接口文档

SAES 类方法说明

1. `encrypt`

描述:
使用指定的密钥对明文进行AES加密。

参数:

- `plaintext` (int): 待加密的明文，如 `0x0f0f`。
- `key` (int): 加密密钥，如 `0x2D55`。

返回:

- (int): 返回AES加密后的密文，如 `0x1234`。

2. `decrypt`

描述:
使用指定的密钥对AES密文进行解密。

参数:

- `ciphertext` (int): 待解密的密文，如 `0x1234`。
- `key` (int): 解密密钥，如 `0x2D55`。

返回:

- (int): 返回AES解密后的明文，如 `0x0f0f`。

3. `encrypt_string`

描述:
将字符串使用AES加密并返回Base64编码的密文。

参数:

- `input_str` (str): 待加密的字符串。
- `key` (int): 加密密钥，如 `0x2D55`。

返回:

- (str): 返回Base64编码的AES加密密文。

4. `decrypt_string`

描述:
解密Base64编码的AES密文并返回明文字符串。

参数:

- `encrypted_base64_str` (str): 待解密的Base64编码密文。
- `key` (int): 解密密钥，如 `0x2D55`。

返回:

- (str): 返回解密后的明文字符串。

5. `double_encrypt`

描述:
使用两个指定的密钥对明文进行双重AES加密。

参数:

- `plaintext` (int): 待加密的明文，如 `0x0f0f`。
- `key1` (int): 第一个加密密钥。
- `key2` (int): 第二个加密密钥。

返回:

- (int): 返回双重AES加密后的密文。

6. `double_decrypt`

描述:
使用两个指定的密钥对双重AES密文进行解密。

参数:

- `ciphertext` (int): 待解密的双重AES密文。
- `key1` (int): 第一个解密密钥。
- `key2` (int): 第二个解密密钥。

返回:

- (int): 返回双重AES解密后的明文。

7. `tri_encrypt`

描述:
使用三个指定的密钥对明文进行三重AES加密。

参数:

- `plaintext` (int): 待加密的明文。
- `key1` (int): 第一个加密密钥。
- `key2` (int): 第二个加密密钥。
- `key3` (int): 第三个加密密钥。

返回:

- (int): 返回三重AES加密后的密文。

8. `tri_decrypt`

描述:
使用三个指定的密钥对三重AES密文进行解密。

参数:

- `ciphertext` (int): 待解密的三重AES密文。

- `key1` (int): 第一个解密密钥。
- `key2` (int): 第二个解密密钥。
- `key3` (int): 第三个解密密钥。

返回:

- (int): 返回三重AES解密后的明文。

9. `MitMAttack`

描述:
利用中间人攻击尝试恢复加密使用的密钥对。

参数:

- `pairs` (list): 一个包含多个明文和相应密文对的列表。每个元素是一个字典，包含`'plaintext'`和`'ciphertext'`两个键。

返回:

- (list): 返回可能的密钥对 (k1, k2) 列表。如果没有找到匹配的密钥对，则返回`None`。

10. `CBC_encrypt`

描述:
使用CBC模式和指定的密钥对明文进行AES加密。

参数:

- `longtext` (int): 待加密的长明文。
- `key` (int): 加密密钥。
- `IV` (int): 初始化向量。

返回:

- (int): 返回AES加密后的密文。

11. `CBC_decrypt`

描述:
使用CBC模式和指定的密钥对AES密文进行解密。

参数:

- `longcipher` (int): 待解密的AES密文。
- `key` (int): 解密密钥。
- `IV` (int): 初始化向量。

返回:

- (int): 返回AES解密后的明文。

Flask API 文档

1. 首页

Endpoint: `/`

Method: `GET`

Description: 返回首页的HTML内容。

---

2. 加密（简单模式）

Endpoint: `/api/encrypt`

Method: `POST`

Body:

```
{
    "plaintext": "<十六进制格式的明文>",
    "key": "<十六进制格式的密钥>"
}
```

Response:

```
{
    "ciphertext": "<十六进制格式的密文>"
}
```

Description: 使用给定的密钥加密十六进制格式的明文。

---

3. 解密（简单模式）

Endpoint: `/api/decrypt`

Method: `POST`

Body:

```
{
    "ciphertext": "<十六进制格式的密文>",
    "key": "<十六进制格式的密钥>"
}
```

Response:

```
{
    "plaintext": "<十六进制格式的明文>"
}
```

Description: 使用给定的密钥解密十六进制格式的密文。

---

4. 字符串加密

Endpoint: `/api/encrypt_string`

Method: `POST`

Body:

```
{
    "input_str": "<待加密的字符串>",
    "key": "<十六进制格式的密钥>"
}
```

Response:

```
{
    "encrypted_base64_str": "<Base64编码的加密结果>"
}
```

Description: 使用给定的密钥加密字符串，并返回Base64编码的加密结果。

---

5. 字符串解密

Endpoint: `/api/decrypt_string`

Method: `POST`

Body:

```
{
    "encrypted_base64_str": "<Base64编码的加密字符串>",
    "key": "<十六进制格式的密钥>"
}
```

Response:

```
{
    "output_str": "<解密后的字符串>"
}
```

Description: 使用给定的密钥解密Base64编码的加密字符串。

---

6. 双密钥加密

Endpoint: `/api/double_encrypt`

Method: `POST`

Body:

```
{
    "plaintext": "<十六进制格式的明文>",
    "key": "<连续的两个十六进制格式的密钥>"
}
```

Response:

```
{
    "double_ciphertext": "<十六进制格式的双密文>"
}
```

Description: 使用两个密钥进行加密操作。

---

7. 双密钥解密

Endpoint: `/api/double_decrypt`

Method: `POST`

Body:

```
{
    "ciphertext": "<十六进制格式的双密文>",
    "key": "<连续的两个十六进制格式的密钥>"
}
```

Response:

```
{
    "double_plaintext": "<十六进制格式的双明文>"
}
```

Description: 使用两个密钥进行解密操作。

---

8. 三重加密

Endpoint: `/api/tri_encrypt`

Method: `POST`

Body:

```
{
    "plaintext": "<十六进制格式的明文>",
    "key": "<连续的三个十六进制格式的密钥>"
}
```

Response:

```
{
    "tri_ciphertext": "<十六进制格式的三重密文>"
}
```

Description: 使用三个密钥进行加密操作。

---

9. 三重解密

Endpoint: `/api/tri_decrypt`

Method: `POST`

Body:

```
{
    "ciphertext": "<十六进制格式的三重密文>",
    "key": "<连续的三个十六进制格式的密钥>"
}
```

Response:

```
{
    "tri_plaintext": "<十六进制格式的三重明文>"
}
```

Description: 使用三个密钥进行解密操作。

---

10. 破解密钥

Endpoint: `/api/crack`

Method: `POST`

Body:

```
{
    "pairs": [
        {
            "plaintext": "<十六进制格式的明文>",
            "ciphertext": "<十六进制格式的密文>"
        },
        ...
    ]
}
```

Response:

```
{
    "result": {
        "keys": [
            {
                "key1": "<第一个密钥>",
```

```
                "key2": "<第二个密钥>"
            },
            ...
        ],
        "timeTaken": "<操作所用的时间（秒）>"
    },
    "status": "<状态：'success'或'failure'>",
    "message": "<相关消息>"
}
```

Description: 通过提供的明密文对尝试破解密钥。

---

11. CBC模式加密

Endpoint: `/api/cbc/encrypt`

Method: `POST`

Body:

```
{
    "longtext": "<十六进制格式的长明文>",
    "key": "<十六进制格式的密钥>",
    "IV": "<十六进制格式的初始化向量>"
}
```

Response:

```
{
    "encrypted": "<十六进制格式的加密结果>"
}
```

Description: 使用CBC模式和给定的密钥、初始化向量进行加密。

---

12. CBC模式解密

Endpoint: `/api/cbc/decrypt`

Method: `POST`

Body:

```
{
    "longcipher": "<十六进制格式的长密文>",
    "key": "<十六进制格式的密钥>",
    "IV": "<十六进制格式的初始化向量>"
}
```

Response:

```
{
    "decrypted": "<十六进制格式的解密结果>"
}
```

Description: 使用CBC模式和给定的密钥、初始化向量进行解密。