

---

# **Security Interface Documentation**

**Bakhtiyor Ruziev**

**Feb 17, 2023**



---

## Contents

---

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Usage</b>	<b>5</b>
2.1	Create security instance with our implementation . . . . .	5
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



This library provides an easy API for authentication and authorization.



# CHAPTER 1

---

## Installation

---

Install with the following command:

```
$ pip install security_interface
```





## CHAPTER 2

---

### Usage

---

First of all you need to implement `IdentityPolicyInterface()` and `AuthorizationPolicyInterface()` interfaces. For example, we can implement JWT Security:

```
import jwt
from security_interface import IdentityPolicyInterface, AuthorizationPolicyInterface

class JwtIdentityPolicy(IdentityPolicyInterface):
    def __init__(self, secret, algorithm="HS256"):
        self.algorithm = algorithm
        self.secret = secret

    async def identify(self, identity):
        if jwt is None:
            raise TypeError("Please install PyJWT")
        try:
            return jwt.decode(
                identity,
                self.secret,
                algorithms=[self.algorithm],
                options={"verify_exp": True, "verify_iat": True},
            )
        except Exception as e:
            return None

class JwtAuthPolicy(AuthorizationPolicyInterface):
    async def can(self, identity, permission):
        return permission in identity["scope"]
```

### 2.1 Create security instance with our implementation

```
from security_interface.api import Security
jwt_identity = JwtIdentityPolicy("SECRET")
jwt_auth_policy = JwtAuthPolicy()
security = Security(jwt_identity, jwt_auth_policy)
# Checking claim
security.identify(CLAIM)
# Checking permission
security.can(CLAIM, "read")
security.can(CLAIM, "write")
```

For full implementation see [DEMO](#)

### 2.1.1 Security Interface API

#### Interfaces

**class** security\_interface.AuthorizationPolicyInterface

Bases: object

**can** (identity, permission)

You need to implement checking permission.

**Returns** True if the identity is allowed the permission, else return False.

**class** security\_interface.IdentityPolicyInterface

Bases: object

**identify** (identity)

You need return the checked claimed identity or None if check is fail.

**Parameters** identity – Claim

**Returns** Checked identity or None if check is failed.

#### Main API

**class** security\_interface.api.Security (identity\_policy: security\_interface.IdentityPolicyInterface, autz\_policy: security\_interface.AuthorizationPolicyInterface)

Bases: object

**can** (identity, permission) → bool

Check user permissions.

**Returns** True if the identity is allowed the permission, else return False.

**check\_authorized** (identity)

Works like Security.identify(), but when check is failed UnauthorizedError() exception is raised.

**Parameters** identity – Claim

**Returns** Checked claim or return None

**Raise** UnauthorizedError()

**check\_permission** (identity, permission)

Works like Security.can(), but when check is failed ForbiddenError() exception is raised.

**Parameters**

- **identity** – Claim
- **permission** – Permission

**Returns** Checked claim

**Raise** `ForbiddenError()`

**identify** (*identity*)

Return the claimed identity or `None` if check is failed.

**Parameters** **identity** – Claim

**Returns** Checked identity or `None` if check is failed.

**is\_anonymous** (*identity*) → bool

**Parameters** **identity** – Claim

**Returns** `True` if user anonymous otherwise `False`

## Exceptions

**exception** `security_interface.exceptions.ForbiddenError`

Bases: `Exception`

**exception** `security_interface.exceptions.UnauthorizedError`

Bases: `Exception`



### S

`security_interface`, [6](#)  
`security_interface.api`, [6](#)  
`security_interface.exceptions`, [7](#)



## A

AuthorizationPolicyInterface (*class in security\_interface*), 6

## C

can() (*security\_interface.api.Security method*), 6

can() (*security\_interface.AuthorizationPolicyInterface method*), 6

check\_authorized() (*security\_interface.api.Security method*), 6

check\_permission() (*security\_interface.api.Security method*), 6

## F

ForbiddenError, 7

## I

identify() (*security\_interface.api.Security method*), 7

identify() (*security\_interface.IdentityPolicyInterface method*), 6

IdentityPolicyInterface (*class in security\_interface*), 6

is\_anonymous() (*security\_interface.api.Security method*), 7

## S

Security (*class in security\_interface.api*), 6

security\_interface (*module*), 6

security\_interface.api (*module*), 6

security\_interface.exceptions (*module*), 7

## U

UnauthorizedError, 7