# 深度剖析

## OAuth2和微服务安全架构


**讲师：杨波**

研发总监/资深架构师

# 课程大纲

01 课程概述

02 问题域和场景

03 **OAuth2**定义和原理

04 典型OAuth Flow和选型

05 OAuth2授权服务器和资源服务器案例实操（lab）

06 OAuth2客户端案例实操（lab）

07 **JWT**令牌原理

08 JWT案例实操（lab）

# 课程大纲

09 **Android无线应用**接入OAuth2案例实操(lab)

10 **Angularjs单页应用**接入OAuth2案例实操（lab）

11 Github**社交登录**案例实操（lab）

12 OAuth2安全风险和案例实操（lab）

13 OpenId Connect简介

14 下一代微服务安全架构

15 参考资源+后续课程预览

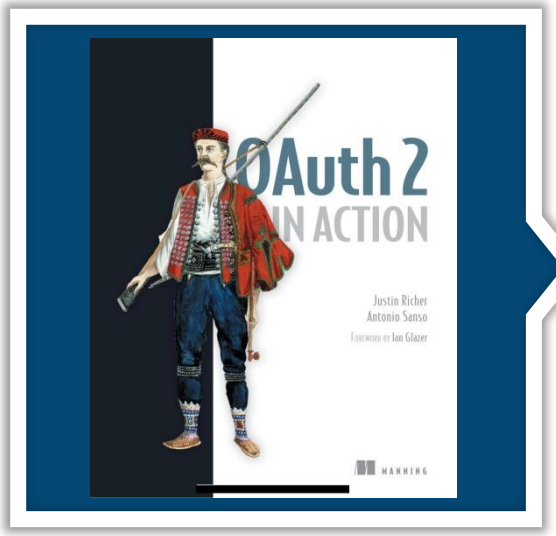
# 第 1 部分

# 课程概述

# 课程概述


波波老师的微服务基础架构体系的第一个模块
结合案例和实操(Spring Security OAuth2)
面向原理和架构
01
02
03
04
05
06
OAuth2协议深度剖析
结合开源项目
面向微服务

# 课程参考书




## OAuth2 in Action

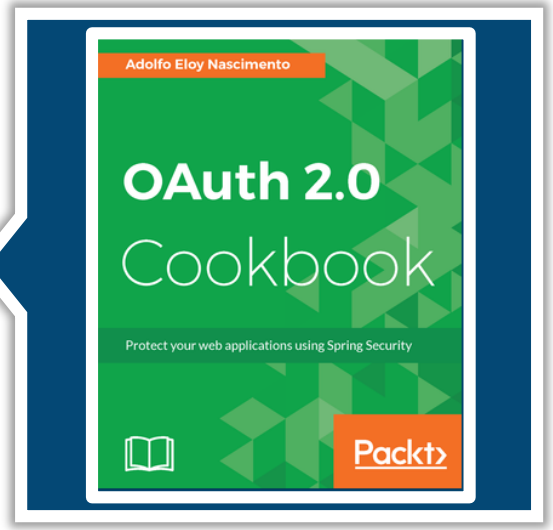
https://www.manning.com/books/oauth-2-in-action

原理

## OAuth 2.0 Cookbook

https://www.packtpub.com/virtualization-and-cloud/oauth-20-cookbook

实践



# 波波老师的微服务基础架构体系2018预览(draft)

01 OAuth2授权认证中心架构和实践

02 微服务配置中心Apollo架构和实践

03 调用链监控CAT架构和实践

04 微服务网关Zuul架构和实践

05 容错限流Hystrix/Turbine架构和实践

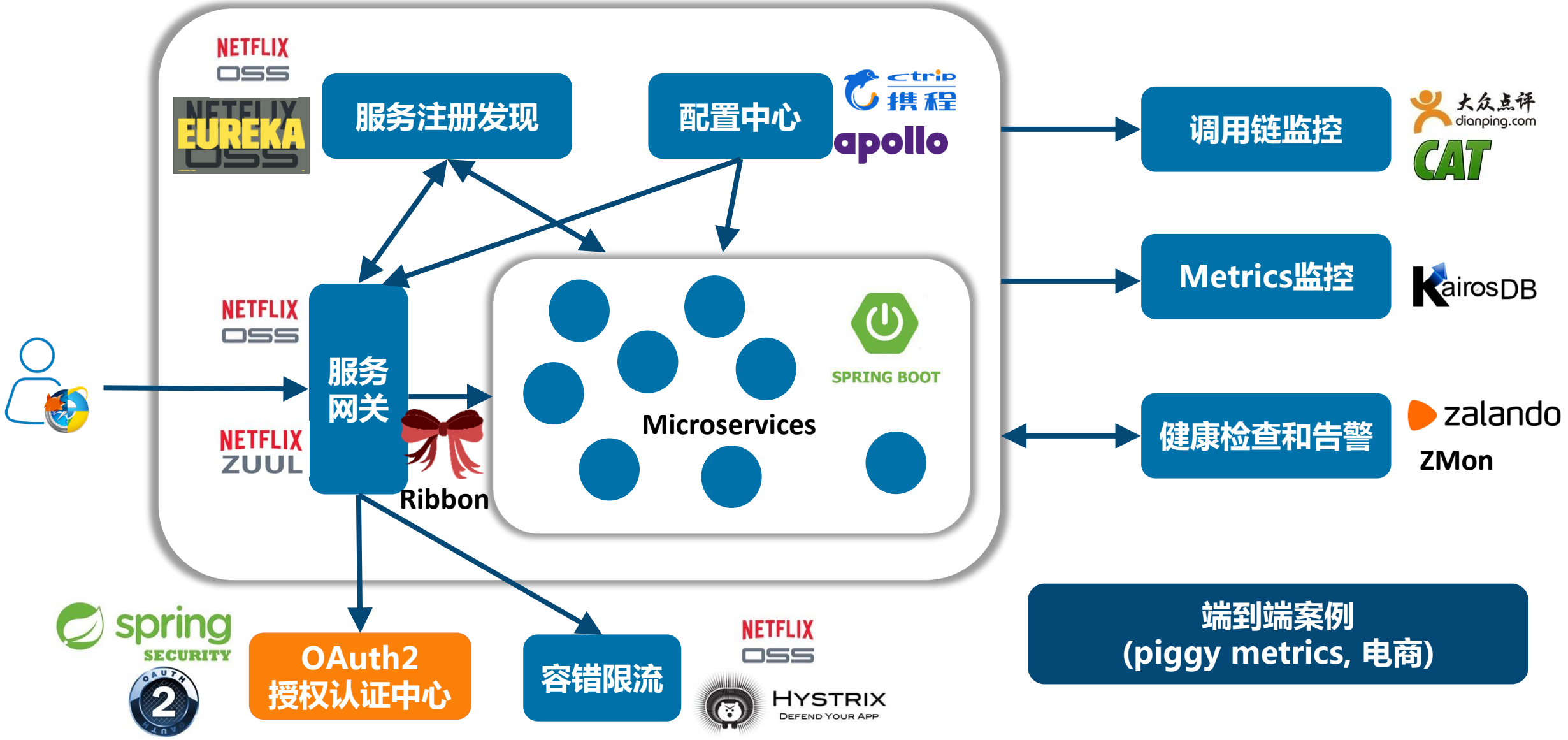
06 微服务注册发现Eureka/Ribbon架构和实践

07 时间序列监控KairosDB架构和实践

08 微服务监控告警ZMon架构和实践

09 综合案例分析

# 架构和技术栈预览


NETFLIX OSS
EUREKA
服务注册发现
配置中心
ctrip 携程
apollo
调用链监控
大众点评 dianping.com
CAT
NETFLIX OSS
服务网关
NETFLIX ZUUL
Ribbon
Microservices
SPRING BOOT
Metrics监控
KairosDB
健康检查和告警
zalando
ZMon
spring SECURITY
OAuth2 授权认证中心
容错限流
NETFLIX OSS
HYSTRIX DEFEND YOUR APP
端到端案例
(piggy metrics, 电商)

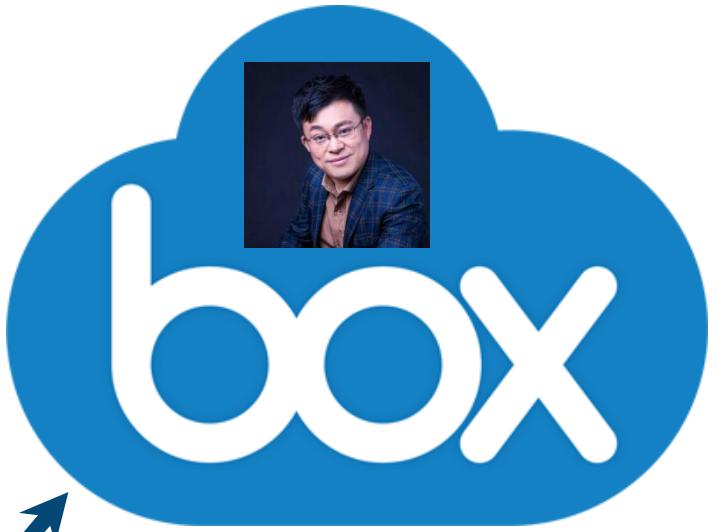
# 第 2 部分

# 问题域

# 开放系统间授权

相片拥有者

云冲印服务




云存储服务

七牛

第三方应用

# 图例

# 办法1：密码用户名复制

# 办法2：万能钥匙

资源拥有者

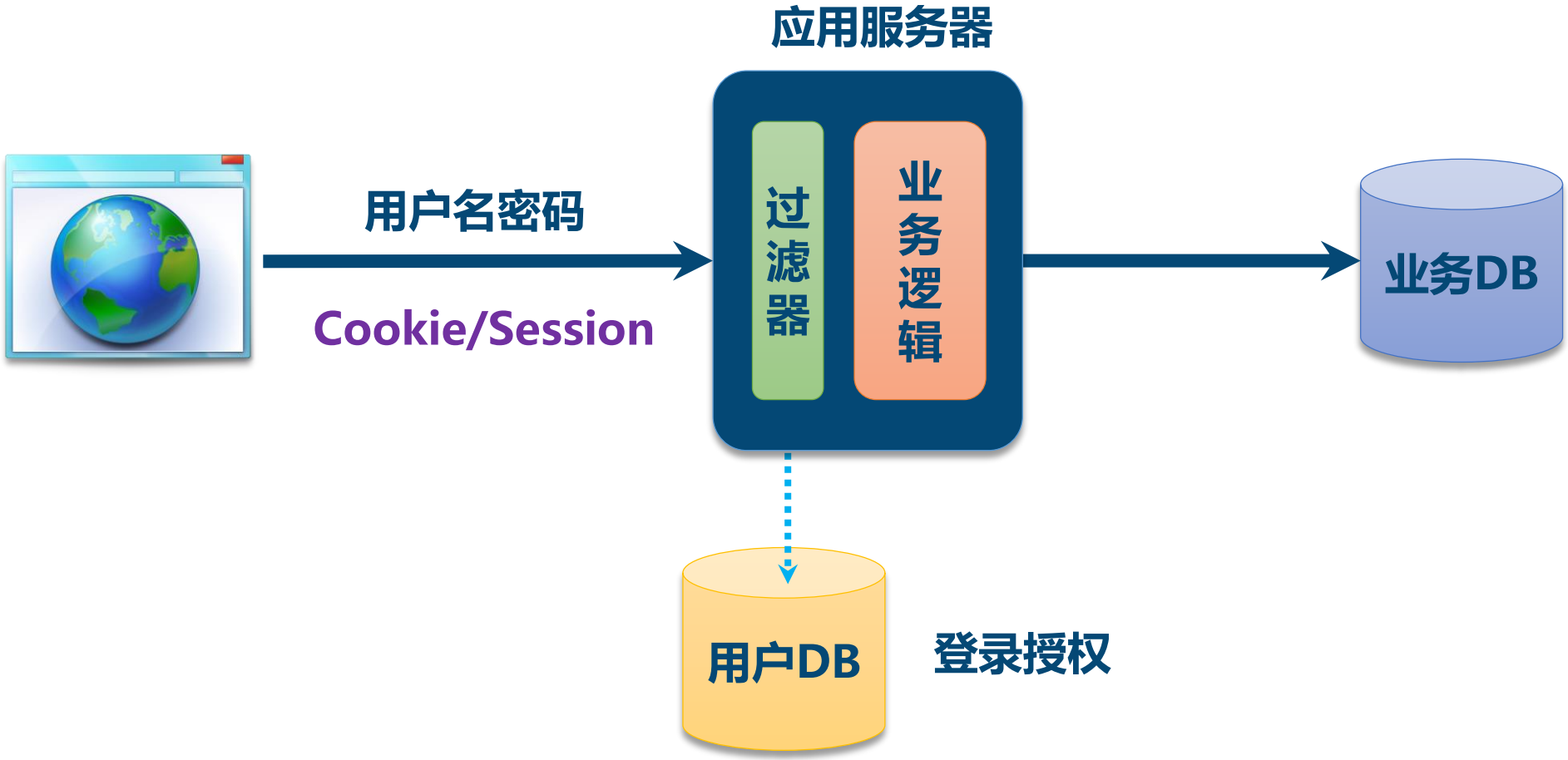
通用developer key

客户应用


受保护的资源

# 办法3：特殊令牌

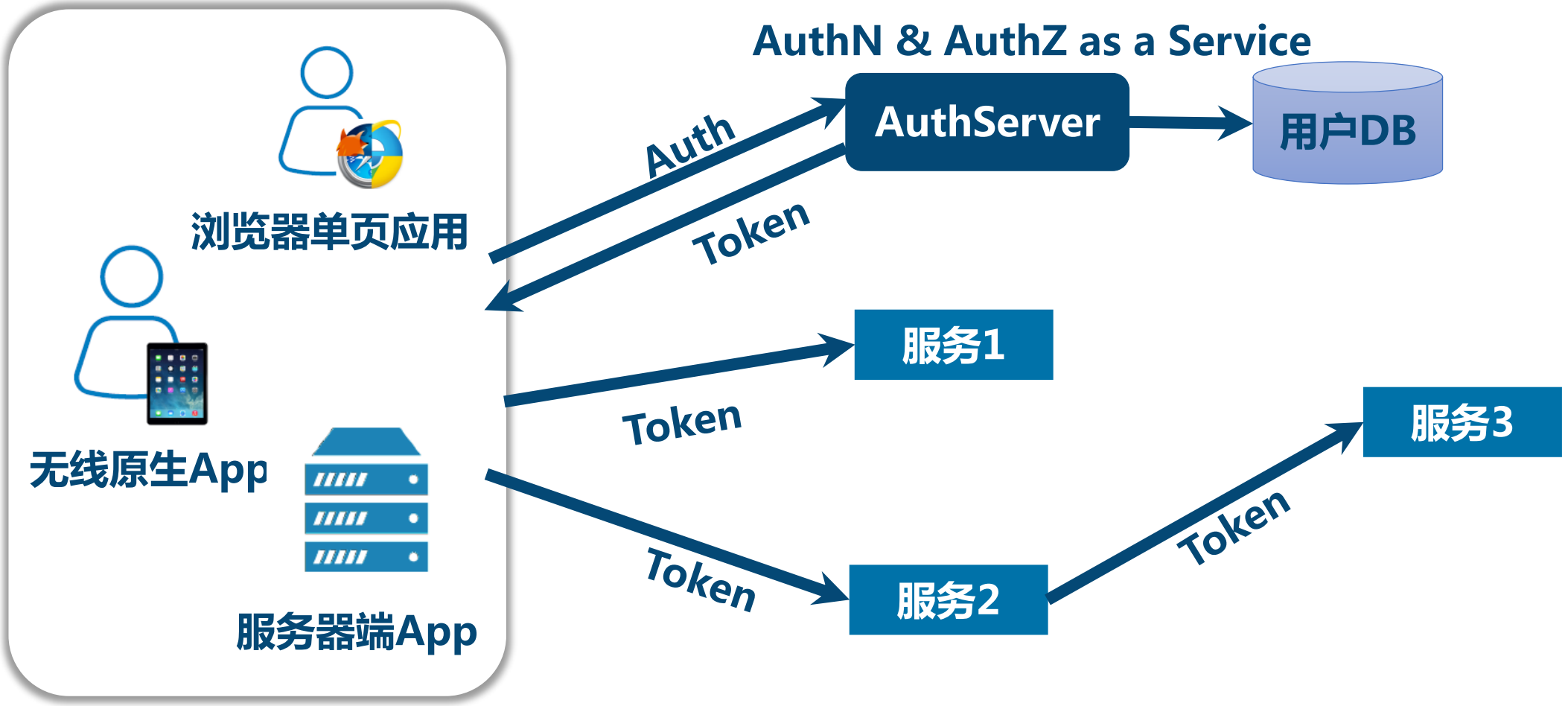
# 传统单块应用安全




登录工程：传统Web应用中的身份验证技术

http://insights.thoughtworkers.org/traditional-web-app-authentication/

# 现代微服务安全


AuthN & AuthZ as a Service
AuthServer
用户DB
Auth
Token
浏览器单页应用
无线原生App
服务器端App
服务1
服务2
服务3
Token
Token
Token

# 你见过OAuth吗?


11:11
53%
返回
微信登录
微信安全登录
该网页由大众点评开发，向其提供以下权限即可继续操作
获得你的公开信息（昵称、头像等）
确认登录


新浪微博 与微博连接
没有微博帐号？ 注册
使用你的微博帐号访问 淘宝网，更方便的分享精彩生活
将允许网站进行以下操作：
获得你的个人信息，好友关系
分享内容
获得
在 我的设置 里取消授权。
淘宝网
http://www.taobao.com/
共 2227972 人连接该网站
帐号：
请输入新浪微博账号
密码：
我已阅读新浪网络服务使用协议
连接
取消


微博
weibo.com
注册
授权 哈哈哈微博 访问你的微博帐号
643055866@qq.com
登录


微博
weibo.com
小小小笑话羽
哈哈哈微博
7567
将允许 哈哈哈微博 进行以下操作：
获得你的个人信息，好友关系
分享内容到你的微博
获得你的评论
取消
授权

# 总结：OAuth2解决问题域和场景


**开放系统间授权**

社交联合登录

开放API平台

**现代微服务安全**

单页浏览器App（HTML5/JS/无状态）

无线原生App

服务器端WebApp

微服务和API间调用

**企业内部应用认证授权（IAM/SSO）**

# 第3部分

# OAuth2定义和原理

# OAuth2最简向导

# 什么是OAuth 2.0

用于REST/APIs的代理授权框架(**delegated authorization** framework)

解耦认证和授权


基于**令牌Token**的授权，在无需暴露用户密码的情况下，使应用能获取对用户数据的有限访问权限

事实上的标准安全框架，支持多种用例场景

- 服务器端WebApp
- 浏览器单页SPA
- 无线/原生App
- 服务器对服务器之间

# 令牌类比仆从钥匙(Valet Key)

**给应用授予有限的访问权限，让应用能够代表用户去访问用户的数据**

# OAuth 2.0历史


大致开始于2007
2010 – RFC 5849定义了OAuth 1.0
2010 – IETF开始OAuth 2.0制定工作
干系人：Google, Microsoft, Facebook, Github, Twitter, Flickr, Dropbox
2012年中 – 第一作者和编辑退出，并将其名字从所有规范中删除（戏剧性）
2012年10月- RFC 6749, RFC 6750

# OAuth 2.0优势

01 OAuth 2.0比OAuth 1.0易于实现

02 更安全，客户端不接触用户密码，服务器端更易集中保护

03 广泛传播并被持续采用

04 短寿命和封装的token

05 资源服务器和授权服务器解耦

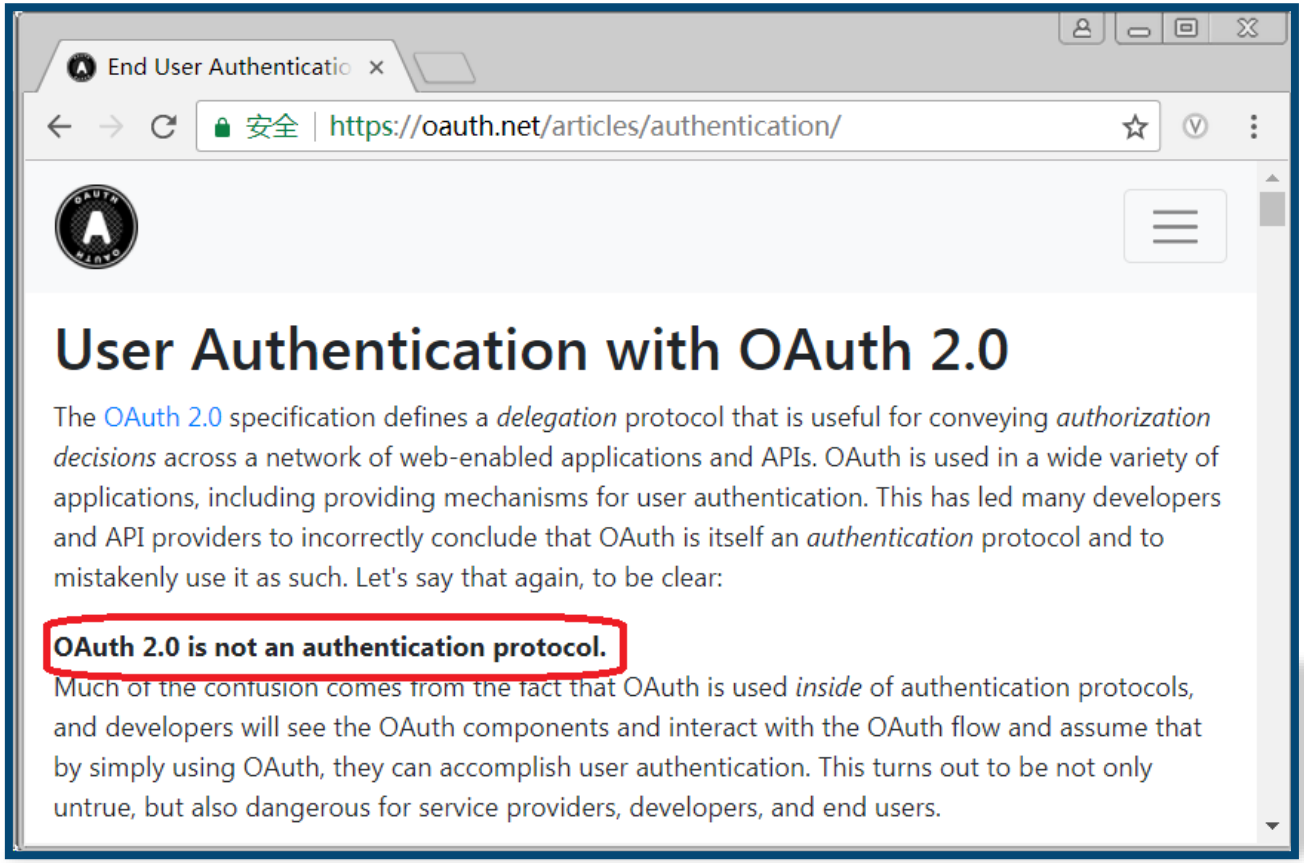
06 集中式授权，简化客户端

07 HTTP/JSON友好，易于请求和传递token

08 考虑多种客户端架构场景

09 客户可以具有不同的信任级别

# OAuth 2.0不足

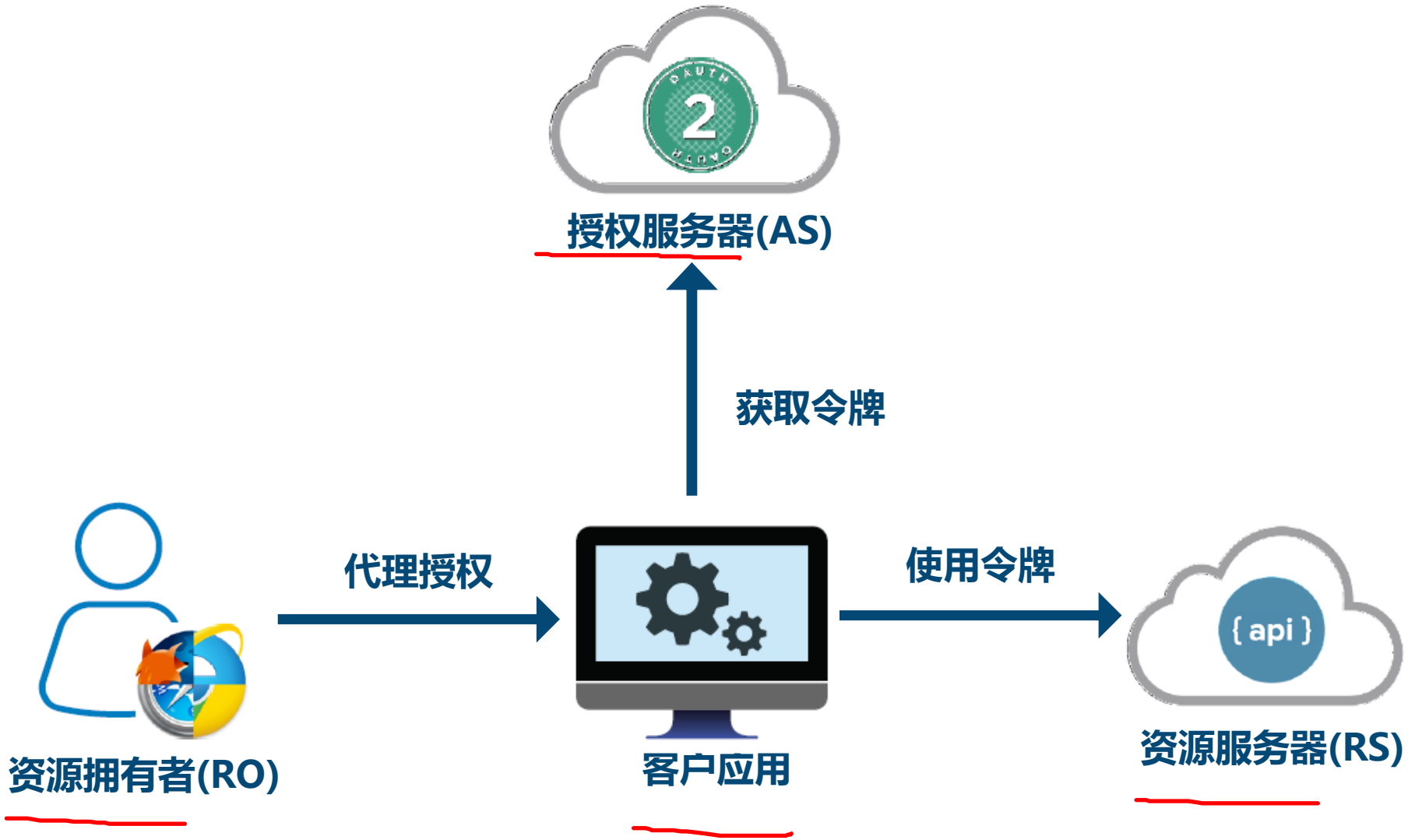



协议框架太宽泛，造成各种实现的兼容性和互操作性差

和OAuth 1.0不兼容

OAuth 2.0不是一个认证协议，OAuth 2.0本身并不能告诉你任何用户信息

# OAuth 2.0主要角色



# OAuth术语

## 客户应用
(Client Application)

通常是一个Web或者无线应用，它需要访问用户的受保护资源

## 资源服务器
(Resource Server)

是一个web站点或者web service API，用户的受保护数据保存于此

## 授权服务器
(Authorized Server)

在客户应用成功认证并获得授权之后，向客户应用颁发访问令牌Access Token

## 资源拥有者
(Resource Owner)

资源的拥有人，想要分享某些资源给第三方应用

# OAuth术语

## 客户凭证
(Client Credentials)

客户的clientId 和密码用于认证客户

## 令牌
(Tokens)

授权服务器在接收到客户请求后，颁发的访问令牌

## 作用域
(Scopes)

客户请求访问令牌时，由资源拥有者额外指定的细分权限(permission)

# OAuth令牌类型

## 授权码(Authorization Code Token)

仅用于授权码授权类型，用于交换获取访问令牌和刷新令牌

## 刷新令牌(Refresh Token)

用于去授权服务器获取一个新的访问令牌


## Bearer Token

不管谁拿到Token都可以访问资源，像现钞

## 访问令牌(Access Token)

用于代表一个用户或服务直接去访问受保护的资源

## Proof of Possession(PoP) Token

可以校验client是否对Token有明确的拥有权

# OAuth 2.0误解

- OAuth并没有支持HTTP以外的协议
- OAuth并不是一个认证协议
- OAuth并没有定义授权处理机制
- OAuth并没有定义token格式
- OAuth 2.0并没有定义加密方法
- OAuth 2.0并不是单个协议
- OAuth2.0仅是授权框架，仅用于授权代理

# 回顾

2 **OAuth是一种在系统之间的代理授权**(delegation authorization)**协议**

1 **OAuth本质**

如何获取token
如何使用token

4 **OAuth使用代理协议的方式解决密码共享反模式问题**

3 **OAuth提供一个宽泛的协议框架，具体安全场景需要定制**

# 第 4 部分

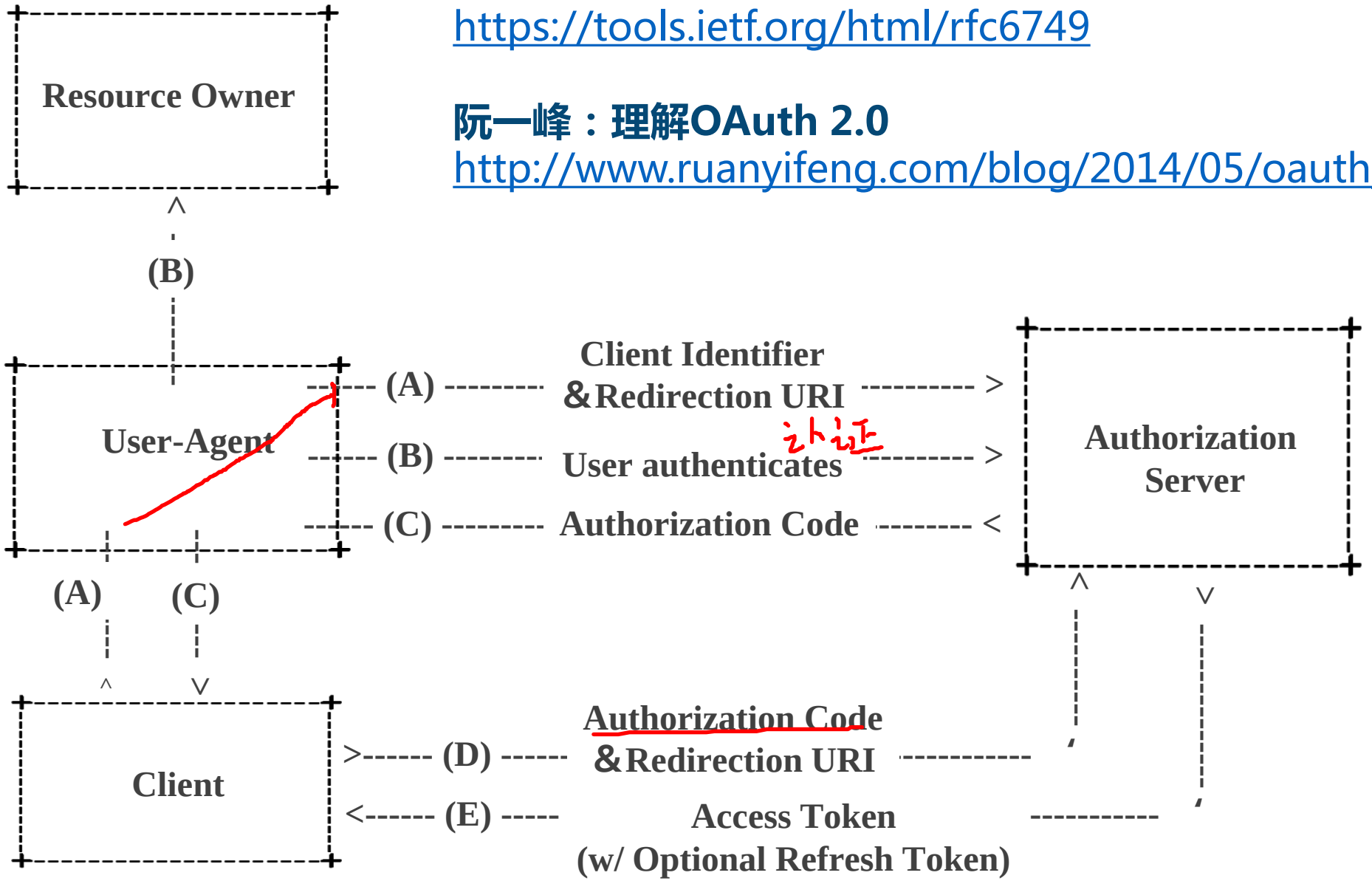
# 典型OAuth Flow和选型

# 授权码模式

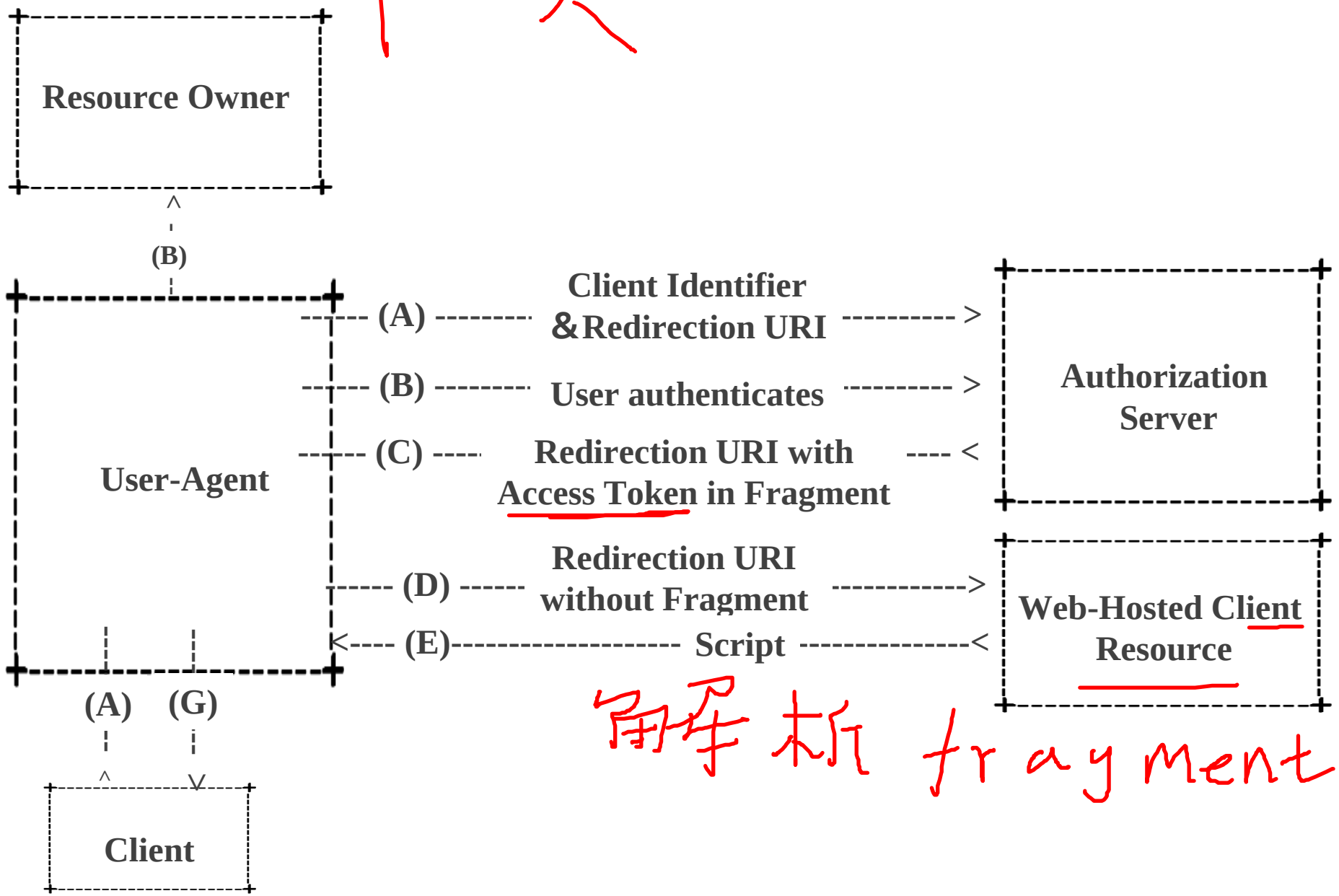
最复杂

https://tools.ietf.org/html/rfc6749

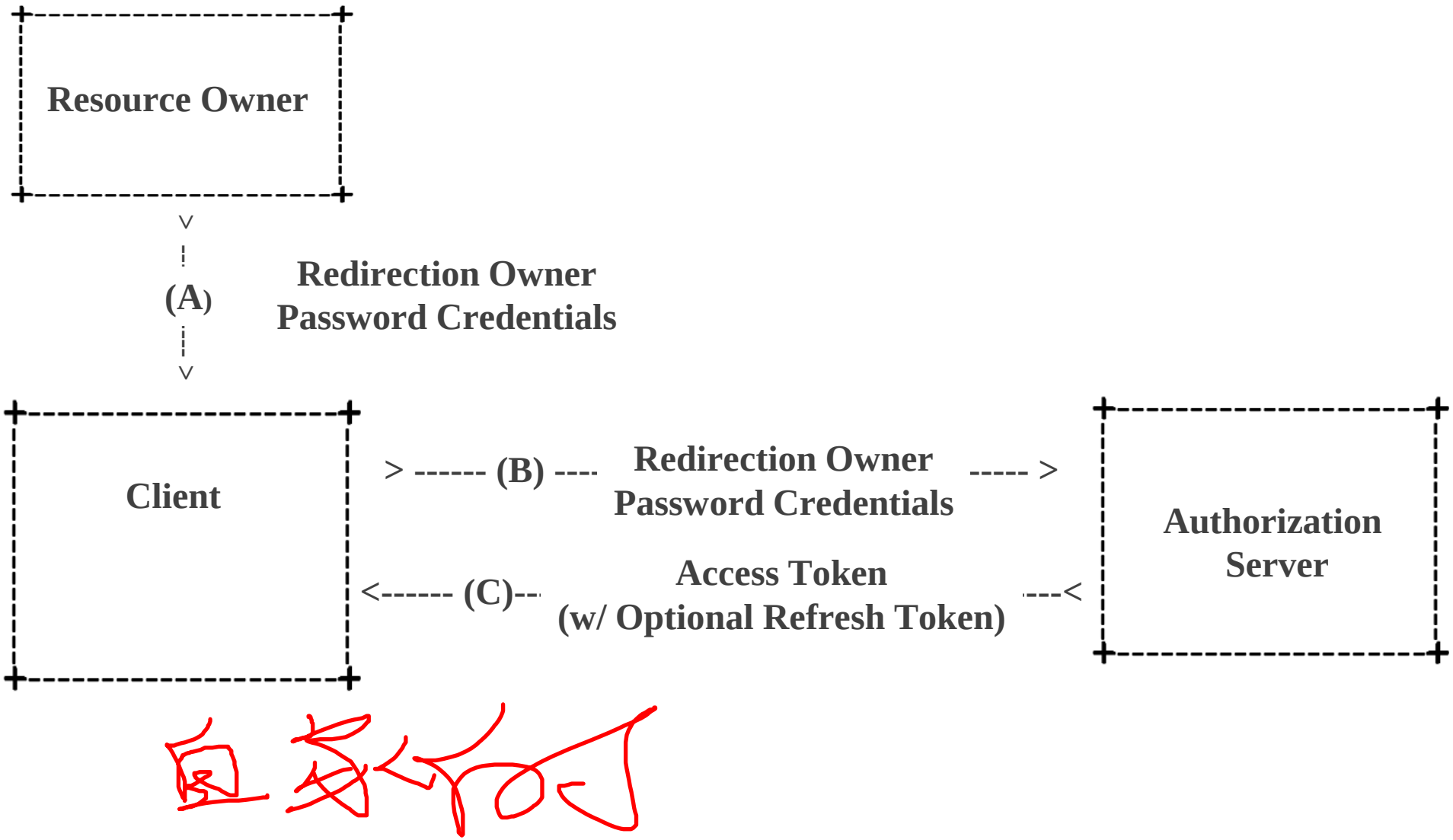
**阮一峰：理解OAuth 2.0**
http://www.ruanyifeng.com/blog/2014/05/oauth_2_0.html



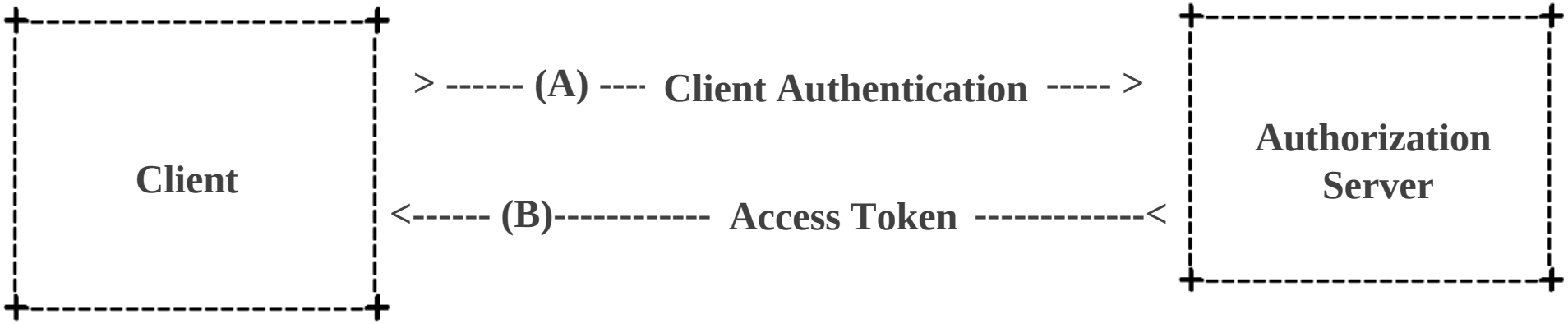
# 简化模式


Resource Owner
(B)
User-Agent
(A)
(G)
Client
(A) Client Identifier &Redirection URI
(B) User authenticates
(C) Redirection URI with Access Token in Fragment
(D) Redirection URI without Fragment
(E) Script
Authorization Server
Web-Hosted Client Resource

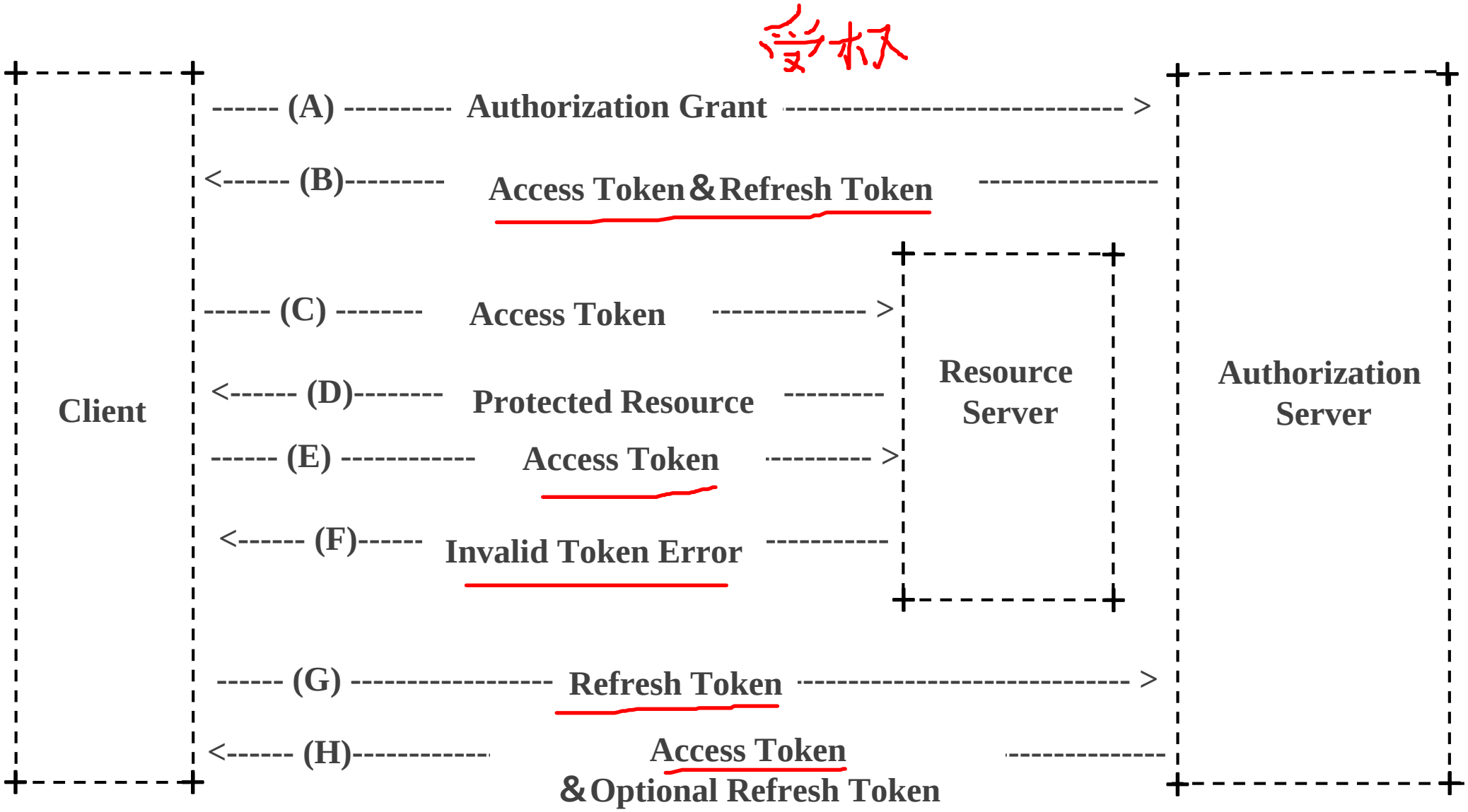
# 密码模式


Resource Owner
(A)
Redirection Owner
Password Credentials
Client
> ------ (B) ----
Redirection Owner
Password Credentials
----- >
Authorization
Server
<------ (C)--
Access Token
(w/ Optional Refresh Token)
---<

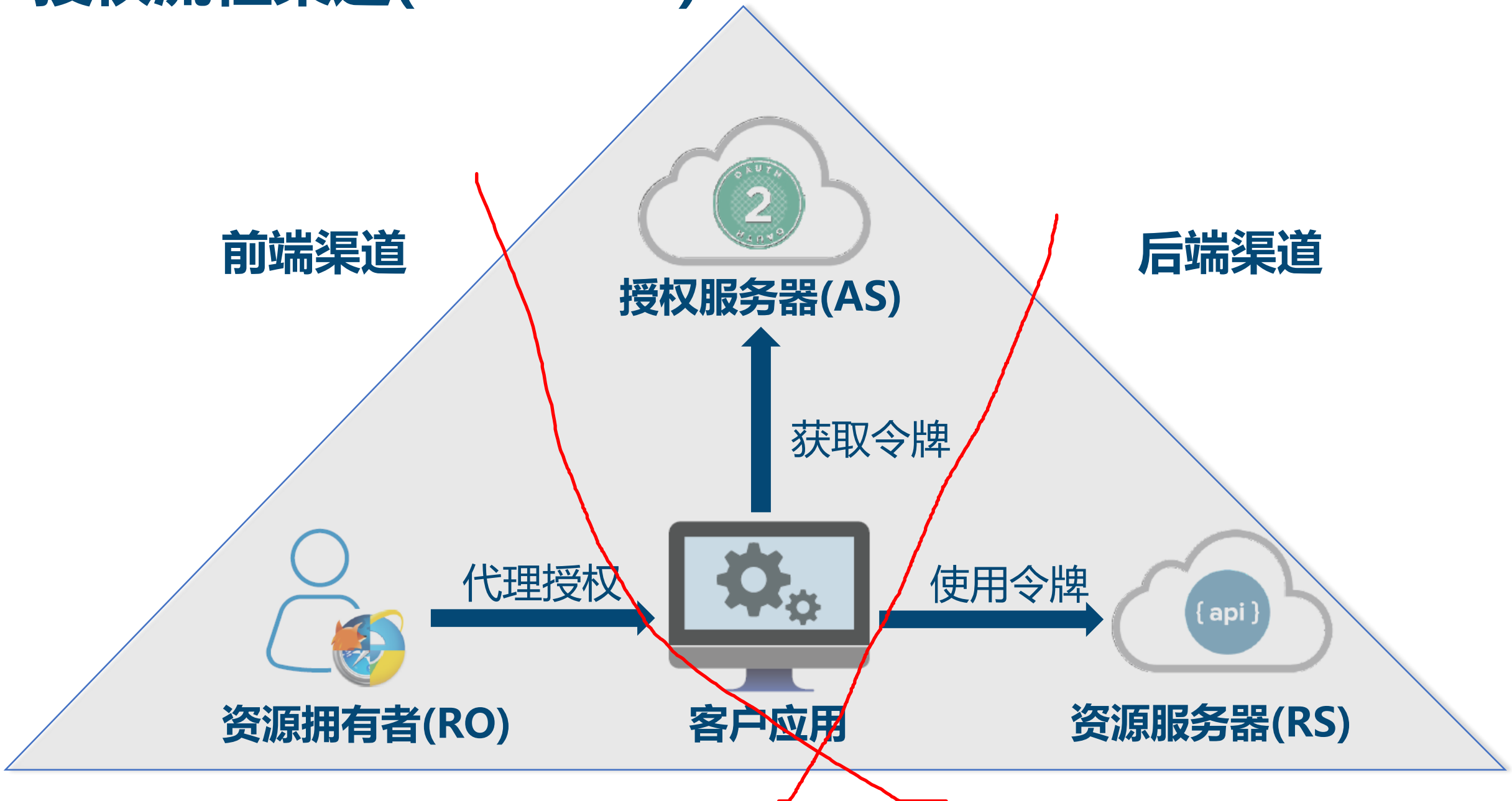
# 客户端模式


Client
> ------ (A) ---- Client Authentication ----- >
<------ (B)---------- Access Token ----------<
Authorization Server

# 刷新令牌


Client
Resource Server
Authorization Server
(A) Authorization Grant
(B) Access Token&Refresh Token
(C) Access Token
(D) Protected Resource
(E) Access Token
(F) Invalid Token Error
(G) Refresh Token
(H) Access Token &Optional Refresh Token

# 授权流程渠道(channels)



# 授权类型选择~客户应用类型

**单页应用SPA**

Angular, React, etc

**原生App**

iOS, Android

**Web服务器端应用**

.NET, Java, etc.


**服务/API**

机器对机器

**公开**

（客户标识）

**私密**

（客户认证）

# 四种OAuth 2.0授权类型(Flows)

## 01 授权码Authorization Code

- 通过前端渠道**客户**获取授权码
- 通过后端渠道，**客户**使用authorization code去交换access Token和可选的refresh token
- 假定**资源拥有者**和**客户**在不同的设备上
- 最安全的流程，因为令牌不会传递经过user-agent

## 02 简化Implicit

- 适用于**公开的**浏览器单页应用
- Access Token直接从授权服务器返回（只有前端渠道）
- 不支持refresh tokens
- 假定资源所有者和公开客户应用在同一个设备上
- 最容易受安全攻击

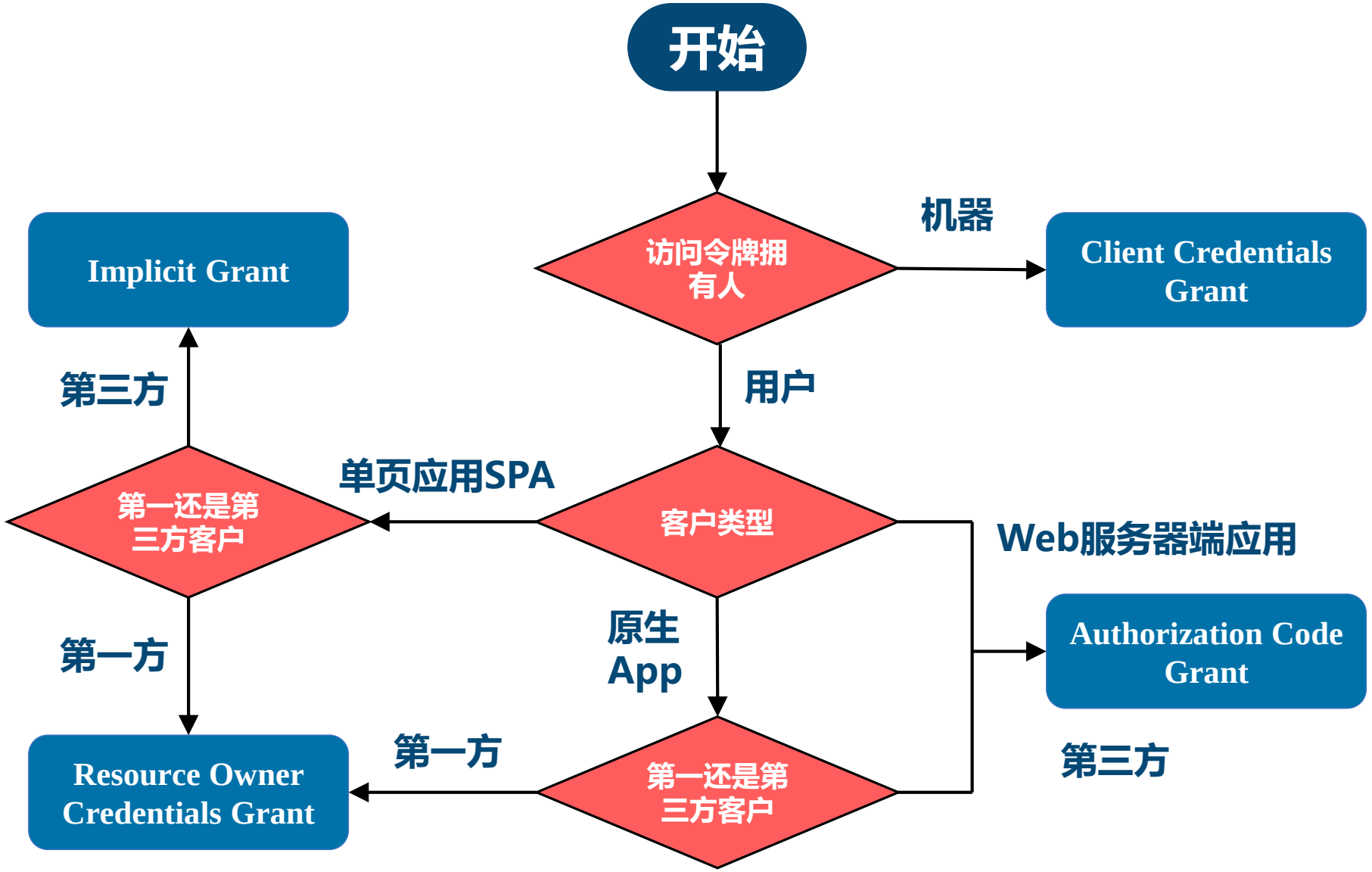
## 03 用户名密码Resource Owner Credentials

- 使用用户名密码登录的应用，例如桌面App
- 使用用户名/密码作为授权方式从授权服务器上获取access token
- 一般不支持refresh tokens
- 假定资源拥有者和公开客户在相同设备上

## 04 客户端凭证Client Credentials

- 适用于服务器间通信场景，**机密客户**代表它自己或者一个用户
- 只有后端渠道，使用**客户**凭证获取一个access token
- 因为**客户**凭证可以使用对称或者非对称加密，该方式支持共享密码或者证书

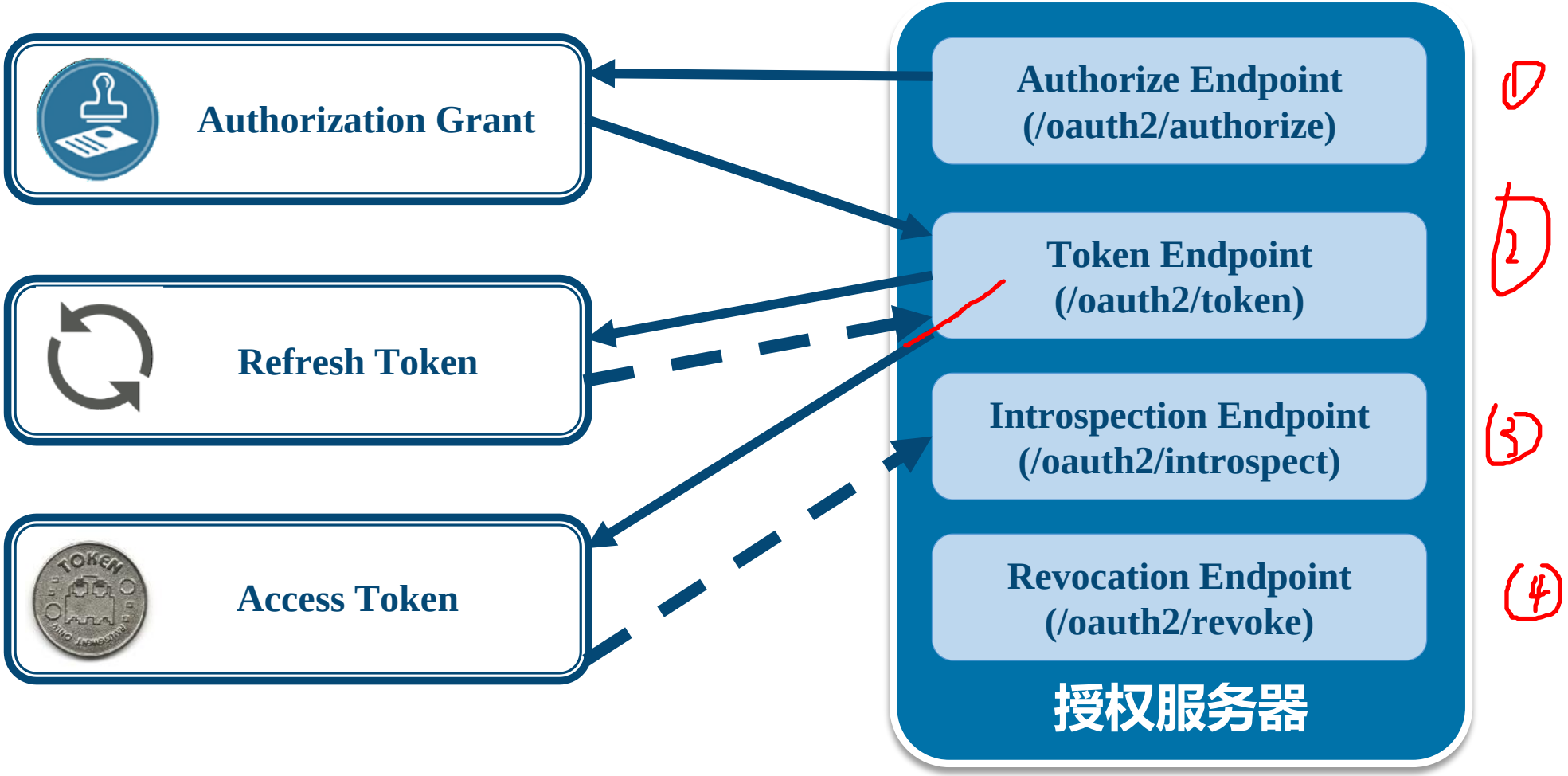
# 授权类型选择~流程


开始
访问令牌拥有人
机器
Client Credentials Grant
用户
客户类型
单页应用SPA
第一还是第三方客户
第三方
Implicit Grant
第一方
Resource Owner Credentials Grant
原生App
第一还是第三方客户
第一方
Web服务器端应用
Authorization Code Grant
第三方

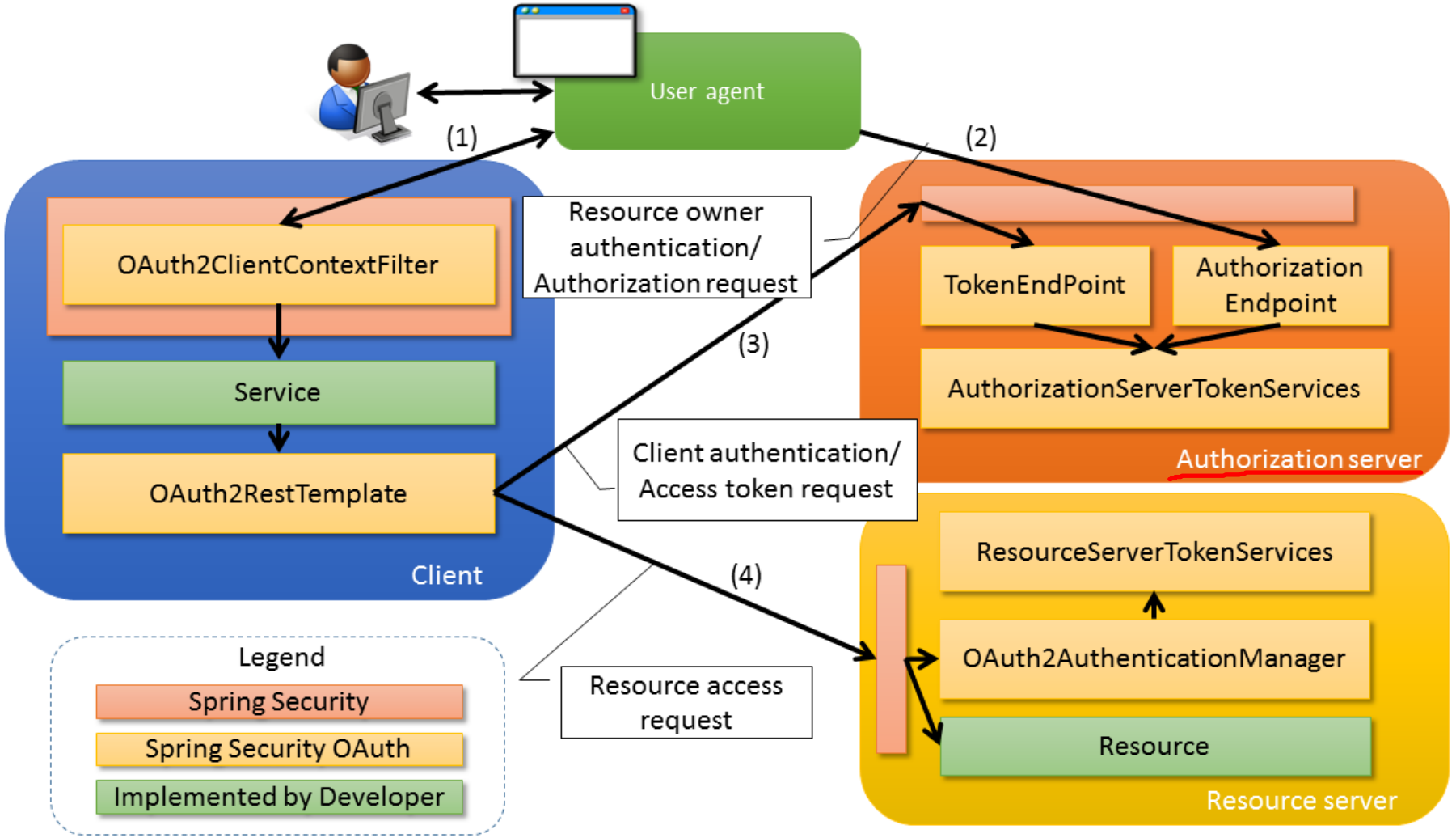
# 第5部分

# OAuth2授权服务器和资源服务器案例实操

# 授权服务器


Authorization Grant
Refresh Token
Access Token
Authorize Endpoint
(/oauth2/authorize)
Token Endpoint
(/oauth2/token)
Introspection Endpoint
(/oauth2/introspect)
Revocation Endpoint
(/oauth2/revoke)
授权服务器
1
2
3
4

# Spring Security OAuth2架构




http://terasolunaorg.github.io/guideline/5.3.0.RELEASE/en/Security/OAuth.html

# Lab01课后扩展

01 支持刷新令牌Refresh Token

02 使用关系数据库存储令牌和客户信息

03 使用缓存Cache存储令牌提升性能

04 授权服务器和资源服务器拆分

05 Revoke端点

06 Introspection端点

第 6 部分

# OAuth2客户端案例实操（lab）

# Lab02课后扩展

01 使用Spring Security OAuth2 客户端支持授权码模式

02 客户端支持简化/密码/客户端模式

03 客户端支持refresh token

# 第 7 部分

# JWT令牌原理

# 访问令牌的类型

## By reference token (透明令牌)

随机生成的字符串标识符，无法简单猜测授权服务器如何颁发和存储

资源服务器必须通过后端渠道发送回OAuth2授权服务器的令牌检查端点，才能校验令牌是否有效，并获取claims/scopes等额外信息

VS

## By value token (自包含令牌)

授权服务器颁发的令牌，包含关于用户或者客户的元数据和声明(claims)

通过检查签名，期望的颁发者(issuer)，期望的接收人aud(audience)，或者scope，资源服务器可以在本地校验令牌

通常实现为签名的JSON Web Tokens(JWT)

# JSON Web Token(JWT)

**Header**

**eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwczovL215LnNwcmluZzJnby5jb20iLCJpYXQiOjE0MzUxNzk2MDMsImV4cCI6MTQzNTE4MTQyMSwiYXVkIjoid3d3LnNwcmluZzJnby5jb20iLCJzdWIiOiJ3aWxsaWFtQGdtYWlsLmNvbSIsIlJvbGUiOlsiYXBwcm92ZXIiLCJ2aWV3ZXIiXX0.c8d5qTxlUshTN4gM9tsYi9SFrBlWSEHSRSdmnHeq_g**

**Signature**

**base64url(Header)+ “.” +base64url(Claims)+ “.” +base64url(Signature)**

**Claims**

# 令牌签发人和目标接收人

# 令牌签发人和目标接收人

# 第 8 部分

# JWT案例实操（lab）

# Lab03课后扩展

01 在JWT令牌中增加定制claims

JWT令牌的非对称签名和校验 02

03 使用JWE加密/解密JWT令牌

# 第 9 部分

# Android无线应用接入OAuth2案例实操(lab)

# Lab04课后扩展

01

支持用户名密码模式

02

使用PKCE(RFC7636)增强无线客户使用授权码模式的安全性

# 第10部分

# Angularjs单页应用接入OAuth2案例实操(lab)

# Lab05课后扩展

# 第11部分

# Github社交登录案例实操(lab)

# Lab06课后练习

01

新浪微博/微信/QQ

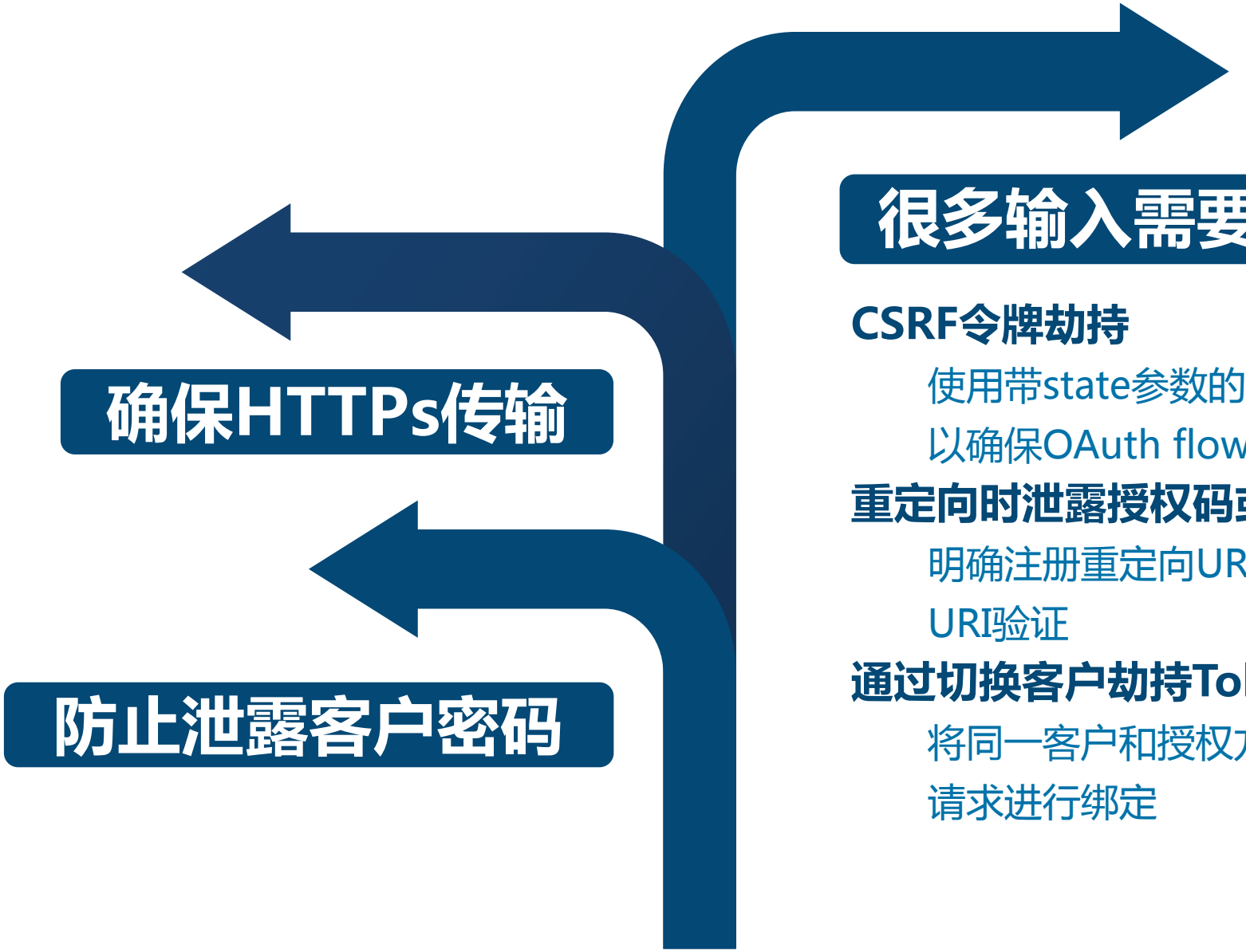
02

Spring Social

# 第12部分

# OAuth2安全风险和案例实操(lab)

# 常见OAuth 2.0安全问题



**很多输入需要验证**

**CSRF令牌劫持**

使用带state参数的CSRF token以确保OAuth flow的一致性

**重定向时泄露授权码或者令牌**

明确注册重定向URIs，并确保URI验证

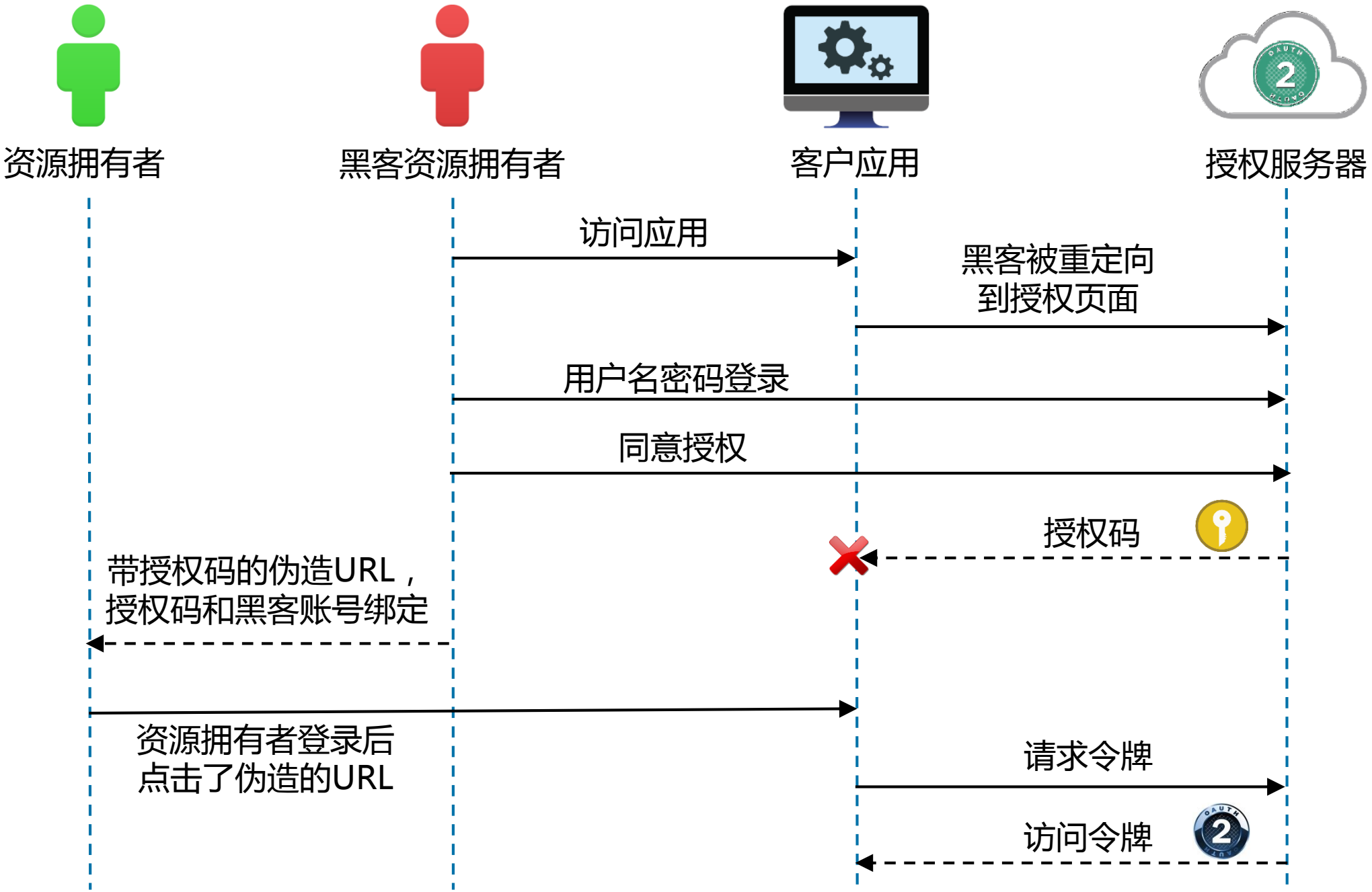
**通过切换客户劫持Token**

将同一客户和授权方式/token请求进行绑定

**确保HTTPs传输**

**防止泄露客户密码**

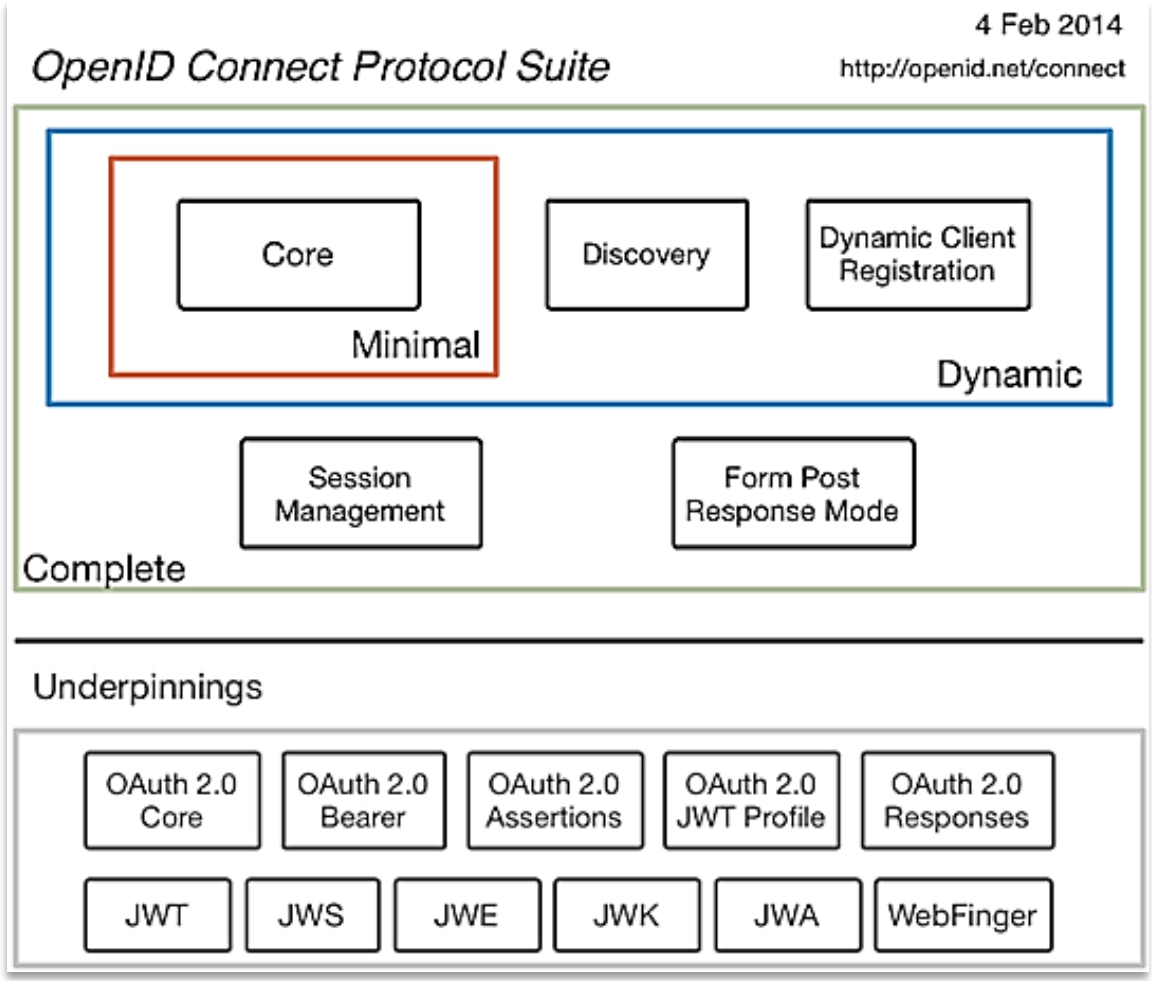
# CSRF


资源拥有者
黑客资源拥有者
客户应用
授权服务器
访问应用
黑客被重定向
到授权页面
用户名密码登录
同意授权
授权码
带授权码的伪造URL，
授权码和黑客账号绑定
资源拥有者登录后
点击了伪造的URL
请求令牌
访问令牌


# 第13部分

# OpenId Connect简介

# OpenID Connect




**基于OAuth2之上构建的简单身份认证层**

**支持新的签名的id_token**

**UserInfo端点**

**提供一组标识身份的标准的scopes和claims**

- profile
- email
- address
- Phone

**支持客户自注册，发现，会话管理(SSO)等额外功能**

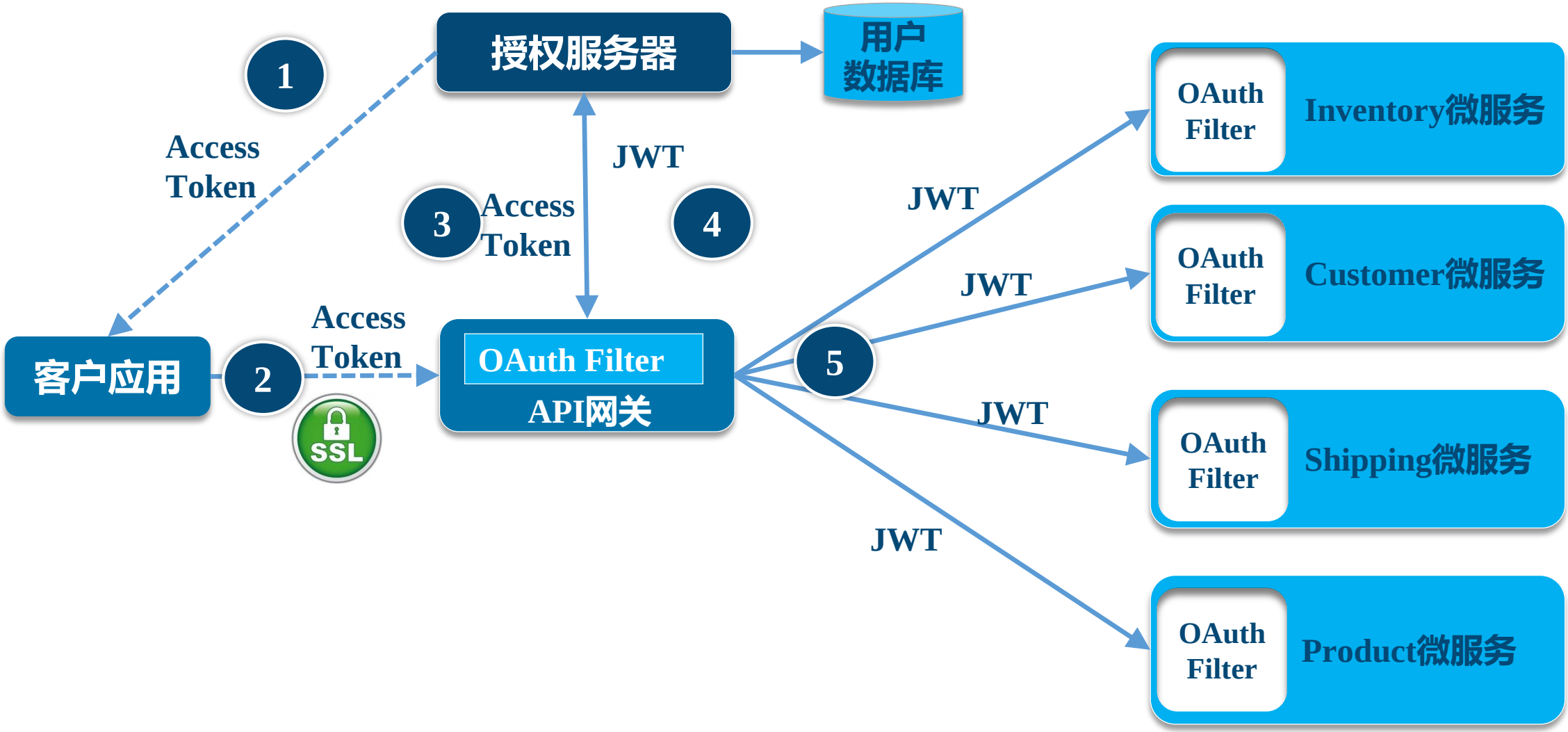
**增加了2个新的flow**

**(Identity, Authentication) + OAuth 2.0 = OpenID Connect**

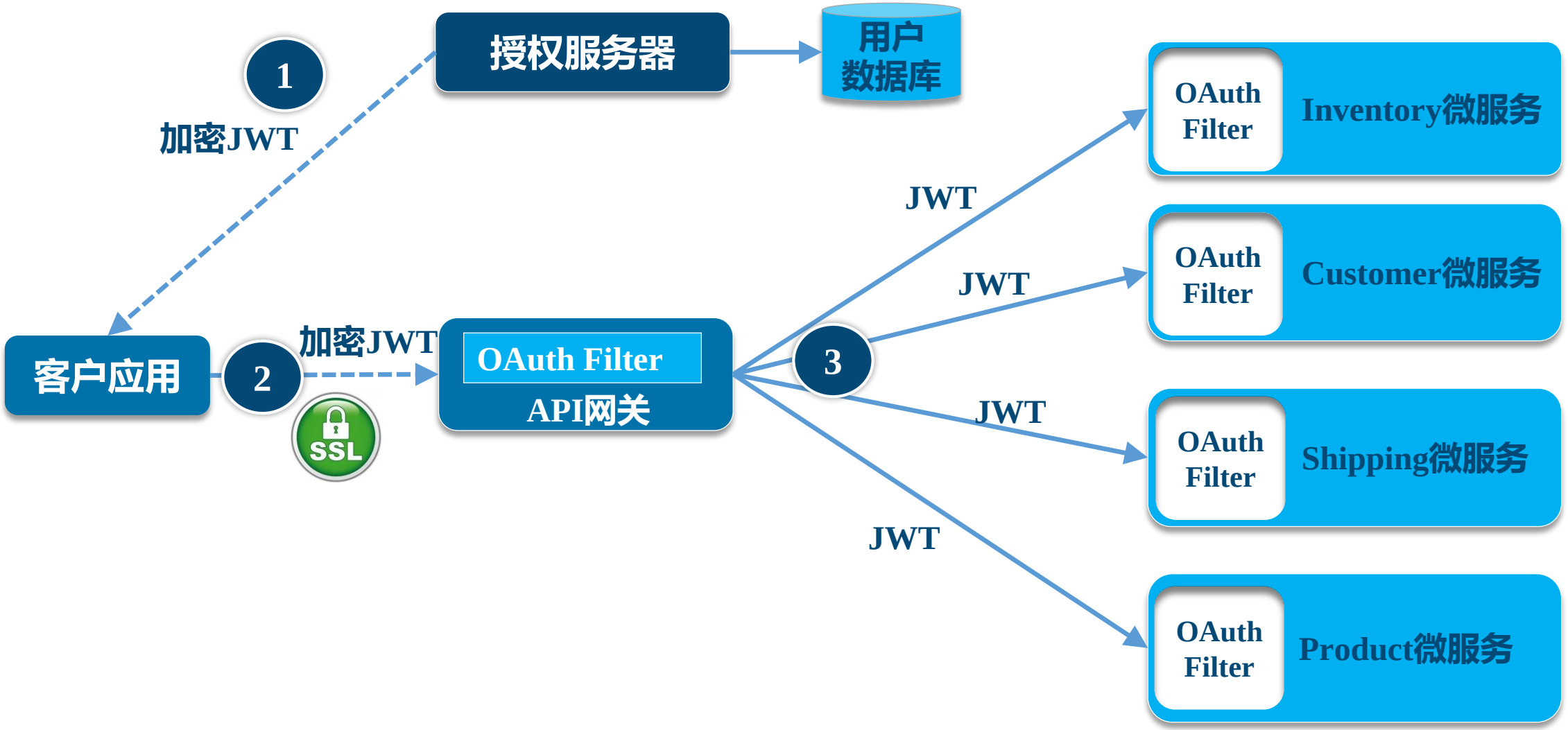
# 第 14 部分

# 下一代微服务安全架构

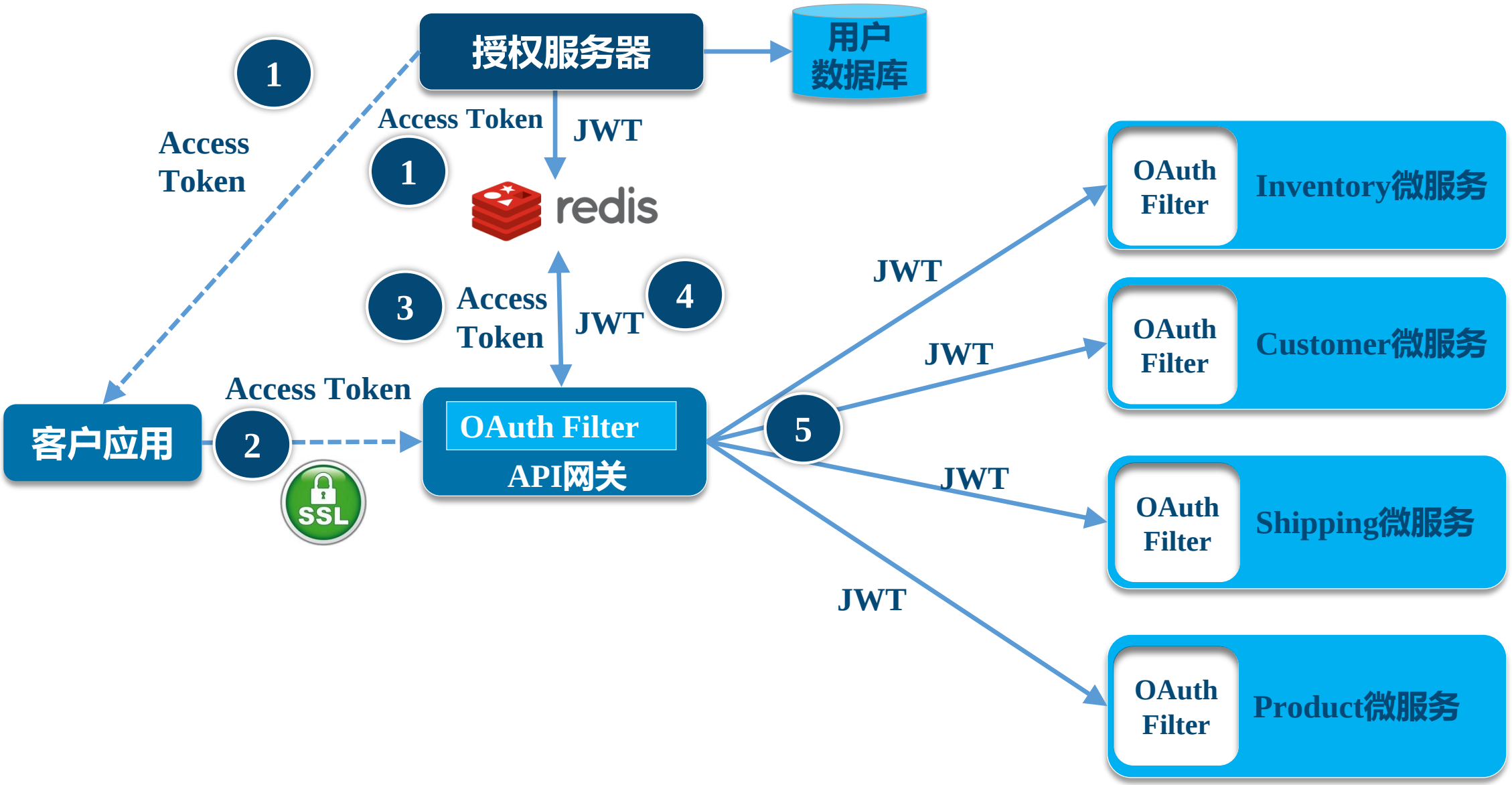
# 方案一


授权服务器
用户
数据库
1
Access
Token
3
Access
Token
JWT
4
客户应用
2
Access
Token
SSL
OAuth Filter
API网关
5
JWT
JWT
JWT
JWT
OAuth
Filter
Inventory微服务
OAuth
Filter
Customer微服务
OAuth
Filter
Shipping微服务
OAuth
Filter
Product微服务

# 方案二


授权服务器
用户
数据库
1
加密JWT
客户应用
2
加密JWT
SSL
OAuth Filter
API网关
3
JWT
JWT
JWT
JWT
OAuth
Filter
Inventory微服务
OAuth
Filter
Customer微服务
OAuth
Filter
Shipping微服务
OAuth
Filter
Product微服务

# 方案三


授权服务器
用户
数据库
1
Access
Token
Access Token
JWT
1
redis
3
Access
Token
JWT
4
Access Token
客户应用
2
SSL
OAuth Filter
API网关
5
JWT
JWT
JWT
JWT
OAuth
Filter
Inventory微服务
OAuth
Filter
Customer微服务
OAuth
Filter
Shipping微服务
OAuth
Filter
Product微服务

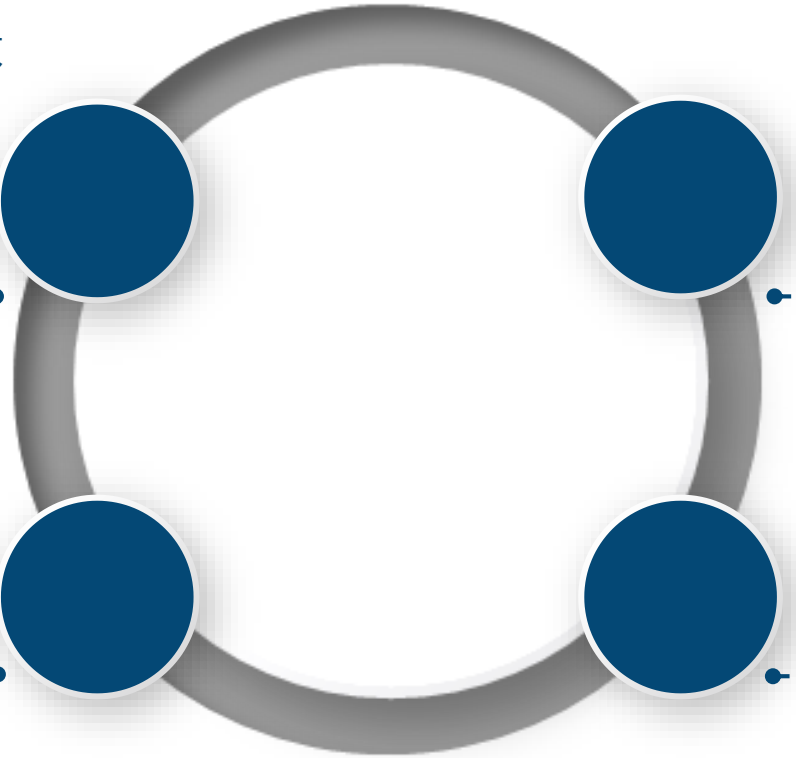
# 生产级部署实践

## 业务指标监控

- 登录次数
- 授权次数
- 颁发访问令牌/刷新令牌个数
- 活跃令牌数
- 令牌校验次数
- 令牌吊销次数
- 注册client数

## 接口调用性能指标

- authorize
- token
- introspect
- revoke



## 缓存Caching

## HA和水平扩容

# 第15部分

# 参考资源和后续课程预览

# OAuth2/OIDC开源产品


## Redhat Keycloak（Java）

- http://www.keycloak.org


## Apereo CAS（Java）

- https://www.apereo.org/projects/cas


## IdentityServer（C#）

- https://identityserver.io/


## OpenId-Connect-Java-Spring-Server

- https://github.com/mitreid-connect/OpenID-Connect-Java-Spring-Server

# Spring Security OAuth2


## Developer Guide

- https://projects.spring.io/spring-security-oauth/docs/oauth2.html

## OAuth 2.0 Cookbook

- https://www.packtpub.com/virtualization-and-cloud/oauth-20-cookbook
- https://github.com/PacktPublishing/OAuth-2.0-Cookbook

# OAuth和OIDC库


01 Google OAuth Client Library

02 ScribeJava

03 Spring Security OAuth

04 Nimbus OAuth SDK

05 各种语言的服务器和客户端库
https://oauth.net/code/

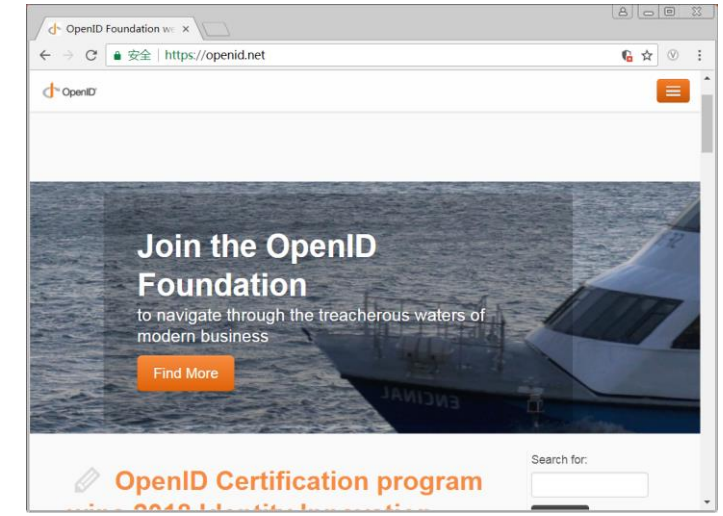
# OAuth2/OIDC SaaS服务


https://www.okta.com/

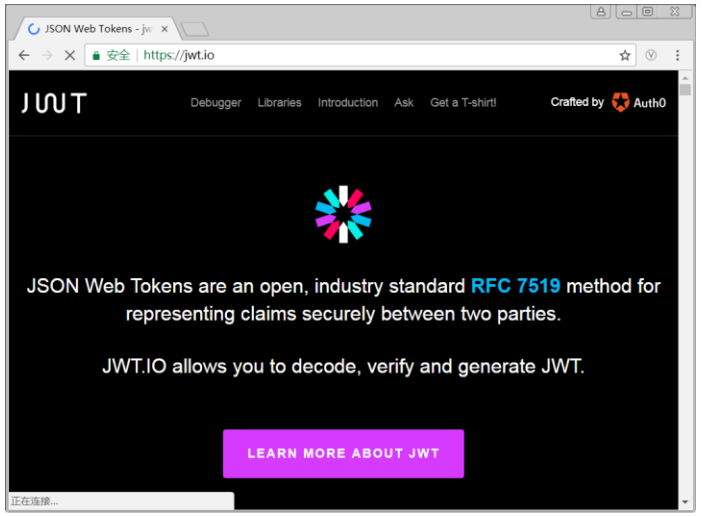
https://auth0.com/

# 规范参考

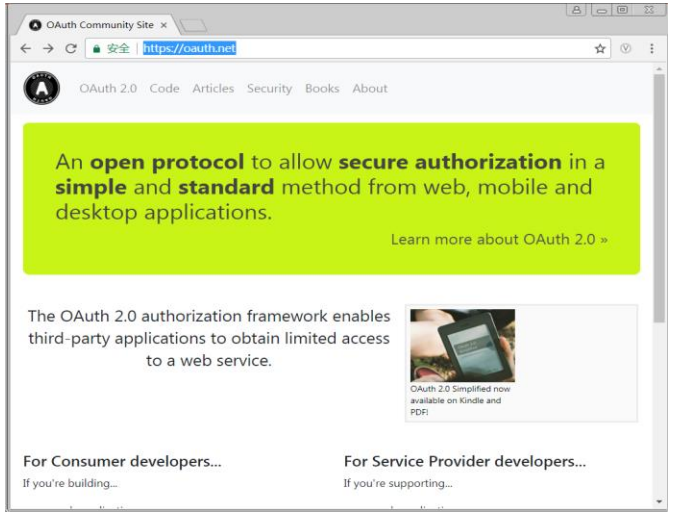
https://openid.net




https://jwt.io




https://oauth.net



# 参考文章

http://www.ruanyifeng.com/blog/2014/05/oauth20.html


https://medium.com/@darutk/the-simplest-guide-to-oauth-2-0-8c71bd9a15bb

# 参考开源代码

https://github.com/newnil/oauth2-family-barrel

OAuth2全家桶项目

https://github.com/monkeyk/oauth2-shiro

Apache Oltu+Shiro实现OAuth2服务器(李胜钊)

http://www.baeldung.com/spring-security-oauth-jwt
https://github.com/Baeldung/spring-security-oauth

Using JWT with Spring Security OAuth

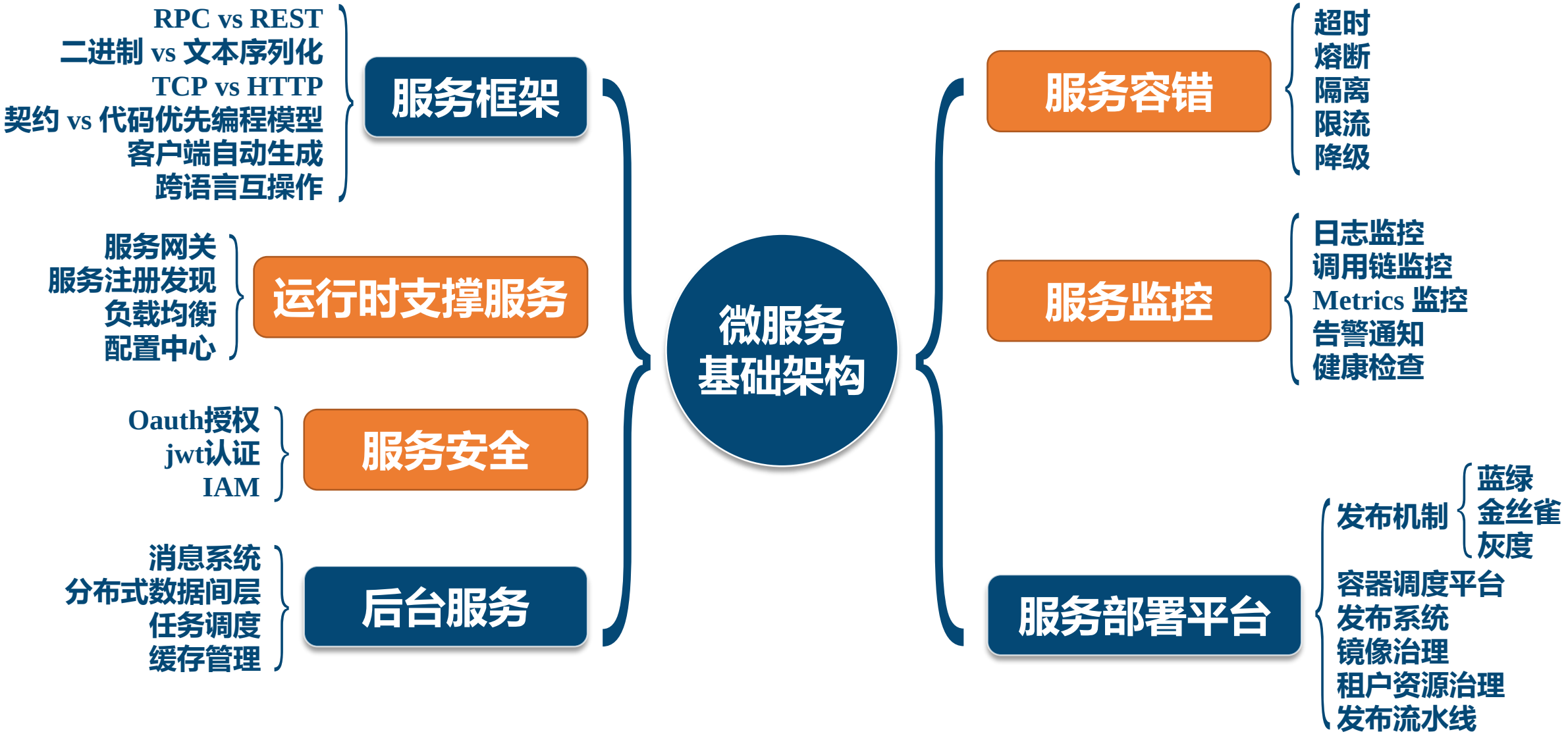
# 课程代码

https://github.com/spring2go/oauth2lab

- Lab01~OAuth2服务器端案例实操
- Lab02~OAuth2客户端案例实操
- Lab03~JWT案例实操
- Lab04~Android客户端案例实操
- Lab05~Angularjs单页应用案例实操
- Lab06~Github社交登录案例实操
- Lab07~OAuth2安全风险案例实操

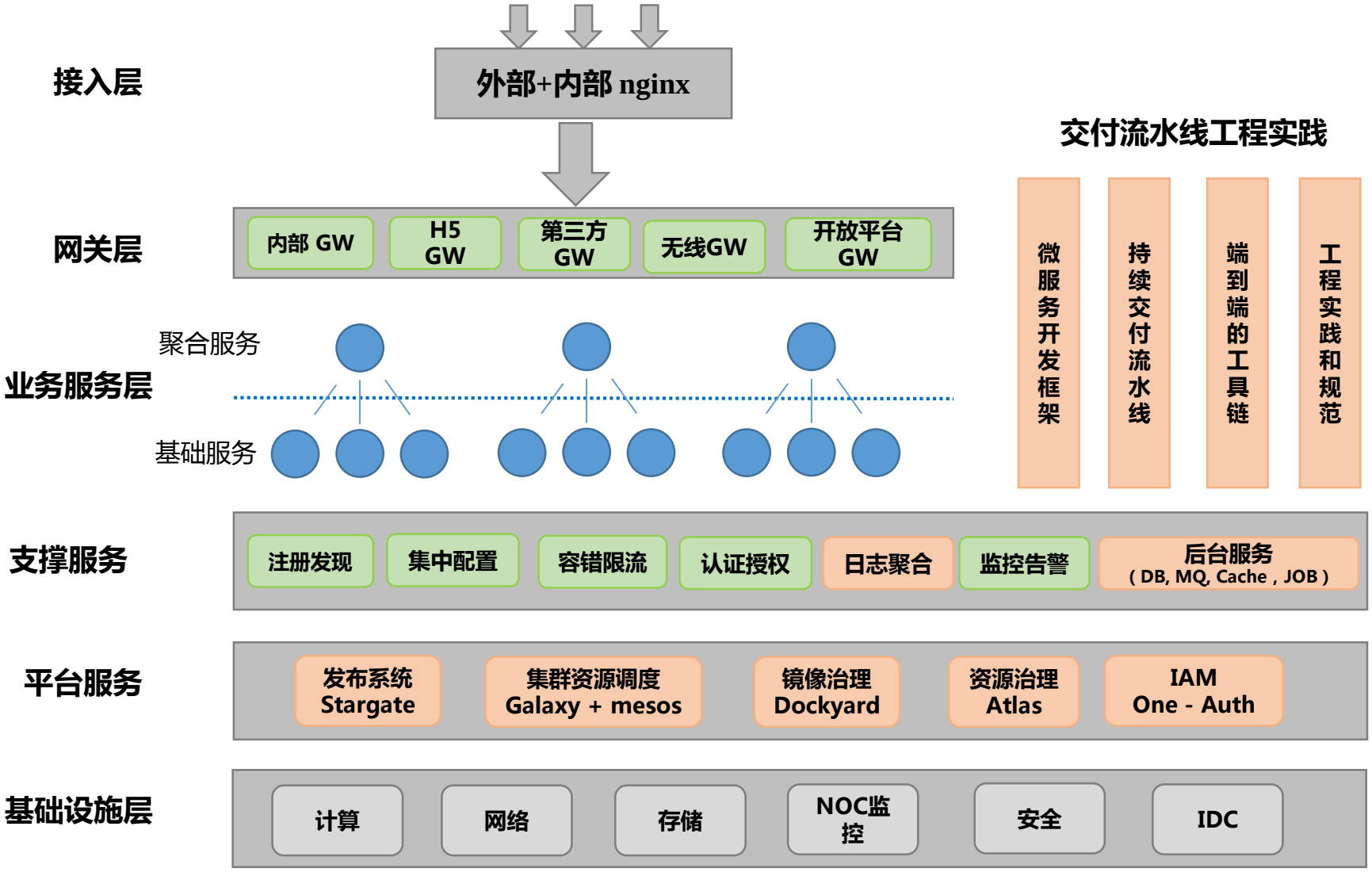
学习参考
非生产级
！

# 后续课程预览~2018课程模块

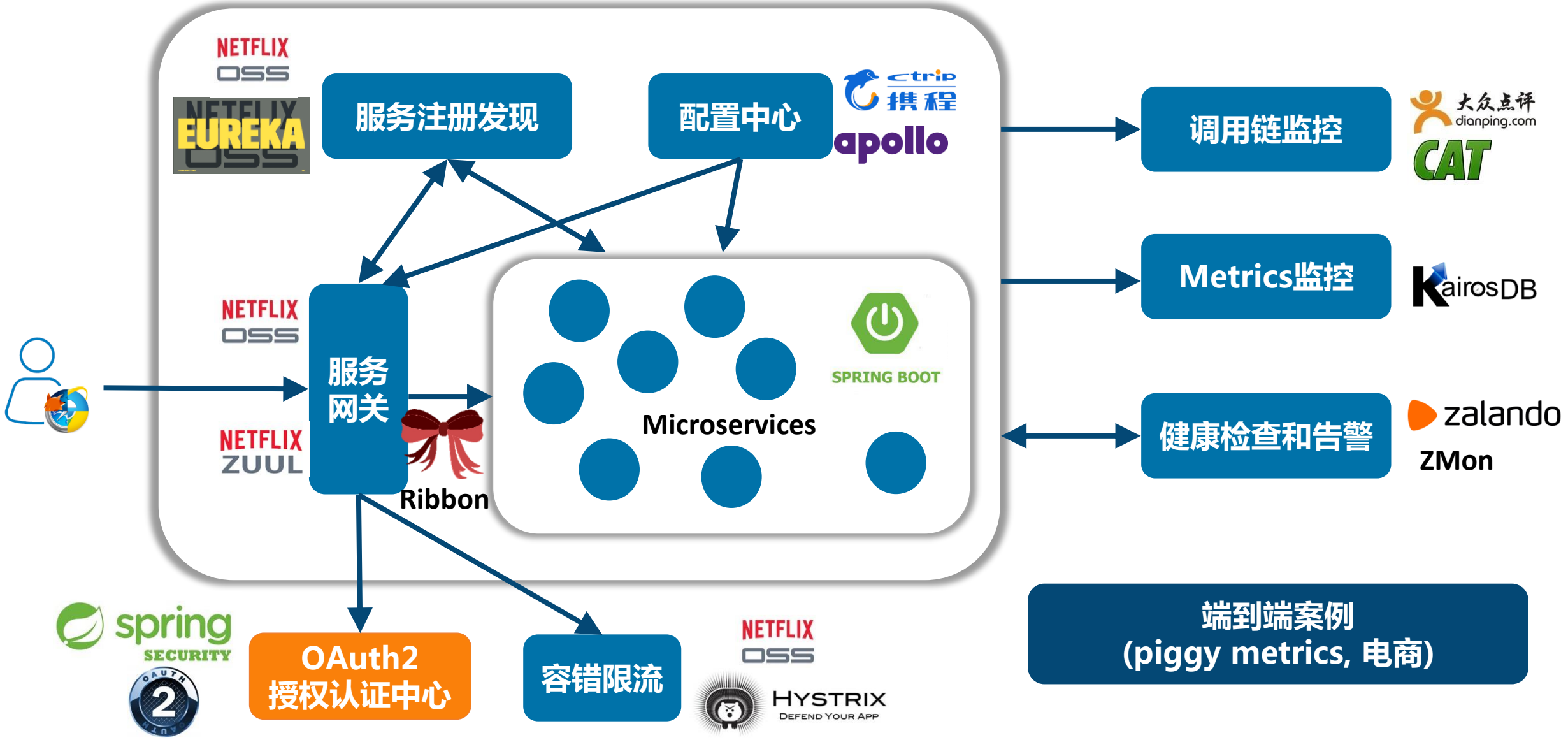


# 后续课程预览~技术体系

微服务架构总体技术体系



# 架构和技术栈预览


NETFLIX OSS
EUREKA
服务注册发现
配置中心
ctrip 携程
apollo
调用链监控
大众点评 dianping.com
CAT
NETFLIX OSS
服务网关
NETFLIX ZUUL
Ribbon
Microservices
SPRING BOOT
Metrics监控
KairosDB
健康检查和告警
zalando
ZMon
spring SECURITY
OAuth2 授权认证中心
容错限流
NETFLIX OSS
HYSTRIX DEFEND YOUR APP
端到端案例
(piggy metrics, 电商)