获取 token

功能介绍

兼容 Openstack Keystone 的 /v3/auth/tokens 接口，用于登录获取 token（在 response header 中），以及 token 下对应的 project+OpSet（操作集）权限或 domain+OpSet 权限。

URI

- URI 格式

  POST /v3/auth/tokens

请求消息

- 获取 token 的方式

参数	必选/可选	类型	描述
password	可选	String	支持用户 ID + 密码，用户名 + 密码 + Domain ID，用户名 + 密码 + Domain Name
totp	可选	String	当开启了虚拟 MFA 方式的登录保护时，该参数必填。
token	可选	String	token 原生语义通过 token 来换取 token
hw_renew_token	可选	String	Console 在 token 过期前进行续期

参数	必选/可选	类型	描述
hw_access_key	可选	String	API 网关通过 access 获取 token
hw_assume_role	可选	String	用于企业间权限委托（如代维）、延期执行、提权等场景

- Request Header 参数说明

参数	必选/可选	类型	说明
Content-Type	必选	String	该字段内容填为“application/json;charset=utf8”。
X-Auth-Token	必选(password 方式除外)	String	获取 token 方式： token 或 hw_renew_token：拥有 op_service、op_cred、op_auth 或 op_bss 权限的 token hw_access_key：拥有 op_cred 权限的 token hw_assume_role：租户名为 op_service、op_team、te_agency 或拥有 op_fine_grained 权限的 token

- Request Body 参数说明

参数	是否为必选	类型	说明
methods	是	String Array	该字段内容填为“hw_assume_role”。
domain_name	是	String	指定租户名。
xrole_name	是	String	委托账户中配置的角色名字。

参数	是否为必选	类型	说明
restrict	否	Json Object	限定的权限集，填写 roles 的信息。
roles	否	List Object	权限信息。
scope	否	Json Object	token 所属范围信息。

- 请求示例
  - user.id 为 0ca8f6，user.password 为 secrete 的用户登录，获取 scope 为 project.id=263fd9 的 Token

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "id": "0ca8f6",// 如果选择 name，则必须在 password 下面加上 domain 的信息
                    "password": "secrete"
                }
            },
            "hw_context": {// 扩展字段，可以设置在每个 token 上
                "order_id": "2015031010000032"
            }
        },
        "scope": {// 每个 Token 必须关联一个 Project 或 Domain
          "project": {   // project 和 domain 各选其一，优先选择 project
                "id": "263fd9",      // id 和名字各选其一，优先选择 name
                "name": "espace",    // name 还要指定 domain 信息
                "domain": {
                    "id": "default", // id 和名字各选其一，优先选择 name
                    "name": "default"
                }
            }
        }
```

```
    }
}
```

用户名密码登录最简可写为

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "id": "0ca8f6", // 参见密码验证例子
                    "password": "secrete"
                }
            }
        }
    }
}
```

开启虚拟 MFA 方式登录保护时可写为

```
{
    "auth": {
        "identity": {
            "methods": [
                "password",
                "totp"
            ],
            "password": {
                "user": {
                    "name": "name",
                    "password": "**********",
                    "domain": {
                        "name": "name"
                    }
                }
            },
            "totp": {
                "user": {
                    "id": "id",
                    "passcode": "******"
                }
            }
        },
        "scope": {
```

```
            "domain": {
                "name": "name"
            }
        }
    }
}
```

- 使用 token 换 token

  该方法在切换 project 时使用。

```
{
    "auth": {
        "identity": {
            "methods": [
                "token"
            ],
            "token": {
                "id": "{tokenId}"
            }
        },
        "scope": {
            "project": {
                "id": "{projectId}"
            }
        }
    }
}
```

- 用 hw_renew_token 进行续期

  使用 token_id 获取一个新的 token。主要是会话管理服务来调用，用于保证在会话有效时间内 token 不会失效。应用服务基本不需理会。续期时间系统默认，不需要设置。

```
{
    "auth": {
        "identity": {
            "methods": [
                "hw_renew_token"
            ],
            "hw_renew_token": {
                "token": {
                    "id": "e80b74"
                }
            }
        }
    }
```

```
}
```

- 通过 access 获取 token

  API 网关中通过 access 获取 token

```
{
    "auth": {
        "identity": {
            "methods": [
                "hw_access_key"
            ],
            "hw_access_key": {
                "access": {
                    "key": "xxxXX",//和 securitytoken 二选一
                    "securitytoken": "xxxXX"//和 key 二选一，临时 ak 支持该项操
作
                }
            }
        },
        "scope": {
            "project": {
                "id": "263fd9"
            }
        }
    }
}
```

- 用 assume_role 完成委托代理

```
{
    "auth": {
        "identity": {
            "methods": [
                "hw_assume_role"
            ],
            "hw_assume_role": {
                "domain_id": "100021000101",// 创建委托的租户 ID，和
domain_name 二选一
                "domain_name": "test_domain_name",// 创建委托的租户名，和
domain_id 二选一
                "xrole_name": "delegate_iaas"// 委托名称
            },
            "hw_context": {
                "order_id": "2015031010000032" // 可选项，设置 token 的上下
文，如账单号，EC2 报话单的时候使用此参数
            }
```

```
        },
        "scope": { // 请参考密码验证例子
            "project": {
                "id": "263fd9"
            }
        }
    }
}
```

- 用 assume_role 完成服务延期执行

实现服务的延期执行。例如订单开通服务和服务真正开通的时间往往不一样（需要人工审批触发），或 autoscalling 服务设置和应用服务真正扩容的时间也不一样（需要 Ceilometer 服务触发），因此我们使用了 assume role 的概念，认为是客户委托对应的应用服务去操纵资源的行为。以订单服务为例，订单服务在订单审批通过后，通过 assume 到 domain_id=100021000101，xrole_name=op_service 的角色上进行资源创建，订单服务所在的域 op_service 已经是内置关联其他所有域了。

```
{
    "auth": {
        "identity": {
            "methods": [
                "hw_assume_role"
            ],
            "hw_assume_role": {
                "domain_id": "100021000101",   // 客户域账号 ID.
                "xrole_name": "op_service",// 客户账户中配置的角色，该角色名称固定，在客户域账户创建时默认创建并关联信任域账号 op_service（系统内置域账号，指所有 HWS 内部服务，例如订单服务）
                "restrict": {   // 可选，用于收窄权限，因为 op_service 的权限是 admin 的。并且方式支持以下任一
                    "user_id": "xxxxxx",
                    "user_name": "abc",
                    "roles": [
                        "create"
                    ]
                }
            },
            "hw_context": {
                "order_id": "2015031010000032"// 可选项，设置 token 的上下文，如账单号，EC2 报话单的时候使用此参数
            }
        },
        "scope": {// 请参考密码验证例子
            "project": {
```

```
                "id": "263fd9"
            }
        }
    }
}
```

- 用 assume_role 完成组合业务程序的提权

实现组合业务的临时提权。部分业务需要临时提权，我们同样使用了 assume role 的概念，认为是客户具备更高权限的 xrole，同时委托业务服务 assume 到这个 xrole 上进行资源操作

```
{
    "auth": {
        "identity": {
            "methods": [
                "hw_assume_role"
            ],
            "hw_assume_role": {
                "domain_id": "100021000101", // 客户域账号 ID.
                "xrole_name": "op_service",// 客户账户中配置的角色，该角色名称固定，在客户域账户创建时默认创建并关联信任域账号 op_service（系统内置域账号，指所有 HWS 内部服务，例如订单服务）
                "restrict": {// 可选，并且方式支持以下任一
                    "user_id": "xxxxxx",
                    "user_name": "abc",
                    "roles": [ // optset 名称
                        "admin"
                    ]
                }
            },
            "hw_context": {
                "order_id": "2015031010000032"// 可选项，设置 token 的上下文，如账单号，EC2 报话单的时候使用此参数
            }
        },
        "scope": {
            "project": {
                "id": "263fd9"
            }
        }
    }
}
```

响应消息

- Response header 参数说明

参数	是否为必选	类型	描述
X-Subject-Token	是	String	pki 签名字符串，校验 token 是否被篡改
Content-Type	是	String	该字段内容填为“application/json;charset=utf8”。

- Token 格式说明

参数	是否为必选	类型	描述
methods	是	String	获取 token 的方式
expires_at	是	String	token 超时时间
issued_at	是	String	token 产生时间
mfa_authn_at	否	String	用户 MFA 认证成功的时间
user	是	Json Object	示例： "user": { "name": "*username*", "id": "*userid*", "domain": { "name": "*domainname*", "id": "*domainid*" } } *username*：用户名称。 *userid*： 用户 id。 *domainname*：用户所属的企业账户名称。 *domainid*：用户所属的企业账户的域 id。

参数	是否为必选	类型	描述
domain	否	Json Object	根据请求中的 scope，判断是否返回该字段。 示例： "domain": { "name" : "domainame", "id" : "domainid" } *domainname：*企业账户名称。 *domainid：*企业账户的域 id。
project	否	Json Object	根据请求中的 scope，判断是否返回该字段。 示例： "project": { "name": "*projectname*", "id": "*projectid*", "domain": { "name": "*domainname*", "id": "*domainid*" } } *projectname：*project 名称。 *projectid：*project 的 id。 *domainname：*poject 所属的企业账户名称。 *domainid：*project 所属的企业账户的域 id。
roles	是	Json Object	示例： "roles" : [{ "name" : "role1", "id" : "roleid1" }, { "name" : "role2", "id" : "roleid2" }]

参数	是否为必选	类型	描述
			name：权限名称。 *id*：权限 id。

- 响应示例
  - 如果使用用户名密码和虚拟 MFA 验证码（totp）获取 token，且指定了 project

```
{
    "token": {
        "catalog": [],
        "expires_at": "2015-02-27T18:30:59.999999Z",
        "issued_at": "2015-02-27T16:30:59.999999Z",
        "mfa_authn_at": "2015-02-27T16:30:59.999999Z",
        "methods": [
            "password",
            "totp"
        ],
        "project": {
            "domain": {
                "id": "1789d1",
                "links": {
                    "self": "http://identity:35357/v3/domains/1789d1"
                },
                "name": "example.com"
            },
            "id": "263fd9",
            "links": {
                "self": "http://identity:35357/v3/projects/263fd9"
            },
            "name": "project-x"
        },
        "roles": [
            {
                "id": "76e72a",
                "links": {
                    "self": "http://identity:35357/v3/roles/76e72a"
                },
                "name": "admin"
            },
            {
                "id": "f4f392",
```

```
                "links": {
                    "self": "http://identity:35357/v3/roles/f4f392"
                },
                "name": "member"
            }
        ],
        "user": {
            "domain": {
                "id": "1789d1",
                "links": {
                    "self": "http://identity:35357/v3/domains/1789d1"
                },
                "name": "example.com"
            },
            "id": "0ca8f6",
            "links": {
                "self": "http://identity:35357/v3/users/0ca8f6"
            },
            "name": "username"//如果使用 assumeRole，此项将显示 xrole 的名字
        },
        "hw_context": { // 扩展字段， 如果没有上下文将不显示
            "order_id": "2015031010000032"// 在 Assume 的时候设置 token 的上下文，如账单号，EC2 报话单的时候使用此参数
        }
    }
}
```

如果指定了 domain

```
{
    "token": {
        "domain": {
            "id": "default",
            "name": "Default"
        },
        "methods": [
            "password"
        ],
        "roles": [
            {
                "id": "799202c6c109493493e5b2b6ff473c89",
                "name": "admin"
            },
            {
                "id": "89f11b3e4c89438eb41a448d8322a86d",
                "name": "service"
```

```
            }
        ],
        "expires_at": "2015-03-27T19:03:12.739076Z",
        "extras": {},
        "user": {
            "domain": {
                "id": "default",
                "name": "Default"
            },
            "id": "6c4a6c26aa254825a875c2a21f353f68",
            "name": "admin"
        },
        "audit_ids": [
            "JHMbMW47RQOUDNr0VKhGBg"
        ],
        "issued_at": "2015-03-27T18:03:12.739108Z"
    }
}
```


状态码

状态码	说明
201	请求成功。
400	请求错误。
401	认证失败。
403	鉴权失败。
404	找不到资源。
500	内部服务错误。
503	服务不可用。