# **Проверка подлинности (Authentication)**

Использует проверку логина и пароля для предоставления доступа к системе:

class Authentication:

def \_\_init\_\_(self):

self.users = {

'john': 'password123',

'mary': 'qwerty'

}

def login(self, username, password):

if username in self.users and self.users[username] == password:

return True

return False

# **Авторизация ролей (Role-based authorization)**

Предоставляет доступ в зависимости от роли пользователя:

class Authorization:

def \_\_init\_\_(self):

self.permissions = {

'admin': ['full\_access'],

'operator': ['view\_stats', 'update\_mission'],

'engineer': ['diagnose\_failure', 'tune\_parameters']

}

def check\_permission(self, role, permission):

if permission in self.permissions[role]:

return True

else:

return False

auth = Authorization()

if auth.check\_permission('operator', 'update\_mission'):

print('Access granted')

else:

print('Access denied')

# **Шифрование данных**

Защищает конфиденциальные данные при передаче:

from cryptography.fernet import Fernet

key = Fernet.generate\_key()

f = Fernet(key)

data = 'secret drone data'

encrypted\_data = f.encrypt(data.encode())

# передача данных

decrypted\_data = f.decrypt(encrypted\_data)

print(decrypted\_data.decode())

# выводит 'secret drone data'

Такие паттерны помогают сделать системы БПЛА более защищенными.