Keystone 代码结构分析

目录

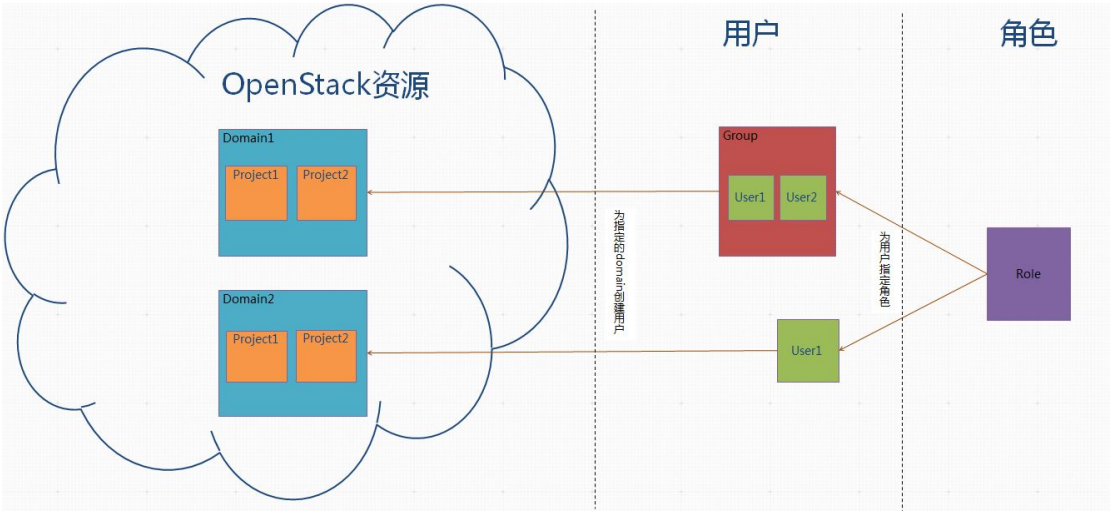
相关概念

名词介绍

Key	说明
Domain	域（v3）是 OpenStack 委托用来资源管理的容器，是项目、组和用户的更高一层的容器，每个域都是一个单独的命名空间，提供的默认域名为 ‘default’
Project	项目（v2.0 称为租户）是资源的容器，OpenStack 中的所有资源必须属于一个指定项目，一个项目必须属于一个指定的域，并且在域中是唯一的
User	用户是使用 OpenStack 资源的人、系统或服务，用户必须属于一个指定的域，并且在域中是唯一的
Role	角色标识了在域或项目中的权限，可以分配给用户和组，角色在 OpenStack 中是全局唯一的
Token	令牌是用来验证用户身份的，用户登录后会使用令牌执行相应的资源请求
Service	服务是指一个 OpenStack 服务，比如：nova、cinder、neutron 等。
Endpoint	端口是一个网络可访问的服务地址，通过它可以访问一个服务，通常是个 URL 地址。比如，当 Nova 需要访问 Glance 服务去获取 image 时，Nova 通过访问 Keystone 拿到 Glance 的 endpoint，然后通过访问该 endpoint 去获取 Glance 服务
Policy	策略是用来鉴别用户对某个服务是否有访问权限的，对于 keystone 来说 policy 就是一个 json 文件

关系及存储

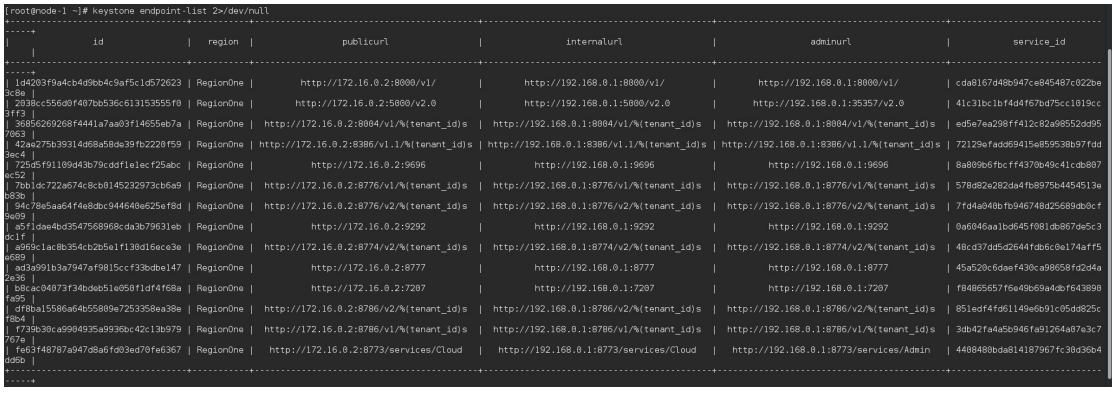
domain、project、user、group、role 之间的关系：



用户或组可以被分配到指定的项目或域中并被赋予角色，用户或组、项目或域、角色以三级关联的关系存储在 assignment 表中。

service、endpoint：

```
[root@node-1 ~]# keystone service-list 2>/dev/null
+----------------------------------+------------+-----------------+----------------------------------+
|                id                |    name    |       type      |           description            |
+----------------------------------+------------+-----------------+----------------------------------+
| 45a520c6daef430ca98658fd2d4a2e36 | ceilometer |     metering    |    Openstack Metering Service    |
| f84865657f6e49b69a4dbf643890fa95 |   chakra   |     account     |    Openstack Account Service     |
| 578d82e282da4fb8975b4454513eb83b |   cinder   |      volume     |         Cinder Service           |
| 7fd4a040bfb946748d25689db0cf9e09 |  cinderv2  |     volumev2    |       Cinder Service v2          |
| 0a6046aa1bd645f081db867de5c3dc1f |   glance   |      image      |     Openstack Image Service      |
| ed5e7ea298ff412c82a98552dd957063 |    heat    |  orchestration  | Openstack Orchestration Service  |
| cda8167d48b947ce845487c022be3c8e |  heat-cfn  |  cloudformation | Openstack Cloudformation Service |
| 41c31bc1bf4d4f67bd75cc1019cc3ff3 |  keystone  |     identity    |    OpenStack Identity Service    |
| 3db42fa4a5b946fa91264a07e3c7767e |   manila   |      share      |   OpenStack Shared Filesystems   |
| 851edf4fd61149e6b91c05dd825cf8b4 |  manilav2  |     sharev2     | OpenStack Shared Filesystems V2  |
| 8a809b6fbcff4370b49c41cdb807ec52 |  neutron   |     network     |    Neutron Networking Service    |
| 48cd37dd5d2644fdb6c0e174aff5e689 |    nova    |     compute     |    Openstack Compute Service     |
| 4408480bda814187967fc30d36b4dd6b |  nova_ec2  |       ec2       |          EC2 Service             |
| 72129efadd69415e859538b97fdd3ec4 |   sahara   | data_processing |    OpenStack Data Processing     |
+----------------------------------+------------+-----------------+----------------------------------+
```



endpoint 以 service_id 关联 service，并把 publicurl、internalurl、adminurl 存储在 endpoint 表中。

代码目录结构


es_keystone [keystone juno]
bin
doc
etc
相关配置文件
examples
httpd
keystone
assignment
提供角色和角色分配
auth
认证
catalog
提供端点发现和端点注册
common
contrib
credential
提供证书
hacking
identity
用户、组身份验证凭据认证和数据
backends
提供服务后端
id_generators
mapping_backends
__init__.py
controllers.py
控制器
core.py
管理器和驱动
generator.py
routers.py
url映射
locale
middleware
中间件
models
openstack
policy
提供基于规则的授权引擎和相关规则管理界面
tests
token
令牌服务验证和管理
trust
__init__.py
backends.py
clean.py
cli.py
config.py
controllers.py
exception.py
i18n.py
notifications.py
routers.py
service.py
rally-scenarios
tools
babel.cfg
CONTRIBUTING.rst
HACKING.rst
LICENSE
MANIFEST.in
openstack-common.conf
README.rst
requirements.txt
依赖
requirements-py3.txt
run_tests.sh
setup.cfg
setup.py
test-requirements.txt
test-requirements-py3.txt
tox.ini

代码流程

1、 keystone-all.py 启动 keystone 服务

首先将 keystone 源码目录下的模块路径进行设置，将目录配置在系统目录下：

```
possible_topdir = os.path.normpath(os.path.join(os.path.abspath(__file__),
                                   os.pardir,
                                   os.pardir))
if os.path.exists(os.path.join(possible_topdir,
                               'keystone',
                               '__init__.py')):
    sys.path.insert(0, possible_topdir)
```

再引入目录下的其他模块：

```
from paste import deploy
import pbr.version
from keystone.openstack.common import gettextutils
gettextutils.install('keystone', lazy=True)
```

引入 paste.deploy 模块——是一个用于发现和配置 WSGI 应用或者服务的系统

pbr 模块是用于安装时配置的一个模块——是 PBR 是一个库 ,用于注入一些有用的和敏感的基本配置信息到 setuptools.setuptools 是一个工具用于下载，编译，安装， 升级，卸载 python 的包文件。

gexttextutil 是一个运行后加载的模块，用于识别用户的提交的语言，帮助返回错误信息时使用用户语言。

通过 eventlet 模块启动配置服务器，完成 eventlet 环境的配置，并启动服务器。

2、 WSGI 接口访问

keystone-paste.ini 的流程如下：

```
[pipeline:public_api]
pipeline = access_log sizelimit url_normalize token_auth admin_token_auth xml_body json_body
ec2_extension user_crud_extension public_service
```

pastedeploy 过滤流程如下：

```
[filter:access_log]
paste.filter_factory = keystone.contrib.access:AccessLogMiddleware.factory
[filter:sizelimit]
paste.filter_factory = keystone.middleware:RequestBodySizeLimiter.factory
[filter:url_normalize]
paste.filter_factory = keystone.middleware:NormalizingFilter.factory
[filter:token_auth]
paste.filter_factory = keystone.middleware:TokenAuthMiddleware.factory
[filter:admin_token_auth]
paste.filter_factory = keystone.middleware:AdminTokenAuthMiddleware.factory
[filter:xml_body]
paste.filter_factory = keystone.middleware:XmlBodyMiddleware.factory
[filter:json_body]
paste.filter_factory = keystone.middleware:JsonBodyMiddleware.factory
[filter:ec2_extension]
paste.filter_factory = keystone.contrib.ec2:Ec2Extension.factory
[filter:user_crud_extension]
paste.filter_factory = keystone.contrib.user_crud:CrudExtension.factory
[app:public_service]
paste.app_factory = keystone.service:public_app_factory
```

pipeline 中的 app_factory 会被依次执行，ec2_extension，s3_extension，这两个过滤 filter 实现的是 Routes 的机制，完成路径的匹配，如果在路径中没有找到，再调用 self.application。

完成 WSGI 的接口后需要调用 Controller 的函数来实现功能，完成并返回结果， Controller 可配置多个 Manager，Manager 可选择 Driver。Driver 在配置文件 keystone.conf 下面可以进行配置，流程是 Controller->Manager->Driver。配置文件的读取功能是由 oslo.config 文件来完成的。

provider 是装饰的这个类注册到 REGISTER[]下面 ,requires 是将 REGISTER 里面的类取出来，注入到装饰的类下面。把 Manager 进行注册和提取，这样就可以方便我们的 Controller 调用了。在 service.py 的全局变量中注册：

```
_IDENTITY_API = identity.Manager()
DRIVERS = dict(
    assignment_api=assignment.Manager(),
    catalog_api=catalog.Manager(),
    credentials_api=credential.Manager(),
    endpoint_filter_api=endpoint_filter.Manager(),
    identity_api=_IDENTITY_API,
    policy_api=policy.Manager(),
    token_api=token.Manager(),
    trust_api=trust.Manager(),
    token_provider_api=token.provider.Manager())
dependency.resolve_future_dependencies()
```

Controller 提取 manager：

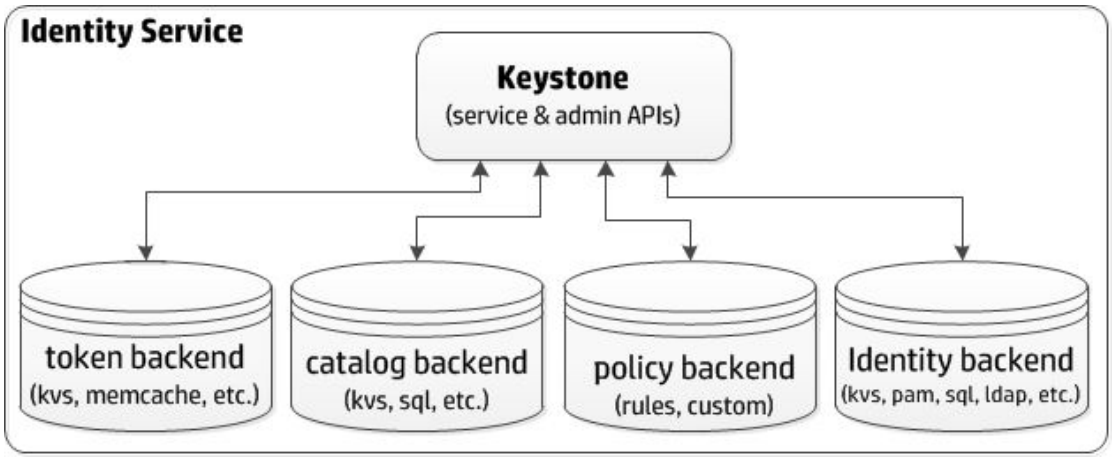
```
@dependency.requires('identity_api', 'policy_api', 'token_api',
                     'trust_api', 'catalog_api', 'credential_api',
                     'assignment_api')
class V2Controller(wsgi.Application):
    """Base controller class for Identity API v2."""
```

Manager 调用 driver 是从 keystone 的配置文件中读取的，例如 identity 的配置文件就记录在 keystone.conf 中：

```
[identity]
driver = keystone.identity.backends.sql.Identity
```

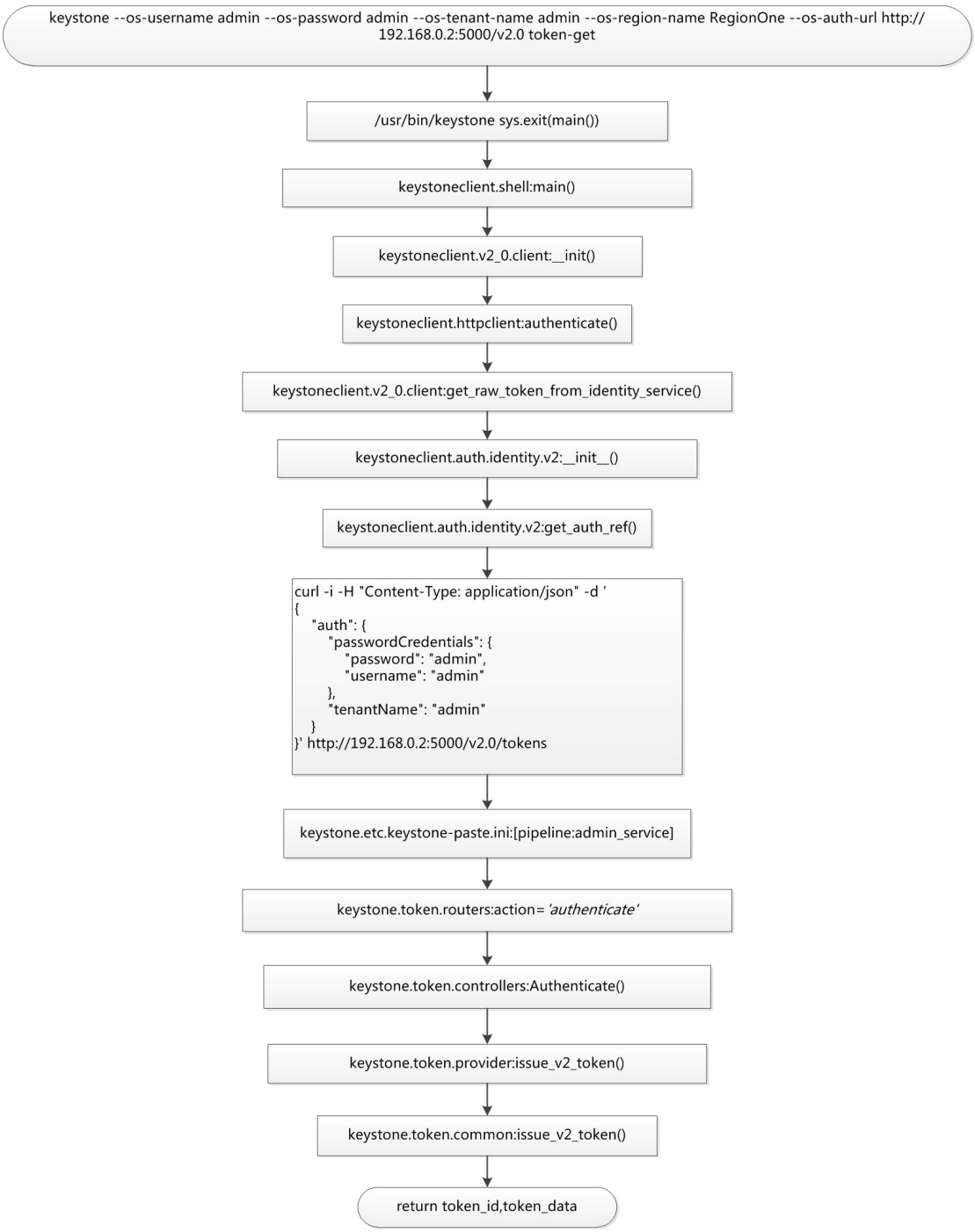
源码分析

Keystone 是 OpenStack 的一个核心项目，在 OpenStack 框架中通过应用程序编程接口（API）提供身份验证、服务目录、服务策略和服务令牌的功能。为 OpenStack 其他各个组件提供统一的身份验证服务。



Token 管理

这一部分我们通过 keystone token-get 命令，分析一下从 keystoneclient 入口到 keystoneserver 生成 token_id 和 token_data 的整个流程。



keystoneclient 的入口在/usr/bin/keystone 文件中，在文件中 from keystoneclient.shell import main 导入了/usr/lib/python2.6/site-packages/keystoneclient/shell.py 中的 main 方法，keystoneclient 处理 keystone token-get 相关方法之后，发送 REST API 调用 keystone server，server 收到请求之后 wsgi 处理 url，调用 admin_service factory 来处理此请求，匹配到 token.router，调用对应的 action 的实现来具体执行处理并返回 json。

目前 token 的存储是有 sql 和 memcache 两种方式，/etc/keystone/keystone.conf:[token]driver 参数可以配置对应的存储后端。使用 sql 把 token 存储在 token 表中，由于 token 增长速度过快，可以采用 memcache 的方式存储到内存中，memcache 会自动删除和替换不使用的缓存。

用户注册流程分析

1、 用户填写相关信息进行用户注册

2、 点击立即注册之后调用 keystoneclient 发送 REST API 请求

3、 Keystone server 的 identity.router 匹配到 url 请求 ,并执行 post_action 进行处理

```
mapper, registration_controller,
path='/users/registrations',
post_action='create_registration',
get_action='get_registration_by_query',
rel=json_home.build_v3_resource_relation('user_registrations'),
```

4、 Create_registration 方法中实现了使用用户邮箱创建 domain，在 domain 下创建 Default project，创建用户，在 domain 和 project 下

为赋予用户 admin 角色，最后发送激活链接到用户的注册邮箱。

```
def create_registration(self, context, registration):
    user_ref = self._normalize_username_in_request(registration)
    user_ref = self._normalize_dict(user_ref)
    # check email
    self._validate_email(user_ref)
    # check reg_code
    self._validate_regcode(user_ref)
    self._require_attribute(user_ref, 'name')
    # set user to disabled
    user_ref['enabled'] = False

    # create default domain with the registration, name same to user name
    domain = self._create_default_domain(user_ref['name'])
    user_ref['domain_id'] = domain['id']
    # create default project name as 'Default'
    project = self._create_default_project(domain['id'])
    user_ref['default_project_id'] = project['id']
    user_ref = self._assign_unique_id(self._normalize_dict(user_ref))
    user_ref = self.identity_api.create_user(user_ref)

    # validate registration
    self._send_validate_email(user_ref['email'], user_ref['id'])
    return Registration.wrap_member(context, user_ref)
```

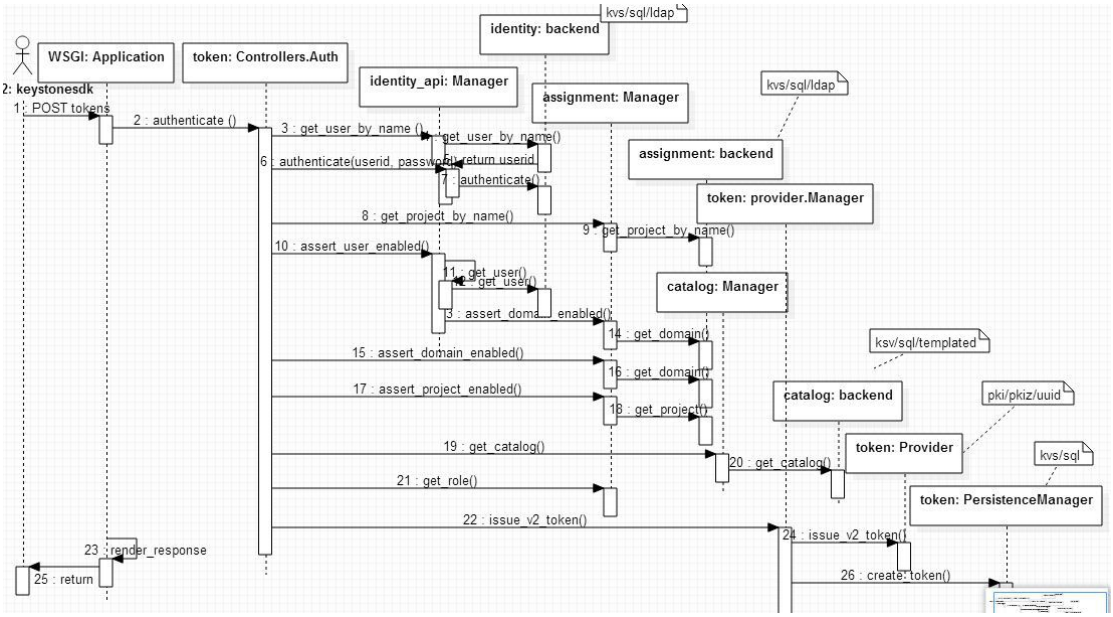
5、 用户点击邮箱中的激活链接，keystone.identity.controllers:update_registration()实现分别激活domain、project、user 的操作，实现用户的激活。

```
def update_registration(self, context, user_id, registration=None):
    old_user_ref = self.identity_api.get_user(user_id)
    domain_id = old_user_ref['domain_id']
    project_id = old_user_ref['default_project_id']
    enabled = old_user_ref.get('enabled', False)
    # if registration is not enabled, the domain maybe not enabled
    if not enabled:
        self._enable_default_domain(domain_id)
        self._enable_default_project(project_id)
        self._add_role_to_registration(user_id, domain_id, project_id)

    # set user to enabled
    if registration:
        registration['enabled'] = True
    else:
        registration = {'enabled': True}

    user_ref = self.identity_api.update_user(user_id,
                                             registration)
    return Registration.wrap_member(context, user_ref)
```

6、 用户注册流程结束，可以使用注册邮箱和密码登录云平台

用户身份验证流程分析

下面通过用户身份验证的过程分析一下源码流程：



首先使用用户的 token 进行身份验证：

```
curl -i \
  -H "Content-Type: application/json" \
  -d '
{
    "auth": {
        "identity": {
            "methods": [
                "token"
            ],
            "token": {
                "id": "'$OS_TOKEN'"
            }
        },
        "scope": {
            "project": {
                "id": "5b50efd009b540559104ee3c03bbb2b7"
            }
        }
    }
}
 http://localhost:5000/v3/auth/tokens ; echo
```

WSGI 接收到请求之后，从 keystone-paste.ini 文件解析 url，最终匹配到：

```
[pipeline:api_v3]

# The last item in this pipeline must be service_v3 or an equivalent

# application. It cannot be a filter.

    pipeline = sizelimit url_normalize build_auth_context token_auth admin_token_auth xml_body_v3 json_body ec2_extension_v3 s3_extension simple_cert_extension revoke_extension service_v3
```

然后一次调用 pipeline 中的 app_factory，最终执行到：

```
[app:service_v3]

    paste.app_factory = keystone.service:v3_app_factory
```

根据 url：/auth/tokens 匹配到 auth 模块的 routers.py：

```
        self._add_resource(
            mapper, auth_controller,
            path='/auth/tokens',
            get_action='validate_token',
            head_action='check_token',
            post_action='authenticate_for_token',
            delete_action='revoke_token',
            rel=json_home.build_v3_resource_relation('auth_tokens'))
```

REST API 的 POST 请求匹配到 post_action=authenticate_for_token，调用 Controller 中的 authenticate_for_token 方法：

```
def authenticate_for_token(self, context, auth=None):
    """Authenticate user and issue a token."""
    include_catalog = 'nocatalog' not in context['query_string']

    try:
        auth_info = AuthInfo.create(context, auth=auth)
        auth_context = AuthContext(extras={},
                                   method_names=[],
                                   bind={})
        self.authenticate(context, auth_info, auth_context)
        if auth_context.get('access_token_id'):
            auth_info.set_scope(None, auth_context['project_id'], None)
        self._check_and_set_default_scoping(auth_info, auth_context)
        (domain_id, project_id, trust) = auth_info.get_scope()

        method_names = auth_info.get_method_names()
        method_names += auth_context.get('method_names', [])
        # make sure the list is unique
        method_names = list(set(method_names))
        expires_at = auth_context.get('expires_at')
        # NOTE(morganfainberg): define this here so it is clear what the
        # argument is during the issue_v3_token provider call.
        metadata_ref = None

        token_audit_id = auth_context.get('audit_id')

        (token_id, token_data) = self.token_provider_api.issue_v3_token(
            auth_context['user_id'], method_names, expires_at, project_id,
            domain_id, auth_context, trust, metadata_ref, include_catalog,
            parent_audit_id=token_audit_id)

        # NOTE(wanghong): We consume a trust use only when we are using
        # trusts and have successfully issued a token.
        if trust:
            self.trust_api.consume_use(trust['id'])

        return render_token_data_response(token_id, token_data,
                                          created=True)
    except exception.TrustNotFound as e:
        raise exception.Unauthorized(e)
```

执行到调用_check_and_set_default_scoping 方法：

```
self._check_and_set_default_scoping(auth_info, auth_context)
```

_check_and_set_default_scoping 方法中调用 identity_api 和

assignment_api 查询用户信息和用户角色，调用 catalog_api 获取 endpoint：

```
def _check_and_set_default_scoping(self, auth_info, auth_context):
    (domain_id, project_id, trust) = auth_info.get_scope()
    if trust:
        project_id = trust['project_id']
    if domain_id or project_id or trust:
        # scope is specified
        return

    # Skip scoping when unscoped federated token is being issued
    if federation.IDENTITY_PROVIDER in auth_context:
        return

    # fill in default_project_id if it is available
    try:
        user_ref = self.identity_api.get_user(auth_context['user_id'])
    except exception.UserNotFound as e:
        LOG.exception(e)
        raise exception.Unauthorized(e)

    default_project_id = user_ref.get('default_project_id')
    if not default_project_id:
        # User has no default project. He shall get an unscoped token.
        return

    # make sure user's default project is legit before scoping to it
```

最终调用 token_provider_api 验证用户 token，并返回 token_data:


```
{
    "token": {
        "methods": [
            "token"
        ],
        "roles": [
            {
                "id": "5090055d6bd547dc83e0e8f070803708",
                "name": "admin"
            }
        ],
        "expires_at": "2015-11-05T22:00:11.000000Z",
        "project": {
            "domain": {
                "id": "default",
```

```
            "name": "Default"
        },
        "id": "5b50efd009b540559104ee3c03bbb2b7",
        "name": "admin"
    },
    "catalog": [
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:9292",
                    "region": "RegionOne",
                    "interface": "admin",
                    "id": "b2605da9b25943beb49b2bd86aca2202"
                },
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:9292",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "c4d1184caf8c4351bff4bf502a09684e"
                },
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:9292",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "cd73bda89e3948738c2721a8c3acac54"
                }
            ],
            "type": "image",
            "id": "495df2483dc145dbb6b34bfbdd787aae",
            "name": "glance"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8773/",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "7d03218a7f4246e8b9e3992318bf5397"
                },
                {
```

```
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:8773/",
                "region": "RegionOne",
                "interface": "public",
                "id": "9ad7f8ce438c4212b8aac930bca04c86"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:8773/",
                "region": "RegionOne",
                "interface": "admin",
                "id": "d84aad1a45c44e4da09b719167383049"
            }
        ],
        "type": "ec2",
        "id": "54204024bb7d4665a8efc34fc758f1f7",
        "name": "ec2"
    },
    {
        "endpoints": [
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9000",
                "region": "RegionOne",
                "interface": "admin",
                "id": "1077687c18514490a3ec980eadd1bd13"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9000",
                "region": "RegionOne",
                "interface": "public",
                "id": "1e86d8bef1514c3fba8d157a22ccce88"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9000",
                "region": "RegionOne",
                "interface": "internal",
                "id": "f6a6b7bbba66443ead3a0e31a008c271"
            }
        ],
        "type": "messaging-websocket",
        "id": "6b8655af7d044a15bec3cdca4f2919f8",
```

```
            "name": "zaqar-websocket"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8004/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "admin",
                    "id": "083663fd231e40ad97384ad3efb9f1b7"
                },
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8004/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "0f4b7054ea27450eac43f685a4fc1d2c"
                },
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8004/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "5f3ea39df2e44378b1802a1a87ef9ac4"
                }
            ],
            "type": "orchestration",
            "id": "6d6346ff2ca842e5968373fbb93e231f",
            "name": "heat"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8774/v2.1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "bc2230a70d6a444e9fba75b85fbda41b"
                },
                {
```

```
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8774/v2.1/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "internal",
                "id": "d8102dc2b9984d04b30b91b0a6037470"
            },
            {
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8774/v2.1/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "admin",
                "id": "f8253a53edd749bf8b107a53a5d47a82"
            }
        ],
        "type": "compute",
        "id": "75df965385cc4120a17110c1fde00182",
        "name": "nova"
    },
    {
        "endpoints": [
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:35357/v2.0",
                "region": "RegionOne",
                "interface": "admin",
                "id": "0ceeb58592274caea5bc942a07d5473f"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:5000/v2.0",
                "region": "RegionOne",
                "interface": "internal",
                "id": "8126f2c7021d413e9c98ec3a0ba0fd58"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:5000/v2.0",
                "region": "RegionOne",
                "interface": "public",
                "id": "c693879254544e3fb502e795a3f6acc8"
            }
        ],
```

```
            "type": "identity",
            "id": "78aad571d38049e69c866c2abac76af6",
            "name": "keystone"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8776/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "admin",
                    "id": "3654138dc64a45aeb5a8153f2a089c74"
                },
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8776/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "7a0d12d0b7314afd9b53d1618ab546ea"
                },
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8776/v1/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "82b68ff3aedb43e2acc8307234d3fd0b"
                }
            ],
            "type": "volume",
            "id": "80491007c0ab462daaa9087250325f59",
            "name": "cinder"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8000/v1",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "24dfa252fba64469b8b1a832f04bded9"
                },
```

```
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:8000/v1",
                "region": "RegionOne",
                "interface": "public",
                "id": "e0a01d6cd3be4f6abcc72367b2d87993"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:8000/v1",
                "region": "RegionOne",
                "interface": "admin",
                "id": "f33f79d42df247e1bf6daf43a548b014"
            }
        ],
        "type": "cloudformation",
        "id": "ac5cc6e3c62840818ab338c981d5603f",
        "name": "heat-cfn"
    },
    {
        "endpoints": [
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9696/",
                "region": "RegionOne",
                "interface": "admin",
                "id": "3e78c357b3c8469fbea12eb681f88a0c"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9696/",
                "region": "RegionOne",
                "interface": "public",
                "id": "89d2aad3dc8e478fbabb21dd7db0962a"
            },
            {
                "region_id": "RegionOne",
                "url": "http://23.253.248.171:9696/",
                "region": "RegionOne",
                "interface": "internal",
                "id": "b6d4a8cf5e4042848a749a3116497e55"
            }
        ],
        "type": "network",
```

```
            "id": "b33660edd1eb45e485f7e5f14401a739",
            "name": "neutron"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8888",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "1f8287cf963948778ab0eb109d9f857d"
                },
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8888",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "3adf5f9cc5184d92af5ff0fdef043e4a"
                },
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8888",
                    "region": "RegionOne",
                    "interface": "admin",
                    "id": "f747223060b3414f947fdcdca2ce8714"
                }
            ],
            "type": "messaging",
            "id": "cf3e38e9aed54e2d84ea64485317d7a0",
            "name": "zaqar"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8774/v2/5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "08f507ccb552476b98f3af7718f25557"
                },
                {
                    "region_id": "RegionOne",
```

```
                "url":
"http://23.253.248.171:8774/v2/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "admin",
                "id": "d20091ba591347b2b419e5fbde9b7976"
            },
            {
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8774/v2/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "internal",
                "id": "e6b667776e7245dea6e39f2820e080b0"
            }
        ],
        "type": "compute_legacy",
        "id": "d442e96b273a48018567aeec5800c3e0",
        "name": "nova_legacy"
    },
    {
        "endpoints": [
            {
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8776/v2/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "internal",
                "id": "012c78a6694a494995c58d5955fb7822"
            },
            {
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8776/v2/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "admin",
                "id": "802d5de210874f068ba31c7e27c29d70"
            },
            {
                "region_id": "RegionOne",
                "url":
"http://23.253.248.171:8776/v2/5b50efd009b540559104ee3c03bbb2b7",
                "region": "RegionOne",
                "interface": "public",
                "id": "b37ada66e02e44c9a9a7976d77365503"
```

```
                }
            ],
            "type": "volumev2",
            "id": "d93e78c7967f49acbdd732b9dd97e0d0",
            "name": "cinderv2"
        },
        {
            "endpoints": [
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8080/v1/AUTH_5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "public",
                    "id": "265ce88a0e1642fc90b2ec20ccb279ff"
                },
                {
                    "region_id": "RegionOne",
                    "url": "http://23.253.248.171:8080",
                    "region": "RegionOne",
                    "interface": "admin",
                    "id": "500b7f066d39492faff8a3f710fb5a2f"
                },
                {
                    "region_id": "RegionOne",
                    "url":
"http://23.253.248.171:8080/v1/AUTH_5b50efd009b540559104ee3c03bbb2b7",
                    "region": "RegionOne",
                    "interface": "internal",
                    "id": "a33b0684f817405280df1f5600777a75"
                }
            ],
            "type": "object-store",
            "id": "da1b1b5c529946fcb3ee3abdcf376fcb",
            "name": "swift"
        }
    ],
    "extras": {},
    "user": {
        "domain": {
            "id": "default",
            "name": "Default"
        },
        "id": "10a2e6e717a245d9acad3e5f97aeca3d",
```

```
            "name": "admin"
        },
        "audit_ids": [
            "wLc7nDMsQiKqf8VFU4ySpg"
        ],
        "issued_at": "2015-11-05T21:32:30.505384Z"
    }
}
```