Python: creare una CSR con chiave privata

In questo articolo andremo a vedere come creare una CSR (Certificate Signing Request) e una chiave privata RSA con Python.

Per svolgere il nostro compito, abbiamo bisogno di raccogliere le seguenti informazioni:

1. Codice paese (ad es. US)
2. Nome dello stato o della provincia (ad es. Virginia)
3. Nome della località (ad es. Richmond)
4. Nome dell'organizzazione (ad es. La mia azienda)
5. Nome comune (il nome a dominio, ad esempio example.com)

Possiamo quindi scrivere una funzione che crei la CSR insieme alla chiave privata e le restituisca come stringhe.

```
from cryptography.hazmat.primitives.asymmetric import
rsa
from cryptography import x509
from cryptography.x509.oid import NameOID
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives import
serialization

def create_csr(attrs):
    key = rsa.generate_private_key(
        public_exponent=65537,
        key_size=2048
    )
```

```
    country = attrs.get('country','')
    state = attrs.get('state','')
    locality = attrs.get('locality','')
    organization = attrs.get('organization','')
    common_name = attrs.get('common_name', '')
    dns_name1 = common_name
    dns_name2 = f'www.{common_name}'

    csr =
x509.CertificateSigningRequestBuilder().subject_name(
x509.Name([
        x509.NameAttribute(NameOID.COUNTRY_NAME,
country),

x509.NameAttribute(NameOID.STATE_OR_PROVINCE_NAME,
state),
        x509.NameAttribute(NameOID.LOCALITY_NAME,
locality),
        x509.NameAttribute(NameOID.ORGANIZATION_NAME,
organization),
        x509.NameAttribute(NameOID.COMMON_NAME,
common_name)
    ])).add_extension(
        x509.SubjectAlternativeName([
            x509.DNSName(dns_name1),
            x509.DNSName(dns_name2)
        ]),
        critical=False,
    ).sign(key, hashes.SHA256())

    return {'key': key.private_bytes(
            encoding=serialization.Encoding.PEM,
```

```
format=serialization.PrivateFormat.TraditionalOpenSSL
,

encryption_algorithm=serialization.NoEncryption()).de
code('utf-8'), 'csr': csr.public_bytes(

encoding=serialization.Encoding.PEM).decode('utf-8')}
```

Per assicurarsi che funzioni sempre, controllate se la libreria OpenSSL è installata sulla macchina ed è correttamente riconosciuta da Python.