的重定向网址上。
3. 选择 "JWT-Empty" 作为令牌格式选项
4. 添加您想要的提供商并补充其他设置。

The screenshot shows the Casdoor configuration interface for a new client. It includes fields for Client ID (07860a229bd0b162cdfa), Client secret (ea021...9373fe3e), Redirect URLs (localhost:9000/callback), and Token format (JWT-Empty). A 'Select JWT-Empty' button is also present.

不出意外的话，您会在应用程序设置页面看到： `Client ID` 和 `Client secret` 就像上面的图片一样。 我们将在下一步中使用它们。

打开你喜欢的浏览器，访问：`http://CASDOOR_HOSTNAME/.well-known/openid-configuration`，你会看到 Casdoor 的 OIDC 配置。

## 步骤3. 配置 APISIX

APISIX 拥有官方 OIDC 支持，由 `lua-resety-openidc` 实现。

您可以根据 APISX OIDC 文档定制设置，使用以下路由设置：

```
#Use your own X-Api-Key
$ curl -XPOST APISIX_HOSTNAME/apisix/admin/routes -H "X-Api-Key: edd1c9f034335f136f87ad84b625c8f1" -d '{
  "uri": "/get",
  "name": "apisix_casdoor_test",
  "plugins": {
    "openid-connect": {
      "client_id": "Client ID",
      "client_secret": "Client secret",
      "discovery": "http://CASDOOR_HOSTNAME/.well-known/openid-configuration",
      "introspection_endpoint_auth_method": "client_secret_basic",
      "logout_path": "/logout",
      "realm": "master",
      "redirect_uri": "http://APISIX_HOSTNAME/REDIRECTWHATYOUWANT",
      "bearer_only": false,
    }
  }
}'
```

现在, 请访问 `http://APISIX_HOSTNAME/get`, 浏览器会将您重定向到Casdoor登录页面, 并且在登录成功后, 您会看到我们已经向 `httpbin.org` 发出了请求。

# PHP

## 禅道

在禅道中使用 Casdoor 进行身份验证

## ShowDoc

在 ShowDoc 中使用 Casdoor 为 oAuth2 服务器

## Flarum

Using OAuth2 to connect various applications, like Flarum

## Moodle

Using Oauth to connect Moodle

# 禅道

禅道 是一个 敏捷(scrum) 项目管理系统/工具，但它不支持 OIDC 本身。For integrating Zentao with Casdoor SSO, we should via 3rd-party OIDC module [zentao-oidc](#), and this document will show you how to do it.

## 步骤1. 部署Casdoor和禅道

Firstly, the [Casdoor](#) and [Zentao](#) should be deployed. 在成功部署后，您需要确保：

1. Casdoor 可以正常登录使用。
2. 您可以成功登录并使用禅道

## 步骤2. 集成禅道 OIDC 第三方模块

安装 [zentao-oidc](#)

```
git clone https://github.com/casdoor/zentao-oidc.git
```

或者您可以下载 ZIP 并解压缩它。

此模块用于在 OpenID 与 SSO 集成时使用 禅道 用法如下：

1. 将整个oidc目录复制到禅道模块中，并使用它作为禅道模块。重命名下载的软件包为“oidc”
2. 配置过滤器

Because the framework of Zentao filters the parameters in URL and does not allow Spaces. So you need to put the following code at the end of `/config/my.php`.

```
$filter->oidc=new stdclass();
$filter->oidc->index=new stdclass();
$filter->oidc->index->paramValue['scope']='reg::any';
```

### 3. Modify `/module/common/model.php`

Put 'oidc' on the anonymous access list and add a line to the `isOpenMethod` method of `model.php`.

```
public function isOpenMethod($module, $method){
    if($module == 'oidc' and $method == 'index') return true;
}
```

### 4. If you do not want the Zentao login screen to appear, go directly to the Casdoor login screen.

Modify the last line of code at `public function checkPriv()` in `/module/common/model.php`.

```
//return print(js::locate(helper::createLink('user', 'login',
"referer=$referer")));
return print(js::locate(helper::createLink('oidc', 'index',
"referer=$referer")));
```

### 5. Modify `setSuperVars()` method inside of `framework/base/router.class.php`, comment out the following statements.

```
public function setSuperVars()  
// unset($_REQUEST);
```

## 步骤3. 配置Casdoor应用程序

1. Create or use an existing Casdoor application.
2. Add Your redirect url

The screenshot shows the Casdoor configuration interface. It includes fields for Client ID (d8d7715e24f077066a20), Client secret (redacted), Cert (cert-built-in), and Redirect URLs (http://127.0.0.1/zentao/oidc-index.html). A 'Redirect URLs' button is also visible.

3. Add provider you want and supplement other settings.

## 步骤4. 配置 Jenkins

在 oidc 中配置 config.php

```
$config->oidc->clientId=<Your ClientId>;  
$config->oidc->clientSecret=<Your ClientSecret>;  
$config->oidc->issuer="http://localhost:8000";
```

在 module/oidc 公共函数索引() 中设置你的重定向Url

```
$oidc->setRedirectURL($path."/zentao/oidc-index.html");
```

① 备注

这里的URL是指调用 'index' 方法在 'oidc' 模块中。 You also need to set a variable separator, which the framework defaults to with a dash : - please refer to zentao's official framework for details. "[禅道框架](#)"

# ShowDoc

## 在 ShowDoc 中使用 Casdoor 进行身份验证

ShowDoc is an online API documentation, technical documentation tool perfect for IT teams. Showdoc 可以轻松使用 Markdown 语法来写美丽的 API 文档、数据字典文档、技术文档、在线Excel文档等等。

Showdoc 支持第三方身份验证，包括Oauth。 以下是操作教程。

### 第1步： 创建一个Casdoor应用程序

转到您的 Casdoor 并添加您的新应用程序 ShowDoc。 下面是在Casdoor中创建 ShowDoc应用程序的示例。

[Edit Application](#)[Save](#)[Save & Exit](#)Name [?](#) :

myApplication

Display name [?](#) :

myApplication

Logo [?](#) :URL [?](#) :[https://cdn.casdoor.com/logo/casdoor-logo\\_1185x256.png](https://cdn.casdoor.com/logo/casdoor-logo_1185x256.png)

Preview:

Home [?](#) :[?](#)Description [?](#) :Organization [?](#) :

built-in

Client ID [?](#) :

208d745196c23df9fd5b

Client secret [?](#) :

4c89f447af77bc276431ab885463ebcb8d6efc3c

Cert [?](#) :

cert-built-in

请记住下一步的 `client ID` 和 `client Secret`。

...信息

请在此步骤中不要填写 `callback url`。Url取决于下一步 `showdoc` 的配置。稍后我们将返回来设置一个正确的回调URL。

...

## 第2步：配置ShowDoc

首先，启动 OAuth2 登录按钮。然后按照示例填写 `callback url` 填写上一步中记住的 `client ID` 和 `client secret`。

The screenshot shows the ShowDoc configuration interface. On the left, there is a sidebar with the following menu items:

- 用户管理
- 项目管理
- 附件管理
- 集成登录** (highlighted with a red box)
- 站点设置
- 关于本站

The main configuration area has the following tabs at the top: `LDAP`, `OAuth2` (highlighted with a red box), and `通用接入`. Below the tabs, there is a toggle switch labeled `启动OAuth2登录` (highlighted with a red box). The configuration fields include:

- `callback url`: `http://127.0.0.1/server/?s=/api/extLogin/oauth2`
- `入口文字提示`: `casdoor sso`
- `Client id`: `208d745196c23df9fd5b`
- `Client secret`: `4c89f447af77bc276431ab885463ebcb8d6efc3c`
- `Oauth host`: `http://` dropdown set to `127.0.0.1:8000`
- `Authorize path`: `/login/oauth/authorize`
- `AccessToken path`: `/api/login/oauth/access_token`

需要 `Authorize path`、`AccessToken path`、`User info path`。您可以填写如下所示。

```
Authorize path: /login/oauth/authorize
AccessToken path: /api/login/oauth/access_token
User info path: /api/get-account
```

## 第3步：在casdoor中配置回调url

返回步骤1中的应用程序编辑页面并添加您填写在ShowDoc中的 `callback url`。

Redirect URLs ② :

Redirect URLs	Add
Redirect URL	<input type="text" value="http://127.0.0.1/server/?s=/api/extLogin/oauth2"/>

## 第4步：在ShowDoc上试试

您应该在登录页面中看到这一点：

# 登录

 用户名/邮箱 密码 验证码[注册新账号](#)[casdoor sso](#)

恭喜您！ 您已完成所有步骤 按 "castor sso" 按钮，您将被重定向到casdoor登录页面。

# Flarum

Casdoor can use OAuth2 to connect various applications. Here we will use Flarum as an example to show you how to use OAuth2 to connect to your applications.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`Flarum_HOSTNAME`: Domain name or IP where Flarum is deployed.

## Step1. Deploy Casdoor and Flarum

Firstly, the [Casdoor](#) and [Flarum](#) should be deployed.

After a successful deployment, you need to ensure:

1. Download the Flarum plugin [FoF Passport](#)
2. Casdoor can be logged in and used normally.
3. You can set `CASDOOR_HOSTNAME = http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find a redirect url: `<CASDOOR_HOSTNAME>/auth/passport`
3. Add your redirect url to casdoor application:

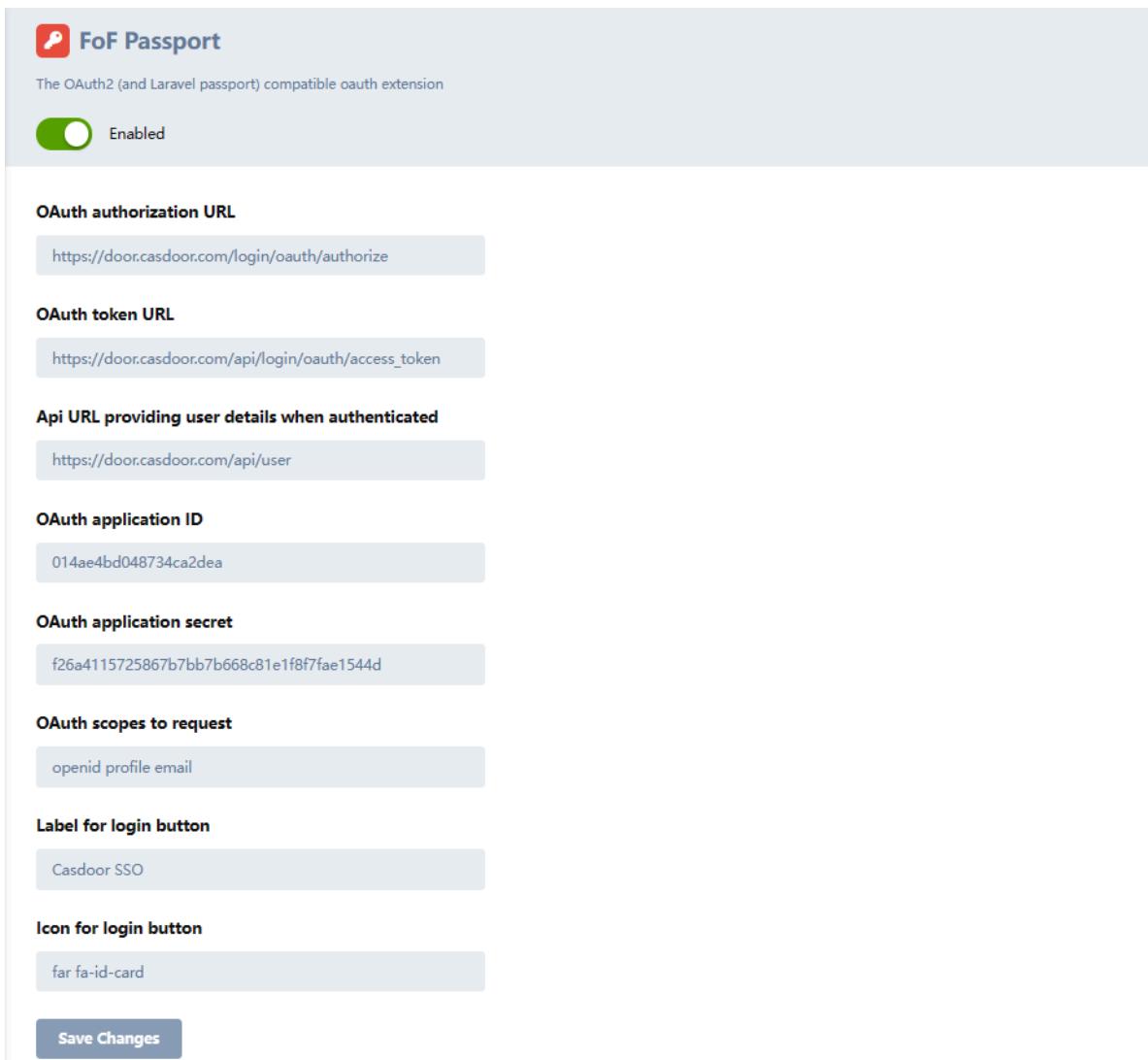
The screenshot shows the 'Application Settings' section of the Casdoor web interface. It includes fields for 'Client ID' (014ae4bd048734ca2dea), 'Client secret' (f26a4115725867b7bb7b668c81e1f8ffae1544d), 'Cert' (cert-built-in), and a 'Redirect URLs' section containing a single entry: <your flarum install>/auth/passport.

Not surprisingly, you can get two values on the application settings page: `Client ID` and `Client secret` like the picture above, we will use them in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`, you will see the OIDC configure of Casdoor.

## Step3. Configure Flarum

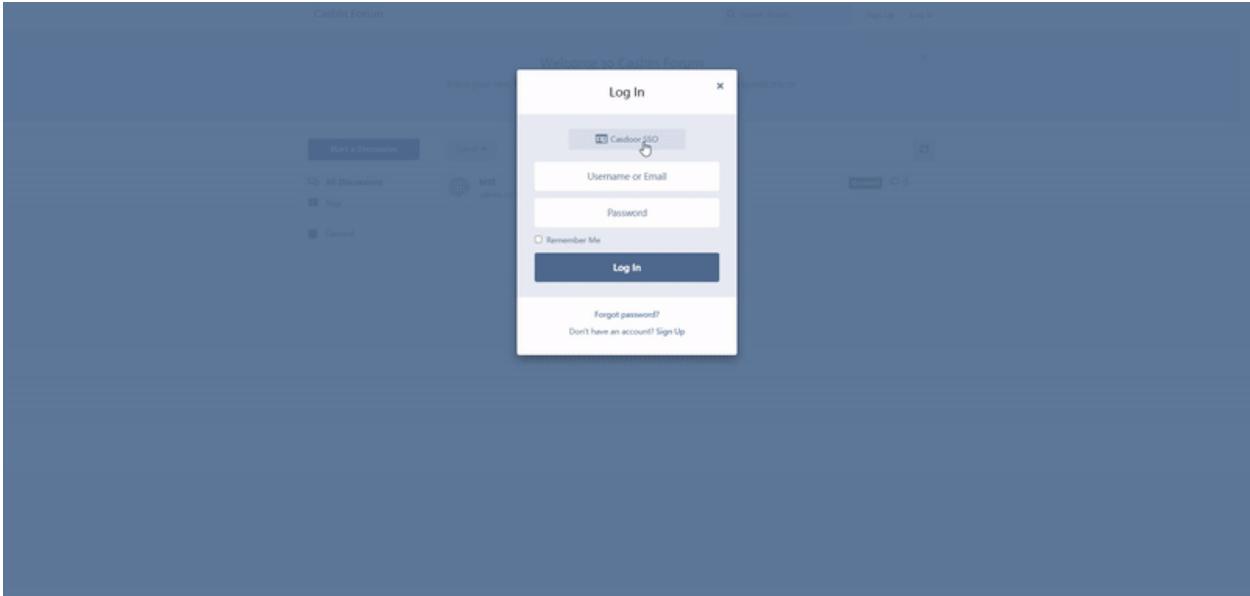
1. You should install a plugin [FoF Passport](#)
2. You should config this app



3. You can find Client Id and Client Secret in Casdoor application page.

- Token server url: http://**CASDOOR\_HOSTNAME**/api/login/oauth/access\_token
- Authorization server url: http://**CASDOOR\_HOSTNAME**/login/oauth/authorize
- UserInfo server url: http://**CASDOOR\_HOSTNAME**/api/get-account
- Scopes: address phone openid profile offline\_access email

Log out of Flarum, and test SSO.



# Moodle

Casdoor can use Oauth to connect Moodle.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`Moodle_HOSTNAME`: Domain name or IP where Moodle is deployed.

## Step1. Deploy Casdoor and Moodle

Firstly, the Casdoor and Moodle should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME = http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find a redirect url: `Moddle_HOSTNAME /admin/oauth2callback.php`
3. Add your redirect url to casdoor application

More infomation for [OAuth](#)

# Step3. Configure Moodle

## 1. You should find OAuth

The screenshot shows the Moodle Site administration interface. The 'Server' tab is active. A red arrow points to the 'OAuth 2 services' link in the list of options.

## 2. You should config this app

The screenshot shows the 'Detailed instructions on configuring the common OAuth 2 services' page. Red arrows point to specific fields: 'Client ID' (154fb67917b18c0a1850), 'Client secret' (380a93b571ab0f8545fbc), 'Service base URL' (https://demo.casdoor.com), and 'Logo URL' (https://cdn.casdoor.com/s).

## 3. You should config this Mappling

## User field mappings for issuer: Casdoor

External field name	Internal field name	Edit
address	address	
email	email	
name	firstname	
phone	phone1	
picture	picture	
preferred_username	username	

Create new user field mapping for issuer "Casdoor"

## 4. Find OAuth2 plugin

The screenshot shows the Moodle Plugins page. At the top, there is a navigation bar with tabs: General, Users, Courses, Grades, Plugins, Appearance, Server, Reports, and Development. The Plugins tab is highlighted with a blue underline. Below the navigation bar, a pink banner displays the message "Your site is not yet registered." with a "Register your site" button. The main content area is divided into several sections:

- Plugins**: A link to "Install plugins" and "Plugins overview".
- Activity modules**: A large list of activity types including: Manage activities, Common activity settings, Assignment, Assignment settings, Submission plugins, Manage assignment submission plugins, File submissions, Online text submissions, Feedback plugins, Manage assignment feedback plugins, Feedback comments, Annotate PDF, File feedback, Offline grading worksheet, Book, Chat, Database, External tool, Manage tools, Feedback, File, Folder, Forum, Glossary, HSP, IMS content package, Lesson, Page, Quiz, General settings, Safe Exam Browser templates, Safe Exam Browser access rules, SCORM package, Text and media area, URL, and Workshop.
- Admin tools**: A list including: Manage admin tools, Accessibility, Brickfield registration, Accessibility toolkit settings, Reports, and Recycle bin.
- Antivirus plugins**: A link to "Manage antivirus plugins".
- Authentication**: A list including: Manage authentication, Email-based self-registration, Manual accounts, and OAuth 2. A red arrow points to the "OAuth 2" item.

## 5. Enable OAuth2 plugin

## Manage authentication

### Available authentication plugins

Name	Users	Enable	Up/Down	Settings	Test settings	Uninstall
Manual accounts	2			<a href="#">Settings</a>		
No login	0					
Email-based self-registration	0			<a href="#">Settings</a>		<a href="#">Uninstall</a>
OAuth 2	8			<a href="#">Settings</a>	<a href="#">Test settings</a>	

## 6. if you want Casdoor's email can't be edited

### Lock user fields

You can lock user data fields. This is useful for sites where the user data is maintained by the administrators manually by editing user records or uploading using the 'Upload users' facility. If you are locking fields that are required by Moodle, make sure that you provide that data when creating user accounts or the accounts will be unusable.

Consider setting the lock mode to 'Unlocked if empty' to avoid this problem.

Lock value (First name) auth_oauth2   field_lock_firstname	Unlocked Default: Unlocked
Lock value (Last name) auth_oauth2   field_lock_lastname	Unlocked Default: Unlocked
Lock value (Email address) auth_oauth2   field_lock_email	Locked Default: Unlocked
Lock value (City/town) auth_oauth2   field_lock_city	Unlocked Default: Unlocked
Lock value (Country) auth_oauth2   field_lock_country	Unlocked Default: Unlocked
Lock value (Language) auth_oauth2   field_lock_lang	Unlocked Default: Unlocked

here is switch to lock email

More infomation for [Moodle](#) and Fields mapping

Log out of Moodle, and test SSO.

The screenshot shows the Moodle login page with the title "MyMoodle Website". At the top right, there is a "Log in" button. The rest of the page is blank, indicating a successful log-out.



&gt; 集成

&gt; Ruby

# Ruby

## GitLab

在 GitLab 服务器中使用 Casdoor 进行身份验证

# GitLab

Casdoor 可以使用 OIDC 协议链接到私有部署的 GitLab，该文档将向您展示如何处理相关问题。

## ⚠ 注意事项

As [GitLab docs](#) said, GitLab only works with OpenID providers that use HTTPS, so you need to deploy Casdoor with HTTPS, like putting Casdoor behind a NGINX reverse proxy with SSL certificate setup. Casdoor itself only listens to 8000 port by default via HTTP and has no HTTPS related functionality.

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed. e.g.,  
`https://door.casbin.com`.

`GITLAB_HOSTNAME`: Domain name or IP where GitLab is deployed. e.g.,  
`https://gitlab.com`.

## 第 1 步：部署 Casdoor 和 GitLab

Firstly, the [Casdoor](#) and [GitLab](#) should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor 可以正常登录使用。
2. 将 Casdoor 的 `origin` 值 (`conf/app.conf`) 设置为 `CASDOOR_HOSTNAME`。

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000"| CASDOOR_HOSTNAME
```

## 第 2 步：配置 Casdoor 应用程序

1. 创建或使用现有的 Casdoor 应用程序。
2. 添加重定向网址：[http://GITLAB\\_HOSTNAME/users/auth/openid\\_connect/callback](http://GITLAB_HOSTNAME/users/auth/openid_connect/callback)。
3. 添加您想要的提供商并补充其他设置。

Description <a href="#">?</a> :	GitLab	
Organization <a href="#">?</a> :	built-in	
Client ID <a href="#">?</a> :	eab9...35b6	Client ID
Client secret <a href="#">?</a> :	95e7...b3a0188a5	Client secret
Redirect URLs <a href="#">?</a> :	<a href="#">Add</a>	
	Redirect URL	
	<a href="http://GITLAB_HOSTNAME/users/auth/openid_connect/callback">http://GITLAB_HOSTNAME/users/auth/openid_connect/callback</a>	GitLab redirect url

Not surprisingly, you can get two values on the application settings page: [Client ID](#) and [Client secret](#) like the picture above, and we will use them in the next

step.

Open your favorite browser and visit: [http://\*\*CASDOOR\\_HOSTNAME\*\*.well-known/openid-configuration](http://<b>CASDOOR_HOSTNAME</b>.well-known/openid-configuration), you will see the OIDC configure of Casdoor.

## 第 3 步： 配置 GitLab

You can follow the steps below to set this up, or make custom changes according to [this document](#)(e.g., you are installing GitLab using source code rather than Omnibus).

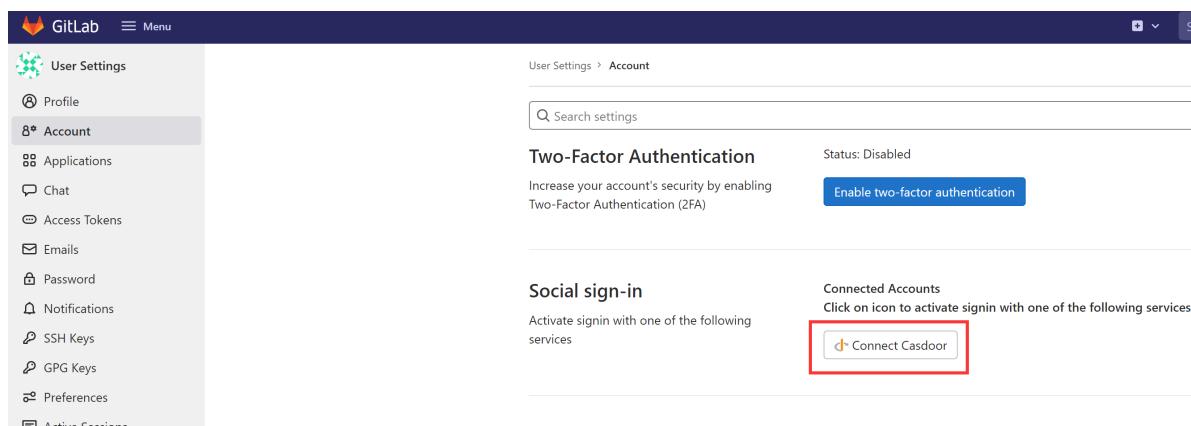
1. 在 GitLab 服务器上，打开配置文件。

```
sudo editor /etc/gitlab/gitlab.rb
```

2. 添加提供商配置。 (HOSTNAME 链接应包括 http 或 https)

```
gitlab_rails['omniauth_providers'] = [
  {
    name: "openid_connect",
    label: "Casdoor", # 登录按钮的可选标签, 默认为"Openid Connect"
    args: {
      name: "openid_connect",
      scope: ["openid", "profile", "email"],
      response_type: "code",
      issuer: "<CASDOOR_HOSTNAME>",
      client_auth_method: "query",
      discovery: true,
      uid_field: "preferred_username",
      client_options: {
        identifier: "<你的 CLIENT ID>",
        secret: "<你的 CLIENT SECRET>",
      }
    }
  }
]
```

3. 重新启动 GitLab 服务器。
4. 每个注册用户都可以打开 `GITLAB_HOSTNAME/-/profile/account`, 连接 Casdoor 账号。



The screenshot shows the 'User Settings > Account' page in GitLab. On the left, there's a sidebar with options like Profile, Account (which is selected), Applications, Chat, Access Tokens, Emails, Password, Notifications, SSH Keys, GPG Keys, and Preferences. The main content area has a 'Two-Factor Authentication' section with a status of 'Disabled' and a 'Enable two-factor authentication' button. Below it is a 'Social sign-in' section with a sub-section for 'Connected Accounts'. A red box highlights the 'Connect Casdoor' button next to the Casdoor icon.

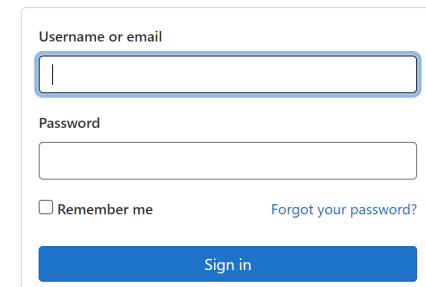
5. 完成！现在，您可以通过 casdoor 登录您自己的 GitLab。

## GitLab

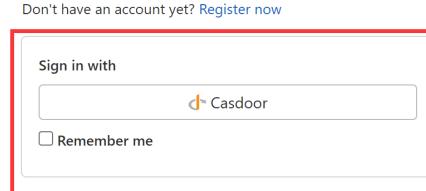
### A complete DevOps platform

GitLab is a single application for the entire software development lifecycle. From project planning and source code management to CI/CD, monitoring, and security.

This is a self-managed instance of GitLab.



The screenshot shows the GitLab login page. It features fields for 'Username or email' and 'Password', and checkboxes for 'Remember me' and 'Forgot your password?'. Below these is a large blue 'Sign in' button. At the bottom, there's a link 'Don't have an account yet? Register now'. A red box highlights the 'Sign in with' section, which includes a 'Casdoor' button and a 'Remember me' checkbox.



The screenshot shows the 'Sign in with' section of the GitLab login page. It includes a 'Casdoor' button and a 'Remember me' checkbox. This section is also highlighted with a red box.



> 集成 >

Haskell

# Haskell

## Hasura

Before the integration, we need to deploy Casdoor locally.

# Hasura

Before the integration, we need to deploy Casdoor locally.

Then we can quickly implement a Casdoor-based login page in our own app with the following steps.

## Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect url: `http://CASDOOR_HOSTNAME/login`



3. Copy the client ID, we will need it in the following steps.

## Add user in Casdoor

Now you have the application, but not a user. That means you need to create a user and assign the role.

Go to the “Users” page and click on “Add user” in the top right corner. That opens a new page where you can add the new user.

A screenshot of the Casdoor Users management page. The page title is 'localhost:8000/users'. The navigation bar includes 'Home', 'Organizations', 'Users' (which is selected and highlighted in blue), 'Roles', 'Permissions', 'Models', 'Adapters', 'Applications', 'Providers', 'Resources', 'Records', 'Tokens', 'Sessions', and 'Admin'. Below the navigation is a search bar with filters for 'Organization', 'Application', 'Name', 'Created time', 'Display name', 'Avatar', 'Email', 'Phone', 'Affiliation', 'Country/Region', 'Tag', and 'Action'. The main content area is a table with the following data:

Organization	Application	Name	Created time	Display name	Avatar	Email	Phone	Affiliation	Country/Region	Tag	Action
built-in	hasur	test1	2023-04-12 12:25:32	test1		0pgox8@example.com	24791921817	Example Inc.			<button>Edit</button> <button>Delete</button>
hasura	app-built-in	user_ij6lmm	2023-04-09 15:45:37	New User - ij6lmm		ij6lmm@example.com	83922294227	Example Inc.			<button>Edit</button> <button>Delete</button>
hasura	hasura	test	2023-04-09 15:29:56	test		91ized@example.com	19309373096	Example Inc.			<button>Edit</button> <button>Delete</button>
built-in	app-built-in	test	2023-04-09 14:57:23	New User - 3asyap		3asyap@example.com	24906027164	Example Inc.			<button>Edit</button> <button>Delete</button>
built-in	app-built-in	admin	2023-03-29 02:01:40	Admin		admin@example.com	12345678910	Example Inc.			<button>Edit</button> <button>Delete</button>

At the bottom right, there is a pagination control showing '5 in total' and a page number '1'.

Save the user after adding a username and adding the organisation Hasura(other details are optional).

Now you need to set up a password for your user, which you can do by clicking manage your password.

Choose a password for your user and confirm it.

# Build Hasura App

Start the Hasura by docker or Hasura Cloud.

Now create a `users` table with the following columns:

- `id` of type Text (Primary Key)
- `username` of type Text

See the image below for reference.

The screenshot shows the Hasura Data Manager interface. On the left, there's a sidebar with 'Data Manager' and 'Databases (1)'. It lists two databases: 'default' and 'public'. Under 'default', it says 'No tables or views in this schema'. Below that is a 'SQL' section. The main area is titled 'Add a New Table' and has a form for creating a 'users' table. The 'Table Name' field contains 'users'. The 'Table Comment' field contains 'comment'. In the 'CONFIGURE FIELDS' section, there are three columns defined: 'id' (Text, nullable, unique), 'username' (Text, nullable, unique), and 'column\_name' (column\_type, nullable, unique). There's also a button '+ Frequently used columns'. In the 'TABLE PROPERTIES' section, the 'Primary Key' is set to 'id'. A 'Foreign Keys' section is at the bottom. A yellow hand icon is in the bottom right corner.

The next step is to create a `user` role for the app. Users should be able to see only their records, but not the other people's records.

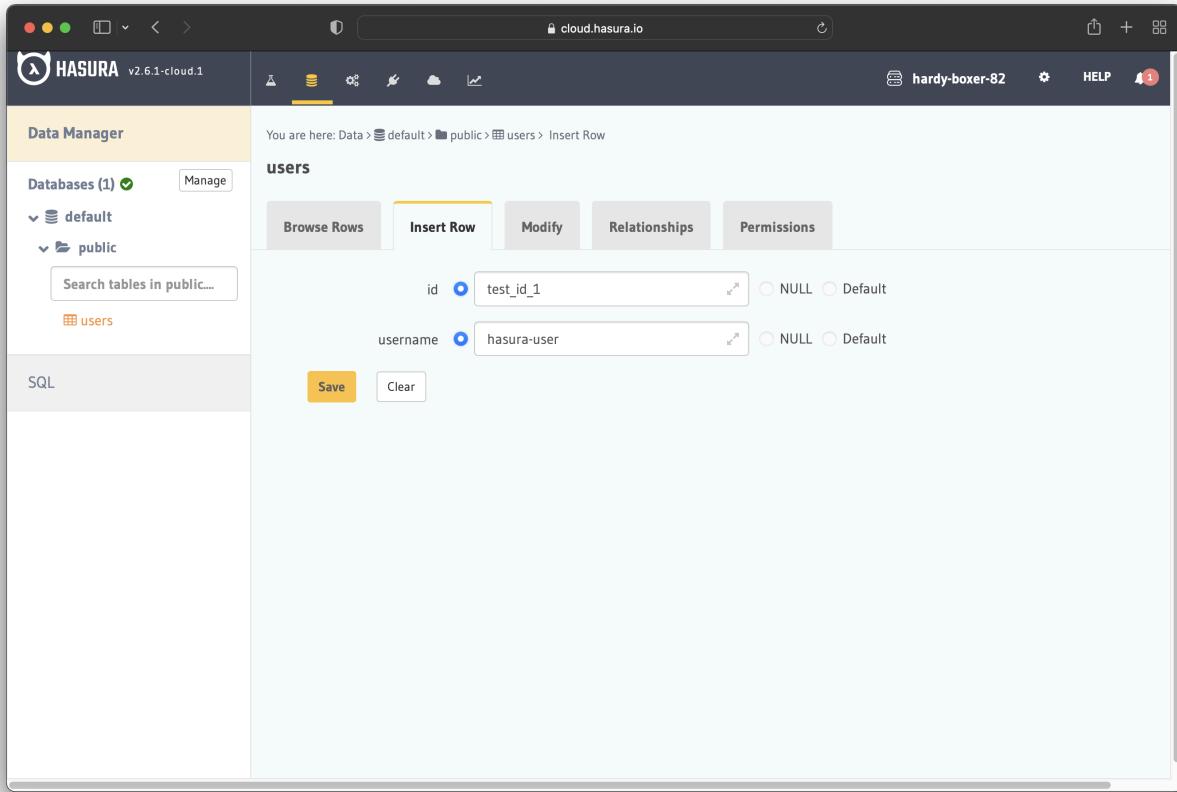
Configure the `user` role as shown in the image below. For more information, read about [configuring permission rules in Hasura](#).

The screenshot shows the Hasura Cloud interface with the following details:

- Role Permissions Table:** A grid showing permissions for 'admin' and 'user' roles across 'insert', 'select', 'update', and 'delete' actions.
- Row select permissions (With custom check):** A detailed view for the 'user' role's 'select' action. It shows a custom check condition: `{"id": {"_eq": "X-Hasura-User-Id"}}`. This condition filters rows where the 'id' column equals the header value 'X-Hasura-User-Id'.
- Column select permissions:** Shows that the 'user' role can access both 'id' and 'username' columns.

This way, users cannot read other people's records. They can only access theirs.

For testing purposes, add a dummy user. This is to ensure that when you use the JWT token, you only see your user's details and not other users' details.



Now you need to set the `JWT_SECRET` in Hasura.

## Configure Hasura with Casdoor

In this step, you need to add the `HASURA_GRAPHQL_JWT_SECRET` to Hasura.

To do so, go to the Hasura docker-compose.yaml and then add the new `HASURA_GRAPHQL_JWT_SECRET` as below.

The `HASURA_GRAPHQL_JWT_SECRET` should be in the following format:

```
HASURA_GRAPHQL_JWT_SECRET: '{"claims_map": {  
    "x-hasura-allowed-roles": ["user", "editor"],  
    "x-hasura-default-role": "user",  
    "x-hasura-user-id": "userID"  
}, "jwk_url": "https://door.casdoor.com/.well-known/jwks"}'
```

Save the change, and reload the docker.

```
## enable debugging mode. It is recommended to disable this in production
HASURA_GRAPHQL_DEV_MODE: "true"
HASURA_GRAPHQL_ENABLED_LOG_TYPES: startup, http-log, webhook-log, websocket-log, query-log
HASURA_GRAPHQL_ADMIN_SECRET: myadminsecretkey
HASURA_GRAPHQL_JWT_SECRET: '{"claims_map": {
  "x-hasura-allowed-roles": ["user", "editor"],
  "x-hasura-default-role": "user",
  "x-hasura-user-id": "4ec7ccee-ec7b-4191-a78d-e11f50686f8b"
}, "jwk_url": "https://door.casdoor.com/.well-known/jwks"}
```

## Retrieve JWT Token

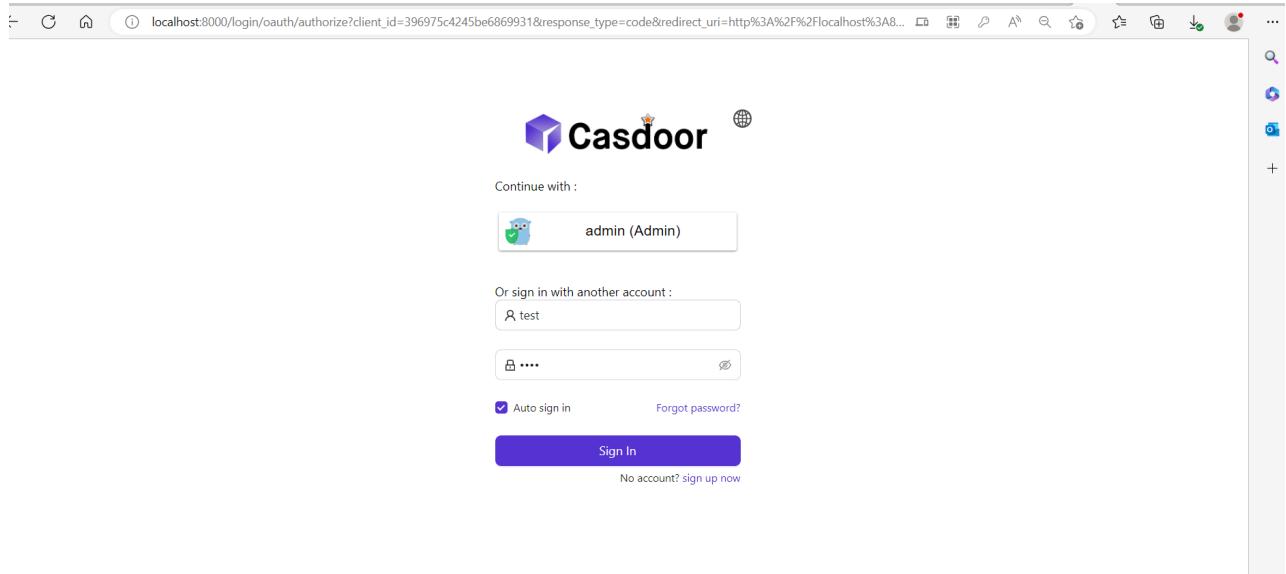
Since there is no client implementation, you can get your access token by making a request by below URL:

```
http://localhost:8000/login/oauth/authorize?client_id=<client
ID>&response_type=code&redirect_uri=http%3A%2Flocalhost%3A8080%2Flogin&scope=read&state=app-
built-in<public certificate>>
```

Change the client ID to the ID you copied before, and input the public certificate of Casdoor which you can find in Casdoor/Certs.

Then input the username and password you create for Hasura before.

Click Sign in.



Go back to Casdoor/Token page

localhost:8000/tokens

Name	Created time	Application	Organization	User	Authorization code	Access token	Action
b6ea3e35-abcf-41d8-a1a2-01f0fd8264b	2023-04-12 13:06:53	hasura	hasura	test	433b504b4f6a593e4a11	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
16024557-df21-4779-bfb9-959e5dae078c	2023-04-12 12:51:47	hasur	built-in	test1	2879fbcc282019cf7f23c	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
f3cb1070-c2d4-40f0-8bc0-59919d26d162	2023-04-11 15:04:00	hasura	hasura	test	2a370971798d403fc6ef	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
64993582-2322-4df7-ab20-cb23201bc77b	2023-04-11 00:37:22	springboot	built-in	admin	a2396037c3ba4fd9221e	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
f65a3813-a655-47f0-9c9a-f08ce4607815	2023-04-11 00:31:37	springboot	built-in	admin	d048c7f9cd1469fd829d	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
5828069e-15eb-4c92-933c-feada8ed621c	2023-04-11 00:06:54	springboot	built-in	admin	7cc27dc752cc4188ac8d	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
2277e0f2-7e78-462f-a654-3c53759784af	2023-04-11 00:05:17	springboot	built-in	admin	56141e709a06931b7faa	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
55bd324a-6039-40f6-b707-2a55d78ae911	2023-04-11 00:05:07	springboot	built-in	admin	9a1413bc172591a64353	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>
4b30acbe-fa22-4387-8098-9a46e70f6972	2023-04-10 23:59:19	springboot	built-in	admin	88b0997b675917f20fdc	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0e	<button>Edit</button> <button>Delete</button>

Find the Username you input before then click edit

Copy the Access Token.

Edit Token		<button>Save</button>	<button>Save &amp; Exit</button>
Name:	b6ea3e35-abcf-41d8-a1a2-01f0fd8264b		
Application:	hasura		
Organization:	hasura		
User:	test		
Authorization code:	433b504b4f6a593e4a11		
Access token:	eyJhbGciOiJSUz1NilsImtpZC16lmNcnQtYnVpbHQtaW4iLCj0eXAIoIKV1QifQ.eyJvd25ciI6lmhhc3VyYSlsm5hbWUiOj0ZXN0IwiY3JlYXRIZFRpbWUiOilyMDIzLTA0LTAsVDE1Oj50JU2KzA4OjAwliwidXBkYXRIZFRpbWUiOiiLCjPZC16ijRly		
Expires in:	604800		
Scope:	read		
Token type:	Bearer		
<button>Save</button>		<button>Save &amp; Exit</button>	

Now you can use the access token to make the authenticated request. Hasura returned the appropriate user rather than returning all the users from the database.

Hasura v2.22.0 API DATA ACTIONS REMOTE SCHEMAS EVENTS SETTINGS HELP Allow List

GraphQL REST

> GraphQL Endpoint  
✓ Request Headers

ENABLE	KEY	VALUE
<input type="checkbox"/>	Hasura-Client-Name	casdoor
<input checked="" type="checkbox"/>	content-type	application/json
<input type="checkbox"/>	x-hasura-admin-secret	*****
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6ImNlcnQtYnVpbHQtaW4iLCJ0eXAiOiJKV1QiLCJyvd25cIi6lmhhc3VyYSlsImShbWL
Enter Key		Enter Value

Explorer X GraphiQL ▶ Prettify History Explorer Code Exporter REST Derive action Analyze ◀ Docs

query MyQuery {  
 users {  
 id  
 username  
 }  
}

1 {  
 "data": {  
 "users": [  
 {  
 "id": "4ec7cce-ec7b-4191-a78d-e11f50686f8b",  
 "username": "test"  
 }  
 ]  
 }  
}

QUERY VARIABLES



> 集成 >

Python

# Python



JumpServer

Using CAS to connect JumpServer

# JumpServer

[Casdoor](#) can use CAS to connect [JumpServer](#).

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.

`JumpServer_HOSTNAME`: Domain name or IP where JumpServer is deployed.

## Step1. Deploy Casdoor and JumpServer

Firstly, the [Casdoor](#) and [JumpServer](#) should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME = http://localhost:8000`. When deploy Casdoor in `prod` mode. See [production mode](#).

## Step2. Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find a redirect url: `CASDOOR_HOSTNAME/cas/your organization/your application/login`
3. Add your redirect url to casdoor application: `JumpServer_HOSTNAME`

More information for [CAS](#)

# Step3. Configure JumpServer

## 1. You should find Auth

The screenshot shows the 'Auth' configuration page in the JumpServer web interface. The left sidebar has a 'Auth' option highlighted with a red arrow. The main content area has a 'Basic' tab selected, and a 'CAS' tab is also present, indicated by another red arrow. The 'Basic' section contains fields for 'Enable CAS Auth' (switched on), 'Server url' (http://101.43.192.216:8000/cas/jump/jumpServer/login), 'Proxy server url' (http://8.140.18.157), and 'Version' (3). The 'Other' section includes 'Logout completely' (switched on) and a 'User attr map' code editor containing:

```
1 = [
2   "uid": "username"
3 ]
```

Buttons for 'Reset' and 'Submit' are at the bottom.

## 2. You should config this app

This screenshot is identical to the one above, showing the 'Auth' configuration page. It highlights the 'Auth' tab in the sidebar and the 'CAS' tab in the main content area. Red arrows point to the 'Enable CAS Auth' switch, the 'Server url' field (containing http://101.43.192.216:8000/cas/jump/jumpServer/login), and the 'User attr map' code editor. The code in the editor is the same as in the previous screenshot.

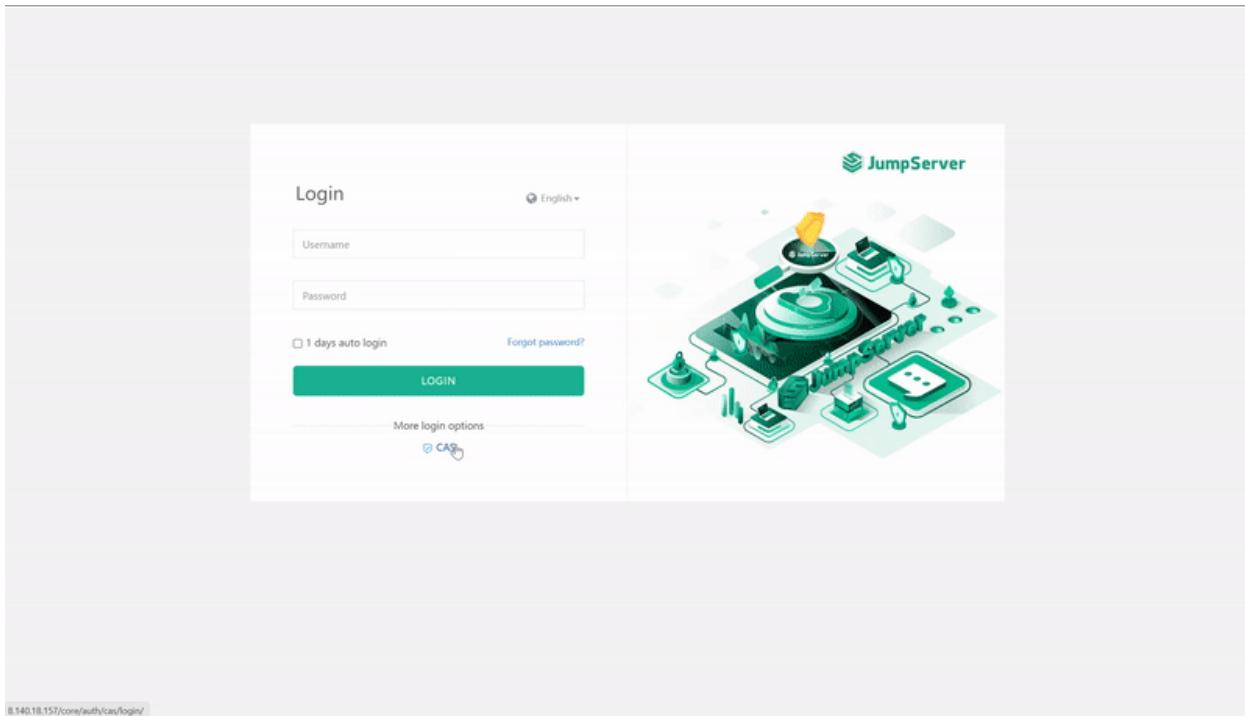
- /login endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/>

## login

- `/logout` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/logout>
- `/serviceValidate` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/serviceValidate>
- `/proxyValidate` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/proxyValidate>

More infomation for [CAS](#) and [JumpServer](#)

Log out of JumpServer, and test SSO.





&gt;

Monitoring

# Monitoring

## Web UI

Monitor runtime information on the casdoor Web page

## Prometheus

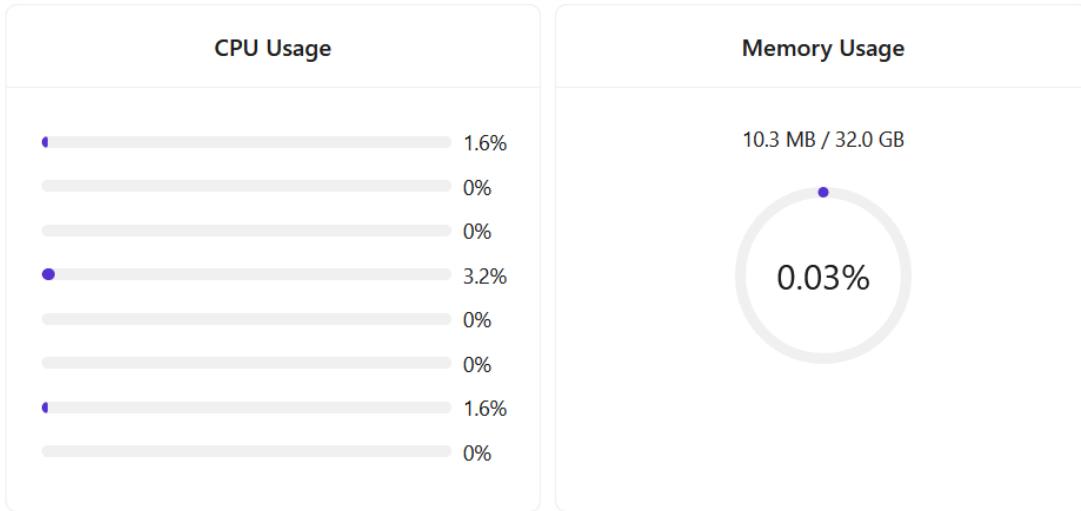
Use prometheus to collect information about running casdoor

# Web UI

You can monitor the runtime information of casdoor on the [casdoor Web page](#), including CPU Usage, Memory Usage, API Latency, and API Throughput

You can view the following information on the UI

- CPU Usage and Memory Usage



- API Latency, Including count times and average latency

API Latency				
Method	Endpoint	Latency (ms)	Throughput (req/s)	
GET	/api/get-cert	3	0.667	
GET	/api/get-certs	27	1.333	
GET	/api/get-chats	27	1.519	
GET	/api/get-default-application	3	5.333	
GET	/api/get-email-and-phone	1	1.000	
GET	/api/get-global-providers	58	1.000	

- API Throughput, Including total throughput and throughput per API

API Throughput			
Total Throughput: 2			
Name	Method	Throughput	
/api/get-prometheus-info	GET	1	
/api/get-system-info	GET	1	

# Prometheus

Use Prometheus to collect casdoor's runtime metrics, including API Throughput, API Latency, CPU Usage, Memory Usage, and so on, as you should configure your Prometheus profile

```
global:  
  scrape_interval: 10s #The time interval for fetching metrics  
  
scrape_configs:  
  - job_name: 'prometheus'  
    static_configs:  
      - targets: ['localhost:9090']  
  - job_name: 'casdoor' #Name of the application to be monitored  
    static_configs:  
      - targets: ['localhost:8000'] #Back-end address of casdoor deployment  
    metrics_path: '/api/metrics' #Path for collecting indicators
```

After the configuration is successful, you will find the following information in Prometheus







&gt;

国际化

# 国际化

Casdoor支持多种语言。 通过 [Crowdin](#)， 我们支持中文、法文、德文、俄文、日文和朝鲜文。

Casdoor使用官方的 Crowdin cli 来同步Crowdin 的翻译。 如果您想要添加更多语言支持，请在 [我们的社区](#)中提出， 如果您想帮助我们加快翻译工作， 可通过 [Crowdin](#)帮助我们翻译。

# 贡献者指南

欢迎使用 Casdoor！此文档是介绍如何为 Casdoor 做出贡献的指南

如果您发现有错误或缺失之处，欢迎留下意见或建议

## 开始参与

有许多方式可以为Casdoor做贡献。以下列出了一些贡献方式：

使用Casdoor并报告问题！使用 Casdoor 时，可以提出遇到的问题以促进Casdoor的开发，不管是漏洞还是提议都很欢迎。Before filing an issue on GitHub, it would be better to discuss first on [Discord](#) or [GitHub Discussions](#)

### ① 信息

创建issue时，请使用英文详细描述您的问题。

帮助编写文档！从文档开始贡献是一个很好的选择。

帮助解决issue！我们准备了一个表格，其中包含适合初学者的简单任务，挑战程度不同带有不同标签的标签，请查看此处的表格[Casdoor Easy Tasks](#)。

## 贡献

现在，如果您已经准备好创建 PR，在这里是贡献者的工作流：

1. Fork到您自己的仓库

2. 克隆您的fork到本地仓库
3. 创建一个新分支并在其上工作
4. 保持您的分支同步
5. 提交您的更改(请确保您的提交信息简洁)
6. 将你的提交推送到你的fork仓库中
7. 创建从您的分支到我们的**master**分支的合并请求。

## 合并请求

### 开始之前

Casdoor使用GitHub 作为其开发平台。 因此， 合并请求是贡献的主要来源。

在您打开拉取请求之前， 您需要知道一些基本准则：

- 当你第一次拉取请求时， 你需要签名 CLA。
- 解释您为什么要发送此 PR 以及此 PR 将会在仓库中做些什么。
- **只允许一次提交。** 请确保每个合并请求只做一件事， 否则请分开提交。
- 如果有新添加的文件，请将新文件顶部的 Casdoor 许可证包括在内。

```
// Copyright 2022 The Casdoor Authors. All Rights Reserved.  
//  
// Licensed under the Apache License, Version 2.0 (the "License");  
// you may not use this file except in compliance with the License.
```

## Semantic PRs

您的合并请求应遵循常规承诺的样本。 基本要求是有PR 标题或至少一个 提交消息。 例如，三个常用的PR标题如下：

### ⚠ 注意事项

PR 标题必须是小写的。

1. fix: a commit of the type `fix` patches a bug in your codebase.

```
fix: prevent racing of requests
```

2. feat: a commit of the type `feat` introduces a new feature to the codebase.

```
feat: allow provided config object to extend other configs
```

3. docs: a commit of the type `docs` add or improve a document.

```
docs: correct spelling of CHANGELOG
```

欲了解更多详情，请参阅 [常规承诺](#)

## 将 PR 链接到问题 (如果存在)

您可以将拉取请求链接到议题，以显示修复正在进行中，并在拉取请求被合并时自动关闭该议题。

## 使用关键词将拉取请求链接到议题

您可以通过在拉取请求说明或提交消息中使用支持的关键词将拉取请求链接到议题。 拉取请求**必须**在默认分支上。

- close
- fix
- resolve

同一仓库中的问题，例如：

```
Fix: #902
```

欲了解更多详情，请参阅 [Link PR to issue](#)。

## 修改PRs

您的 PR 可能不可避免地需要修改。 当代码需要更改时，请 **重新使用同一PR**。 不要关闭 PR 并创建新的 PR

下面是一个示例。

- 修改本地代码。
- 修改此提交。

```
git commit --amend
```

- 推送代码到远程仓库

```
git push --force
```

然后，PR 已成功修改！您可以在casdoor仓库中查看。

## 相关代码

一些原则：

可读性 - 重要代码应有充分的文件记录。代码风格应与现有的风格一致。

## 命名规范

例如，`signupUrl`对于变量名字，`Signup URL`对于前端UI显示

## 如何更新 i18n 数据？

请注意，我们使用 [Crowdin](#) 作为翻译平台和i18next作为翻译工具。当您在`web/`目录中使用i18next添加一些单词时，您可以运行`i18n/generate_test.go` 自动生成`web/src/locales/**/data.json`。

Run `i18n/generate_test.go`:

```
cd i18n && go test
```

默认情况下，所有语言都以英文填写。鼓励您在您的 PR 已被合并后用 [Crowdin](#)帮助翻译在`web/src/locales/zh/data.json`新添加的字符串

### ⚠ 注意事项

如果您不熟悉其他语言，请不要翻译。保持文件原样。

# 许可证书

为Casdoor作出贡献，即代表您同意您的贡献将遵循 Apache 许可协议。