|  |  |
| --- | --- |
|  | 北京九州云腾科技有限公司 |

|  |
| --- |
| **JWT单点登录(SSO)集成文档** |

|  |
| --- |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **编制人：** | 陈淘 | **日　期：** | 2017-12-14 |  |
|  |  |  |  |  |  |
|  | **标检人：** |  | **日　期：** |  |  |
|  |  |  |  |  |  |
|  | **审核人：** |  | **日　期：** |  |  |
|  |  |  |  |  |  |
|  | **批准人：** |  | **日　期：** |  |  |

|  |
| --- |
| **北京九州云腾科技有限公司发布** |

**变更记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **变更内容** | **拟制/更改人** | **日期** |
| V1.1 | 内容更新 | 陈淘 | 2018-06-06 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

**[简介 5](#_Toc7797)**

**[实现原理 5](#_Toc30351)**

[插件式SSO-JWT原理 6](#_Toc2224)

**[添加JWT应用 6](#_Toc10161)**

**[JAVA插件式集成 8](#_Toc10911)**

[配置环境 8](#_Toc22288)

[接收令牌 9](#_Toc30279)

[解析令牌 9](#_Toc20593)

**[PHP插件式集成 11](#_Toc29982)**

[配置环境 11](#_Toc6394)

[接收令牌 11](#_Toc14339)

[解析令牌 13](#_Toc29118)

**[.NET插件式集成 15](#_Toc7161)**

[配置环境 15](#_Toc817)

[接收令牌 16](#_Toc5953)

[解析令牌 16](#_Toc19353)

**[Python插件式集成 17](#_Toc13285)**

[下载资源库 17](#_Toc9778)

[接收令牌 18](#_Toc23593)

[解析令牌 18](#_Toc10332)

**[处理结果 20](#_Toc31595)**

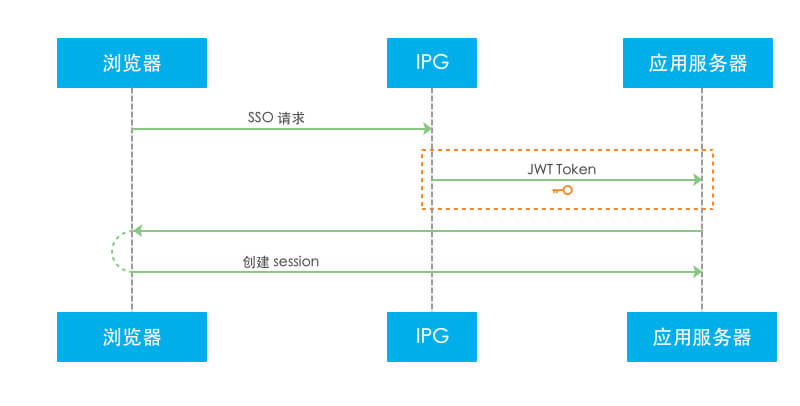
**[返回码 20](#_Toc5493)**

**[相关下载 21](#_Toc7975)**

### 简介

整个插件式JWT的流程，是接收IPG平台向callback url发出的id\_token参数（即JWT令牌），并使用我们提供的（或第三方提供的）JWT解密库/方法对jwt进行解析，并验证身份。   
你可以自己按照这个逻辑完成代码，也可以在我们提供的代码/demo的基础上进行修改。

### 实现原理



插件式SSO-JWT原理

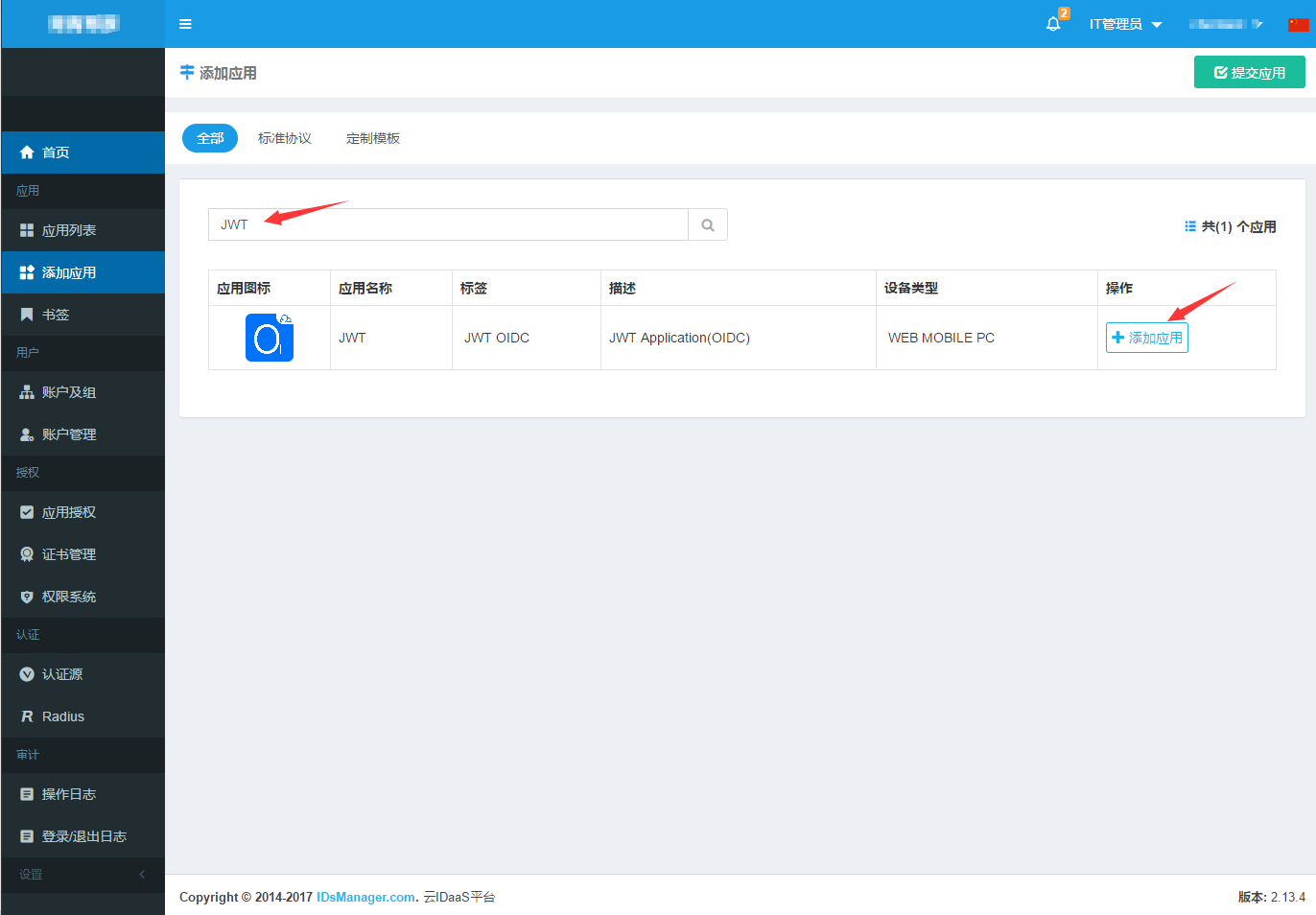
1. 通过浏览器登录集成的IPG系统后，确认要单点登录的应用，发起SSO请求到IPG系统。

2. IPG生成token令牌发送到业务应用。

3. 你的应用获取到token令牌，用我们提供的插件或方法解析token令牌，解析成功获取到用户信息并验证后，重定向进行登录；如果解析失败则拒绝登录。

### 添加JWT应用

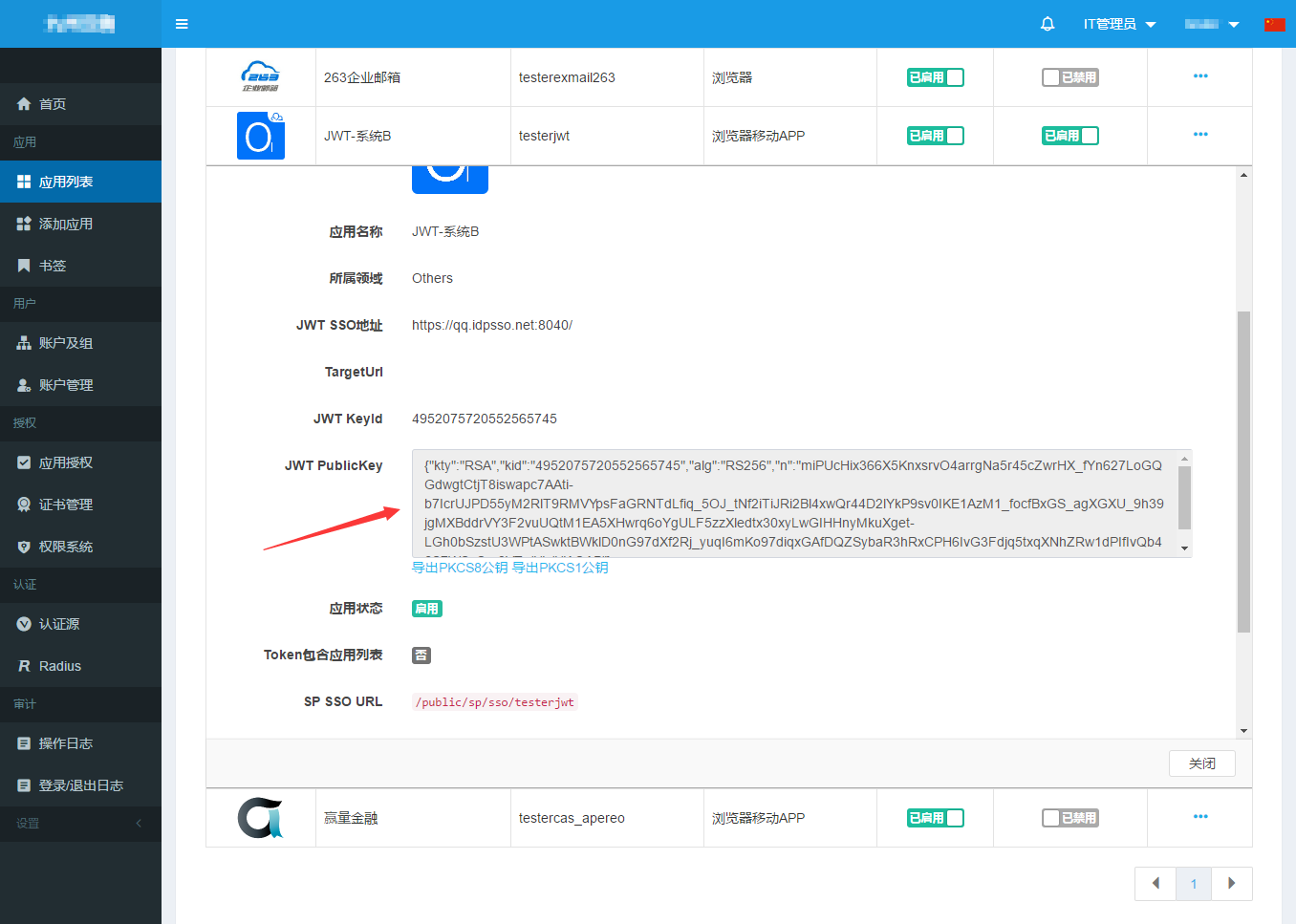
1. 在应用中添加一个JWT应用：



1. 填写应用信息完成后保存：



1. 添加完应用之后，点击应用详情获取publicKey：



至此准备工作完成，我们开始进行服务器端的快速开发。

### JAVA插件式集成

#### **配置环境**

JDK1.7以上

请下载[JWT-JAVA-SDK下载](https://file.idsmanager.com/static/v2/download/java/JWT-SDK-1.0.jar)，下载下来的.jar包含了我们封装好的帮助方法，jar包请引入你开发的应用中。

在申请JWT应用步骤的结尾获取到的publicKey

**接收令牌**

//id\_token 是IPG请求时带来的，在body里获取，publicKey是在IPG里注册应用时生成的，注册完可见，此示例代码是获取用户信息。

//JWT SSO

@RequestMapping(value = "/jwt/sso/login")

public String ssoUrl(@RequestParam String id\_token, String redirect\_url, Model model, HttpServletRequest request) {

//1.接收方法为GET方式,参数名为id\_token

//2.<解析令牌>为解析id\_token并验证代码

}

#### **解析令牌**

//1.使用公钥，解析id\_token

// 使用publicKey解密上一步获取的id\_token令牌

DingdangUserRetriever retriever = new DingdangUserRetriever(id\_token, publicKey);

DingdangUserRetriever.User user = null;

try {

//2.获取用户信息

user = retriever.retrieve();

} catch (Exception e) {

LOG.warn("Retrieve SSO user failed" , e);

return "error";

}

//3.判断用户名是否在自己系统存在isExistedUsername()方法为业务系统自行判断数据库中是否存在

if (isExistedUsername(user.getUsername())) {

//4.如果存在,登录成功，返回登录成功后的页面

User spUser = userService.updateLoginTimes(user.getUsername());

request.getSession().setAttribute(HttpSessionSecurityContextRepository.SPRING\_SECURITY\_CONTEXT\_KEY, saveSecurity(spUser));

//5.如果注册时添加redirect\_url，那么返回此自定义url页面

if (StringUtils.isNotEmpty(redirect\_url)) {

return "redirect:" + redirect\_url;

}

//6.否则返回系统默认操作页面

return "redirect:../../index";

} else {

//7.如果不存在,返回登录失败页面,提示用户不存在

model.addAttribute("error", "username { " + user.getUsername() + " } not exist");

return "error";

}

### PHP插件式集成

#### **配置环境**

在本例中，我们使用composer管理第三方库（可选）

公钥：开发前需要在IT管理员权限下前往应用->详细->导出PKCS8公钥来获取解密JWT用的公钥，并安全地放置在能访问到的目录内。

#### **接收令牌**

JWT的id\_token将会以url参数的方式传进callback页面，我们直接将其读取出来：

/\* 使用composer 载入 php-jwt第三方库

\* 命令行 composer require firebase/php-jwt

\* 库链接：https://github.com/firebase/php-jwt

\* 我们使用Firebase的这个第三方库来实现对jwt的解密，如果不用composer的话，请自行添加源文件

\* 你也可以使用其他能对JWT token进行RS256解码的工具或库

\*/

// 在这里将JWT库引入，在这里为了便捷demo直接使用

// 推荐使用

require 'vendor/autoload.php';

use \Firebase\JWT\JWT;

// 本地存储public key公钥的位置

$public\_key\_location = "LOCATION/TO/YOUR/PUBLIC-KEY/XXX.pem";

// 读取公钥信息，公钥在这里存储在一个.pem文件内

$public\_key = file\_get\_contents($public\_key\_location);

// 从url的参数中读取id\_token，即令牌

if (!empty($\_GET["id\_token"])) {

$jwt = $\_GET["id\_token"];

// 这里继续第二步：解析令牌

}

#### **解析令牌**

通过第三方库php-jwt，使用公钥对收到的JWT token（即id\_token）进行解密，获取到用户信息并错误码为0的话，验证通过跳转到登录页面，失败则拒绝：

try {

/\*\*

\* You can add a leeway to account for when there is a clock skew times between

\* the signing and verifying servers. It is recommended that this leeway should

\* not be bigger than a few minutes.

\*

\* Source: http://self-issued.info/docs/draft-ietf-oauth-json-web-token.html#nbfDef

\*/

// Firebase的JWT库的一个参数，不出问题的话可以忽略

// （可选）当服务器时间与本地时间不符时，可以通过这个leeway参数来调整容错

JWT::$leeway = 60; // $leeway in seconds

// 使用公钥、使用RS256算法对JWT（即第一步传进来的id\_token）进行解密进行解密

$decoded = JWT::decode($jwt, $public\_key, array('RS256'));

// 将解密的结果从class转化成PHP array

$decoded\_array = (array) $decoded;

// 打印出解密的结果，成功！

print("解密结果:<br>");

foreach ($decoded\_array as $key => $value) {

print $key . ": " . $value . "<br>";

}

// 获取到用户信息后，判断该用户是否存在于你的系统内

if (userExistsInSystem()) {

// 如果存在，那么登录成功，跳转到登录后页面

} else {

// 如果不存在，那么登录失败，跳转到显示错误页面

}

} catch(Exception $e) {

print "错误：" . $e->getMessage();

}

### .NET插件式集成

#### **配置环境**

.NET Framework 4及以上

[.NET SDK下载](https://file.idsmanager.com/static/v2/download/net/IDsManagerJwt-SDK-0.1.rar)

#### **接收令牌**

#### 牌

#### //id\_token 是IPG请求时带来的，在body里获取，publicKey是在IPG里注册应用时生成的，注册完可见,此示例代码是获取用户信息。

#### //JWT SSO

#### [Route("jwt/sso/login")]

#### public ActionResult ssoUrl(String id\_token)

#### {

#### //1.接收方法为GET方式,参数名为id\_token

#### //2.<解析令牌>为解析id\_token并验证代码

#### }

#### **解析令牌**

publicKey: 解析令牌的过程中，我们会使用到应用的publicKey。请在 JWT应用->详细 中将publicKey字段对应的内容拷贝并存储起来。

//1.使用公钥,解析id\_token

string username;

DingdangSDK.DingdangUserRetriever retriever = new DingdangSDK.DingdangUserRetriever(id\_token, publickey);

DingdangSDK.User user = null;

//2.获取用户信息

user = retriever.retrieve();

username = user.sub;

//3.判断用户名是否在自己系统存在

//4.如果存在,登录成功，返回登录成功后的页面

//5.如果注册时添加redirect\_url，那么返回此自定义url页面

//6.否则返回系统默认操作页面

//7.如果不存在,返回登录失败页面,提示用户不存在

### Python插件式集成

#### **下载资源库**

本Python JWT示例使用PyJWT库来进行JWT的解密

公钥：开发前需要在IT管理员权限下前往应用->详细->导出PKCS8公钥来获取解密JWT用的公钥，并安全地放置在能访问到的目录内。

JWT的token将会以url参数的方式传进来，解密后进行认证判断

#### // 库的github链接 https://github.com/jpadilla/pyjwt

#### pip install PyJWT

#### // 注：CentOS系统如果使用时无法导入算法 RSAAlgorthm时需要下载pyjwt的2个依赖包

#### yum install ibffi-devel

#### pip install cryptography

#### **接收令牌**

#### def get\_idToken(token):

#### if not token.strip():

#### print('token信息不能为空')

#### else:

#### //这里继续第二步：解析令牌

#### get\_idToken(my\_idtoken); // 运行程序

#### **解析令牌**

## 2.解析令牌

通过jwt解密库，使用公钥对传入的id\_token进行解密。将公钥以字符串的形式从文件中读取出来，并作为key进行解密：

// 引入用到的包文件

import jwt

import json

from jwt.algorithms import RSAAlgorithm

from jwt.utils import force\_bytes

from utils import key\_path // 本例中key\_path辅助方法是写在utils工具类中的

def get\_user\_ifon(id\_token):

try:

algo = RSAAlgorithm(RSAAlgorithm.SHA256)

pem\_key = open(key\_path('D:\pythonDemo\key\public\_key\_pkc8.pem'), 'r')

public\_key = algo.prepare\_key(pem\_key.read())

token\_info = jwt.decode(force\_bytes(id\_token), key=public\_key,verify=True)

user\_info = json.loads(json.dumps(token\_info))

username = user\_info['sub']

print(username)

# 3.判断用户名是否在自己系统存在

# 4.如果存在, 登录成功，返回登录成功后的页面

# 5.如果注册时添加redirect\_url，那么返回此自定义url页面

# 6.否则返回系统默认操作页面

# 7. 如果不存在, 返回登录失败页面, 提示用户不存在

except Exception as e:

print(e)

上面用到的key\_path方法是用来获取public key位置的辅助方法，具体如下：

def key\_path(key\_name):

return os.path.join(os.path.dirname(os.path.realpath(\_\_file\_\_)), 'keys', key\_name)

### 处理结果

解析后数据的数据为JSON，校验数据后进行登录处理

请求成功示例数据：

{"username":"zhangsan","state","",error:"0","errorMes":"operation success"}

返回参数：

|  |  |
| --- | --- |
| **常量** | **说明** |
| username | 子账户(业务应用)用户名 |
| state | 自定义参数，原值返回 |
| error | 错误码，0表示成功 |
| errorMsg | 错误信息描述 |

## 返回码

|  |  |  |
| --- | --- | --- |
| **字段** | **错误码** | **说明** |
| error | 0 | 表示操作成功 |
| error | 101 | code不合法 |
| error | 102 | state不合法 |
| error | 103 | 尚未关联主账号 |

## 

## 相关下载

JAVA 相关下载

[Java\_无插件式CAS Demo下载](https://file.idsmanager.com/static/v2/download/java/Java-CAS-Demo.zip)   [Java\_插件式JWT Demo下载](https://file.idsmanager.com/static/v2/download/java/Java-JWT-Demo.zip)

[Java\_插件式JWT JDK1.6下载](https://file.idsmanager.com/static/v2/download/java/jwt/JWT-SDK-1.1.0_1.6.zip)  [Java\_插件式JWT JDK1.7下载](https://file.idsmanager.com/static/v2/download/java/jwt/JWT-SDK-1.1.0_1.7.jar)

[Java\_插件式JWT JDK1.8下载](https://file.idsmanager.com/static/v2/download/java/jwt/JWT-SDK-1.1.0_1.8.jar)

PHP 相关下载

[PHP-CAS-无插件式资料下载](https://file.idsmanager.com/static/v2/download/php/PHP-CAS-Demo.zip)   [PHP-JWT-插件式资料下载](https://file.idsmanager.com/static/v2/download/php/PHP-JWT-Demo.zip)

.NET 相关下载

[.NET JWT 插件式资料下载](https://file.idsmanager.com/static/v2/download/net/NET-JWT-Demo.rar)   [.NET JWT+CAS Demo下载](https://file.idsmanager.com/static/v2/download/net/NET-JWT+CAS-Demo.rar)

Python 相关下载

[Python-JWT-插件式资料下载](https://file.idsmanager.com/static/v2/download/python/Python-JWT-Demo.rar)