S-AES测试结果

**第1关：基本测试**

**测试目标：**验证S-AES算法程序是否能够正确加密和解密16位的数据。

**测试结果：**经过测试，S-AES算法程序能够正确加密和解密16位的数据，并产生准确的密文。

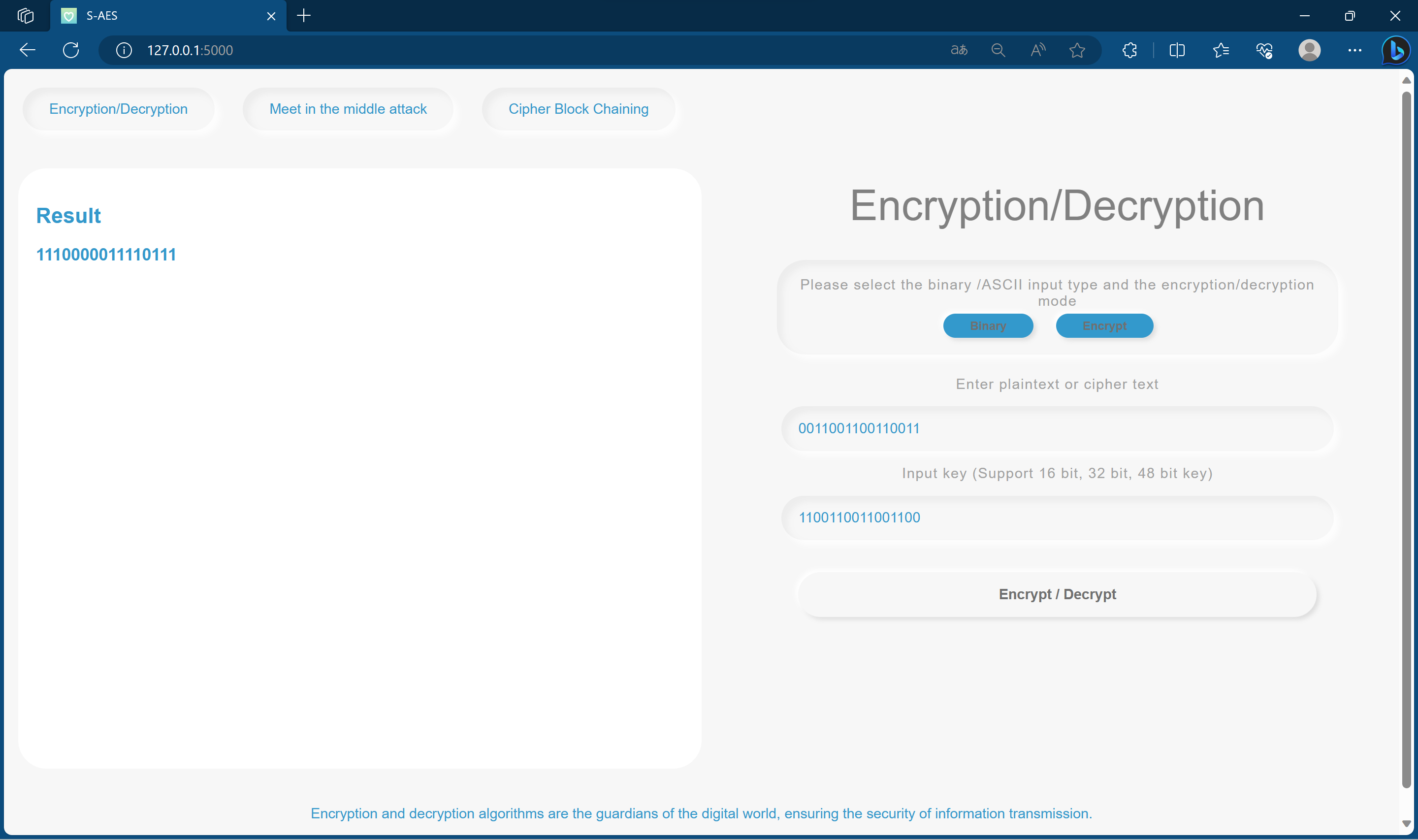
测试步骤和结果：

（1）加密：

明文：0011001100110011

密钥：1100110011001100

密文：1110000011110111

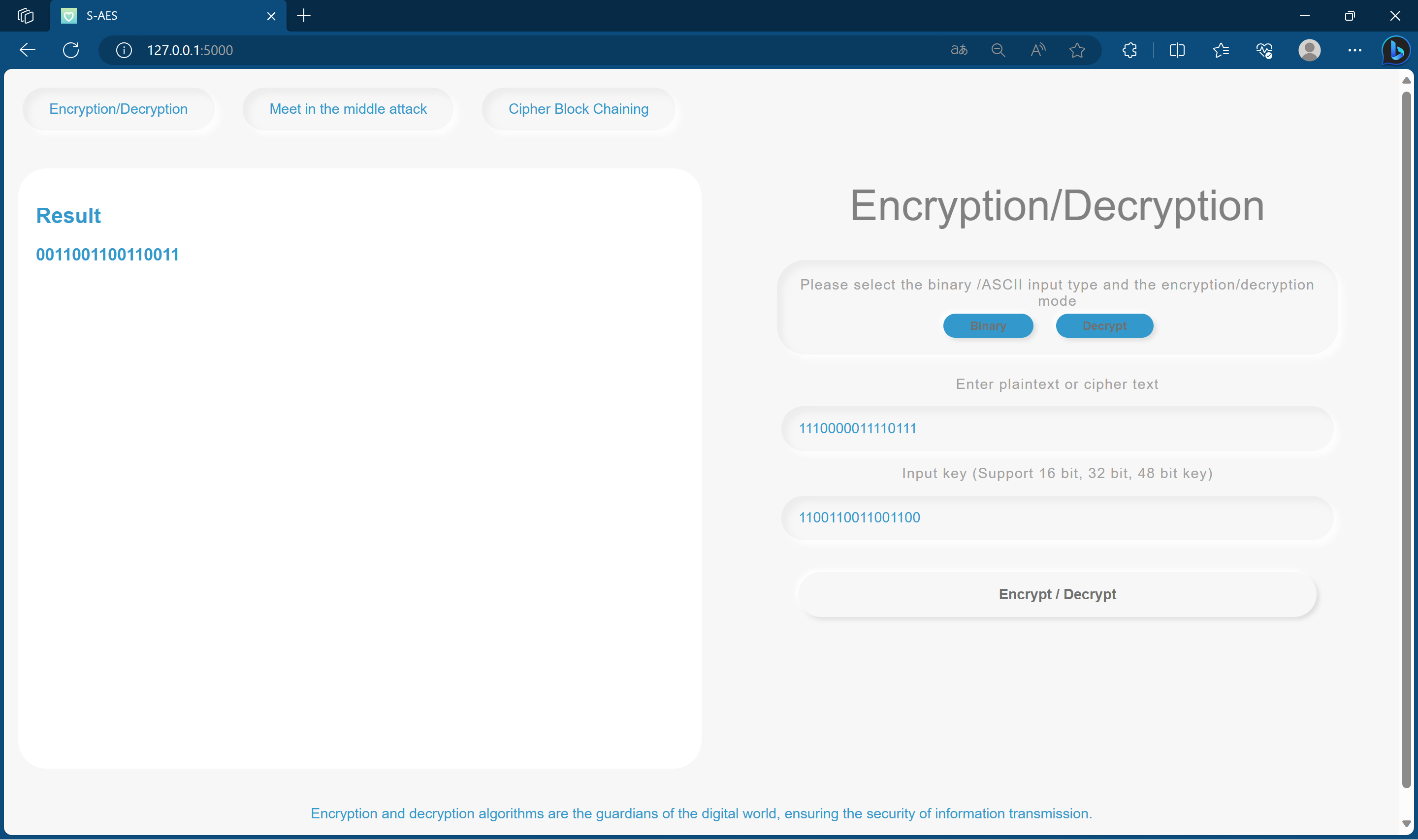


（2）解密：

密文：1110000011110111

密钥：1100111100001111

明文：1110000000011100



**第2关：交叉测试**

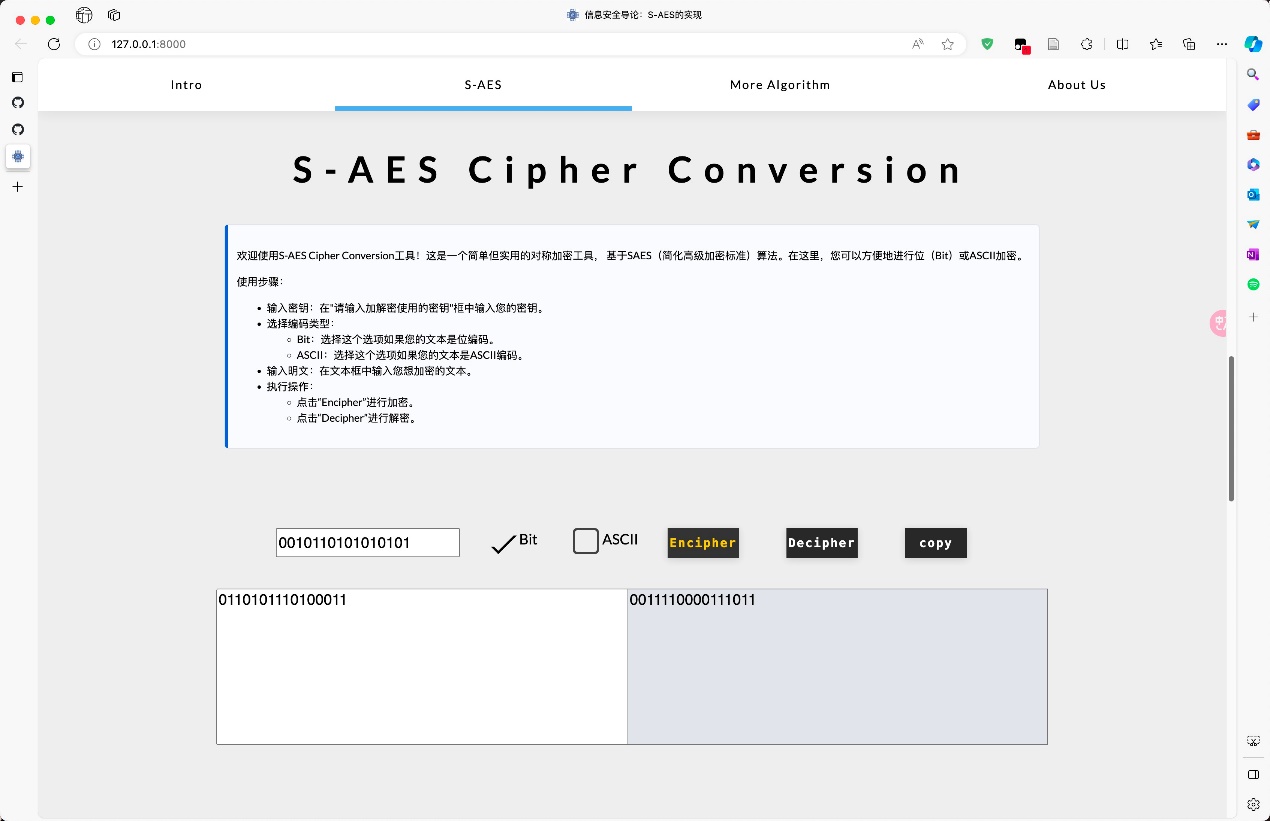
**测试目标：**验证不同人员编写的S-AES算法程序使用相同的密钥和明文时，能够得到相同的密文。

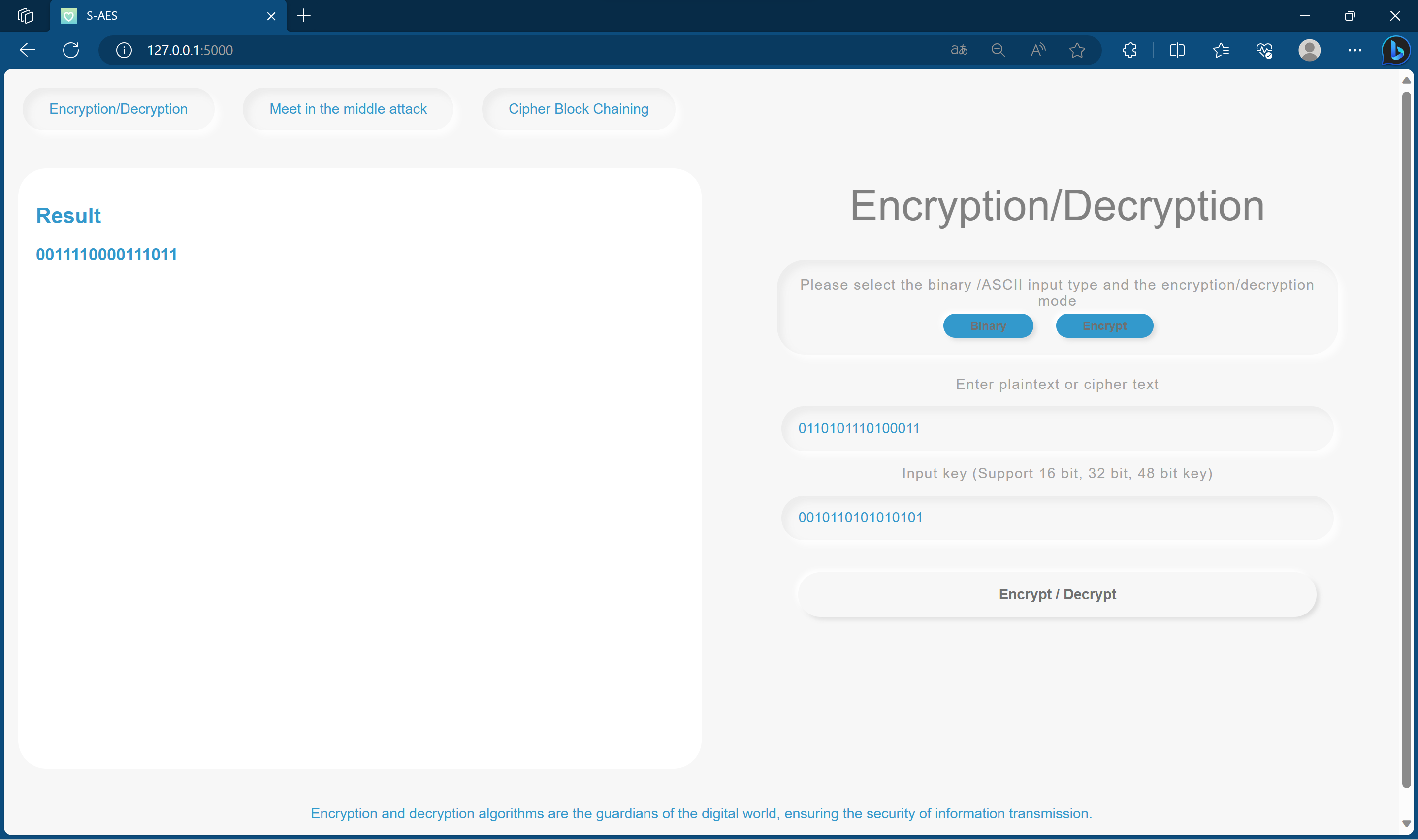
**测试结果：**经过交叉测试，不同编写的S-AES算法程序能够使用相同的密钥和明文得到相同的密文。

测试步骤和结果：

|  |  |  |
| --- | --- | --- |
| **测试输入** | **程序A输出** | **程序B输出** |
| 明文：  0110101110100011  密钥：0010110101010101 | 密文：0011110000111011 | 密文：0011110000111011 |

程序A：



程序B（当前）：  


**第3关：扩展功能**

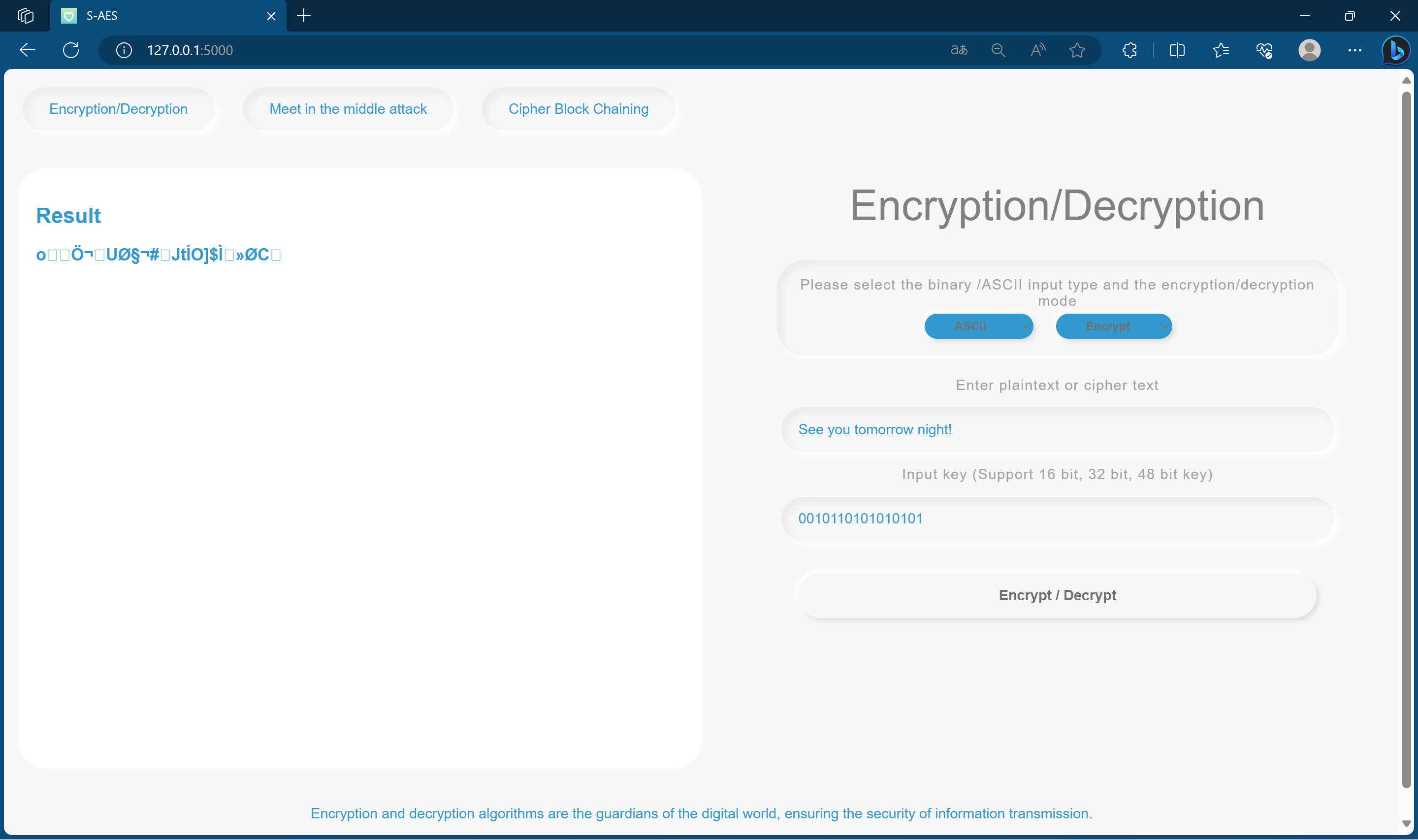
**测试目标：**验证S-AES算法程序是否能够处理ASCII编码字符串的加密和解密。

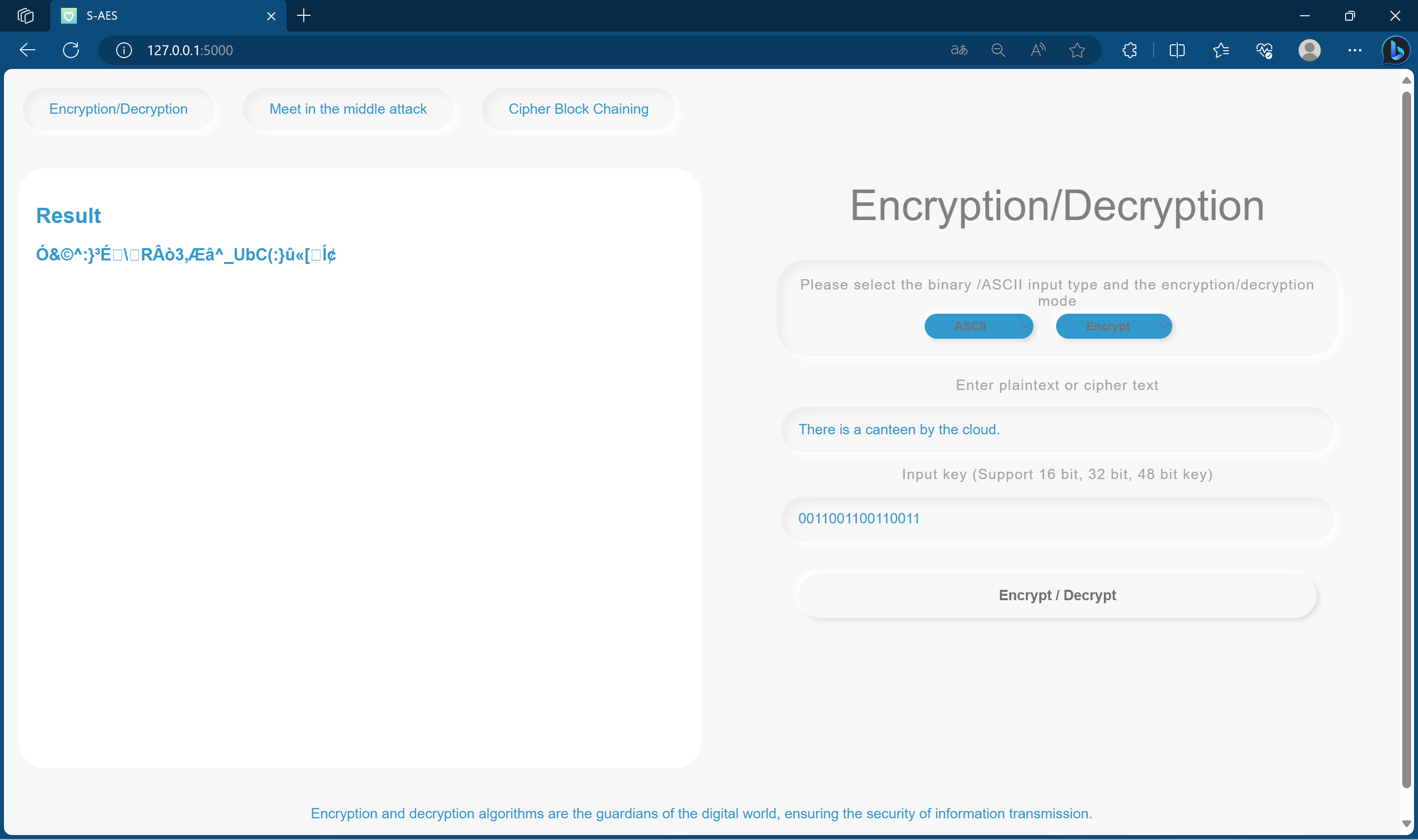
**测试结果：**经过测试，S-AES算法程序能够处理ASCII编码字符串的加密和解密，并产生相应的ASCII编码的密文。

测试步骤和结果：

（1）加密

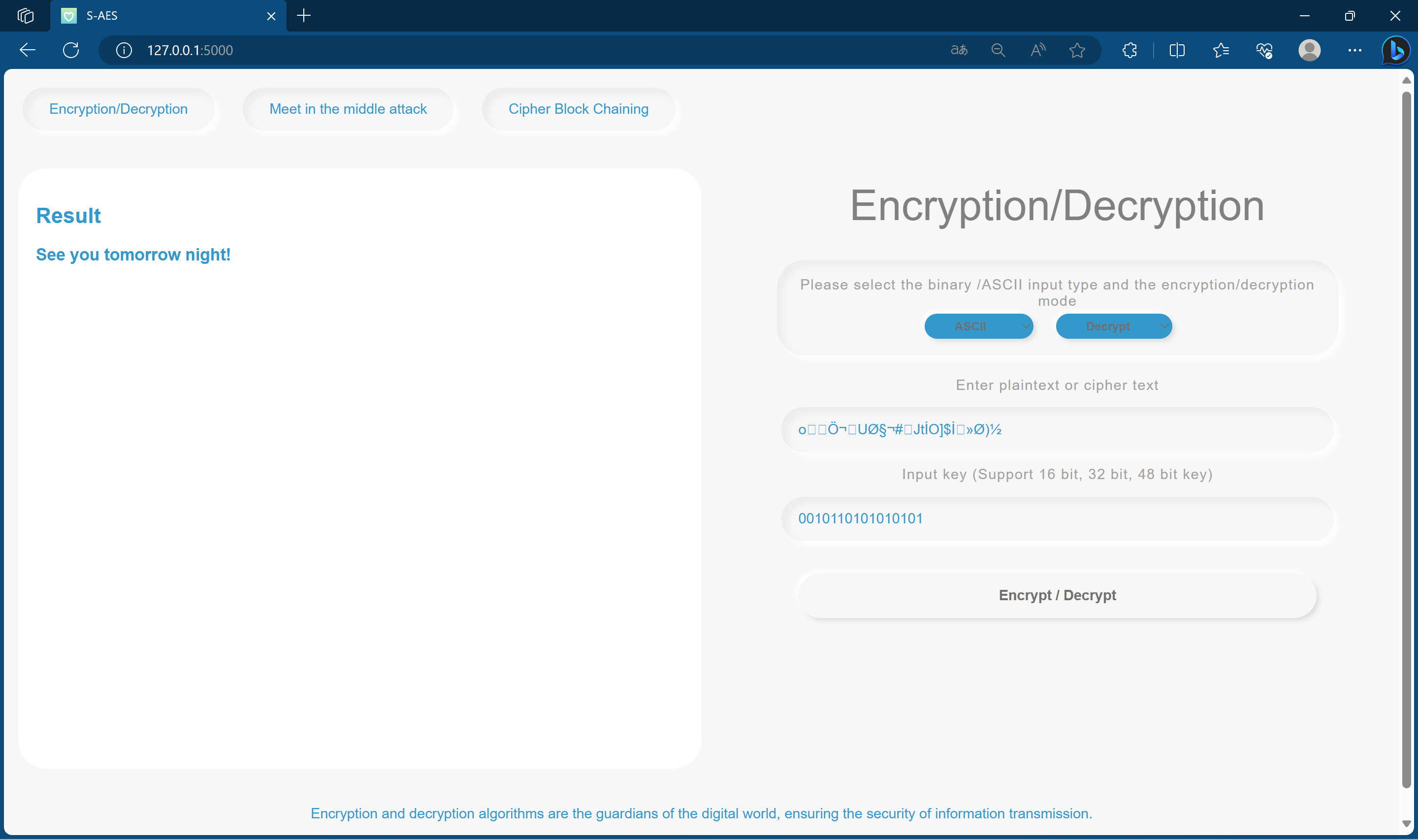
|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：See you tomorrow night! 密钥：0010110101010101 | 密文：oÖ¬UØ§¬#JtÍO]$Ì»Ø)½ |
| 明文：There is a canteen by the cloud. 密钥：0011001100110011 | 密文：Ó&©^:}³É\RÂò3,Æâ^\_UbC(:}û«[Í¢ |

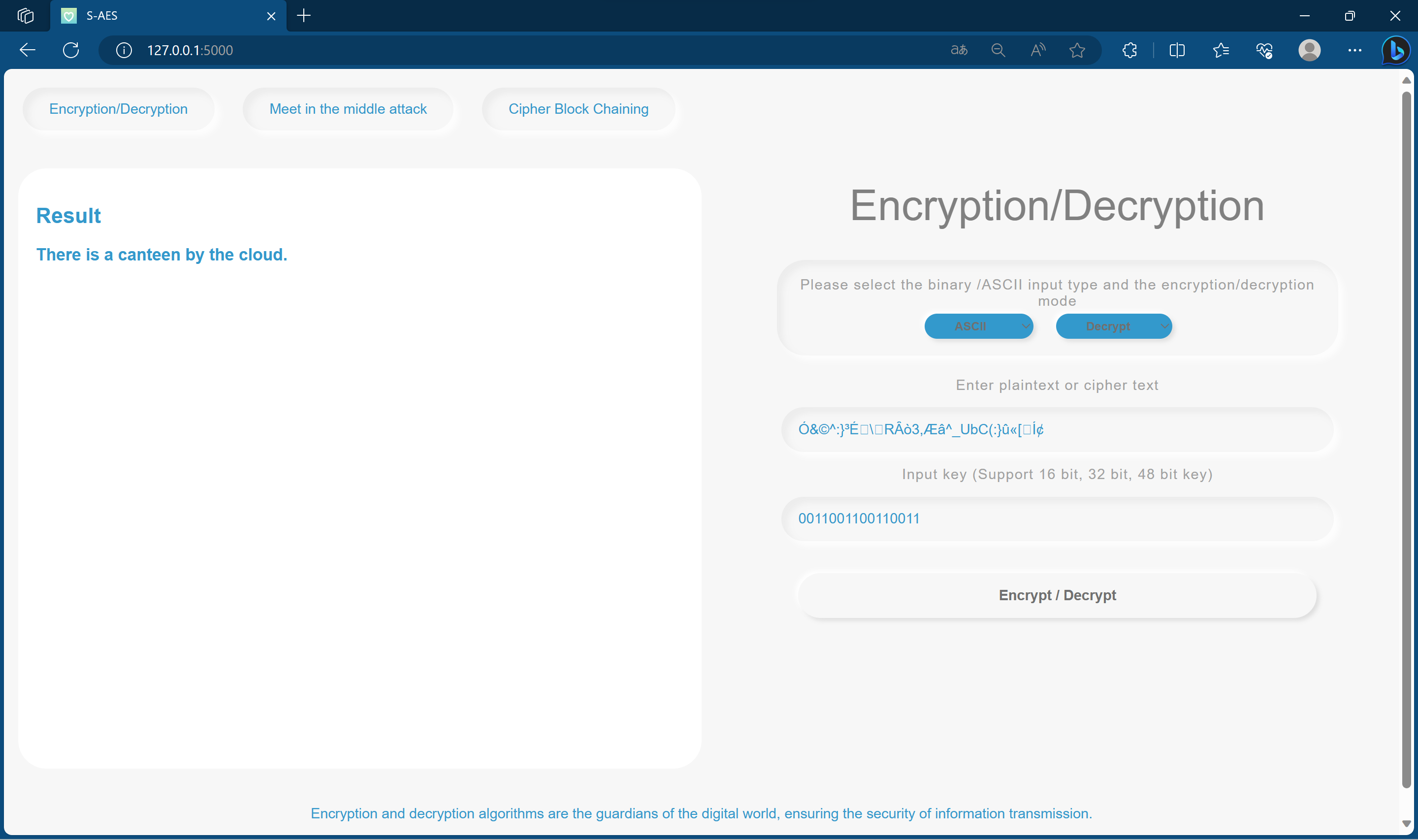




（2）解密

|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 密文：oÖ¬UØ§¬#JtÍO]$Ì»Ø)½ 密钥：0010110101010101 | 明文：See you tomorrow night! |
| 密文：Ó&©^:}³É\RÂò3,Æâ^\_UbC(:}û«[Í¢  密钥：0010110101010101 | 明文：There is a canteen by the cloud. |





**第4关：多重加密**

**4.1双重加密**

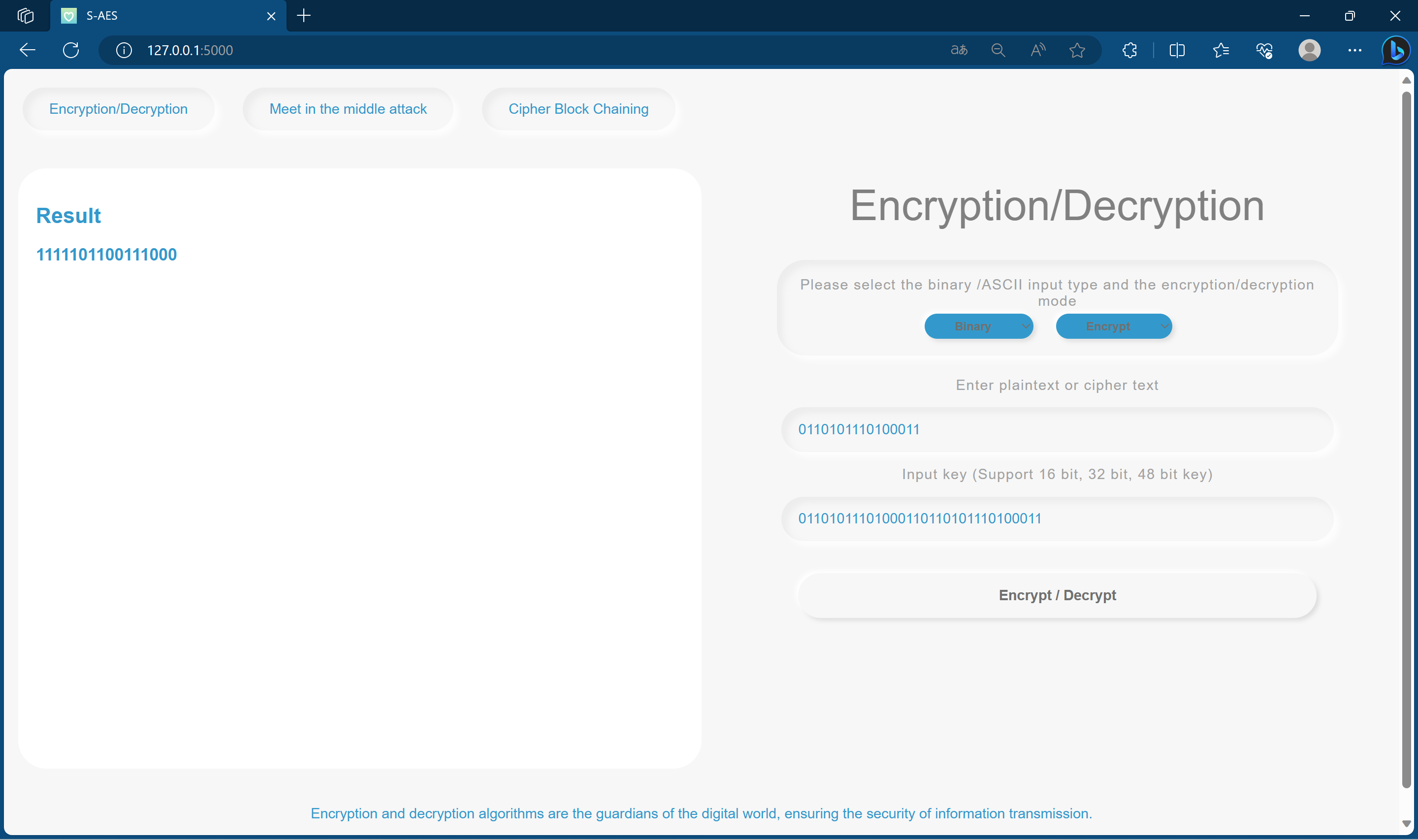
**测试目标：**将S-AES算法通过双重加密进行扩展，分组长度仍然是16 bits，但密钥长度为32 bits。

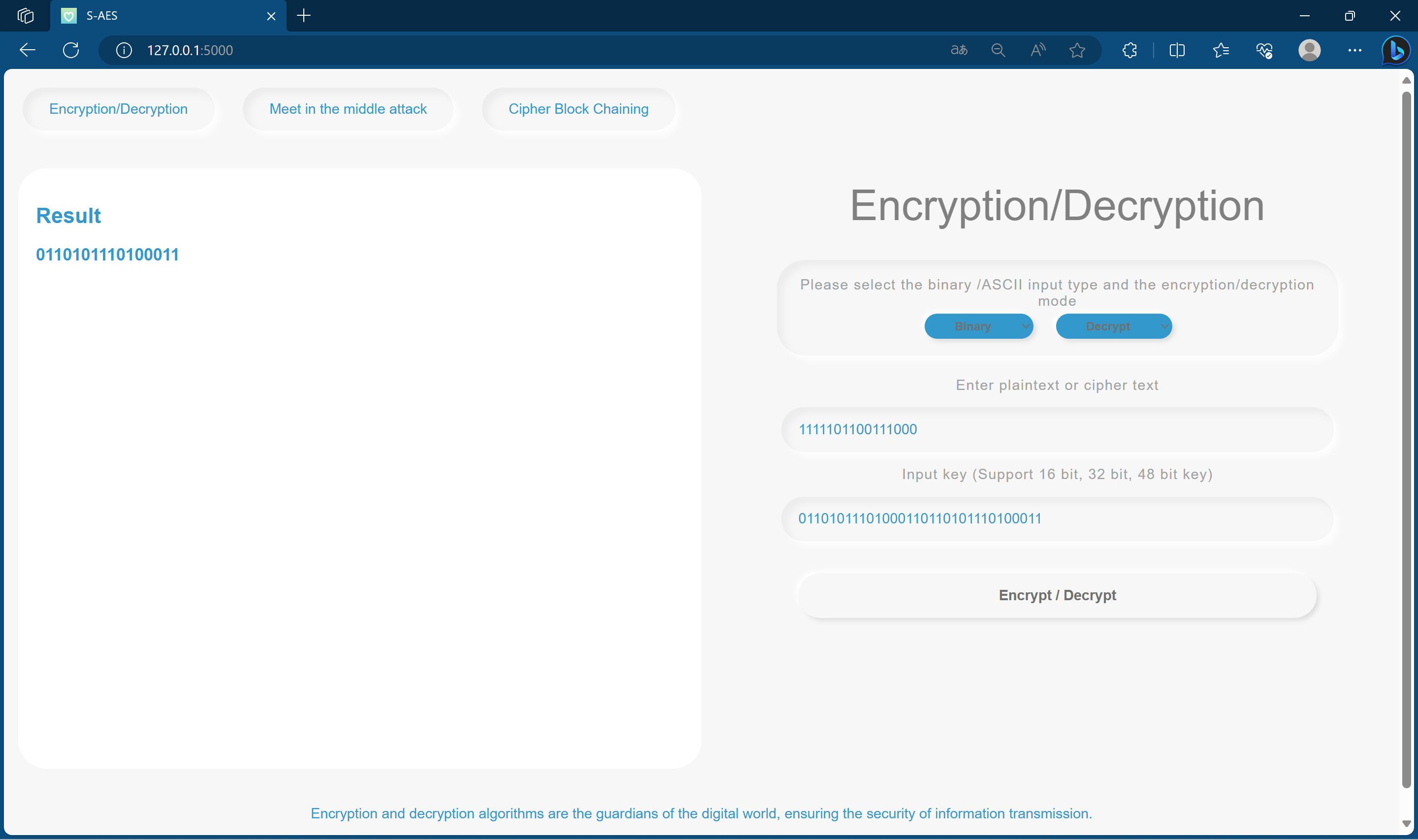
**测试结果：**经过测试，S-AES算法通过要求，实现对密钥长度进行判断，当密钥为32位时进行双重加密和解密。

测试步骤和结果：

* 二进制进行双重加密/解密

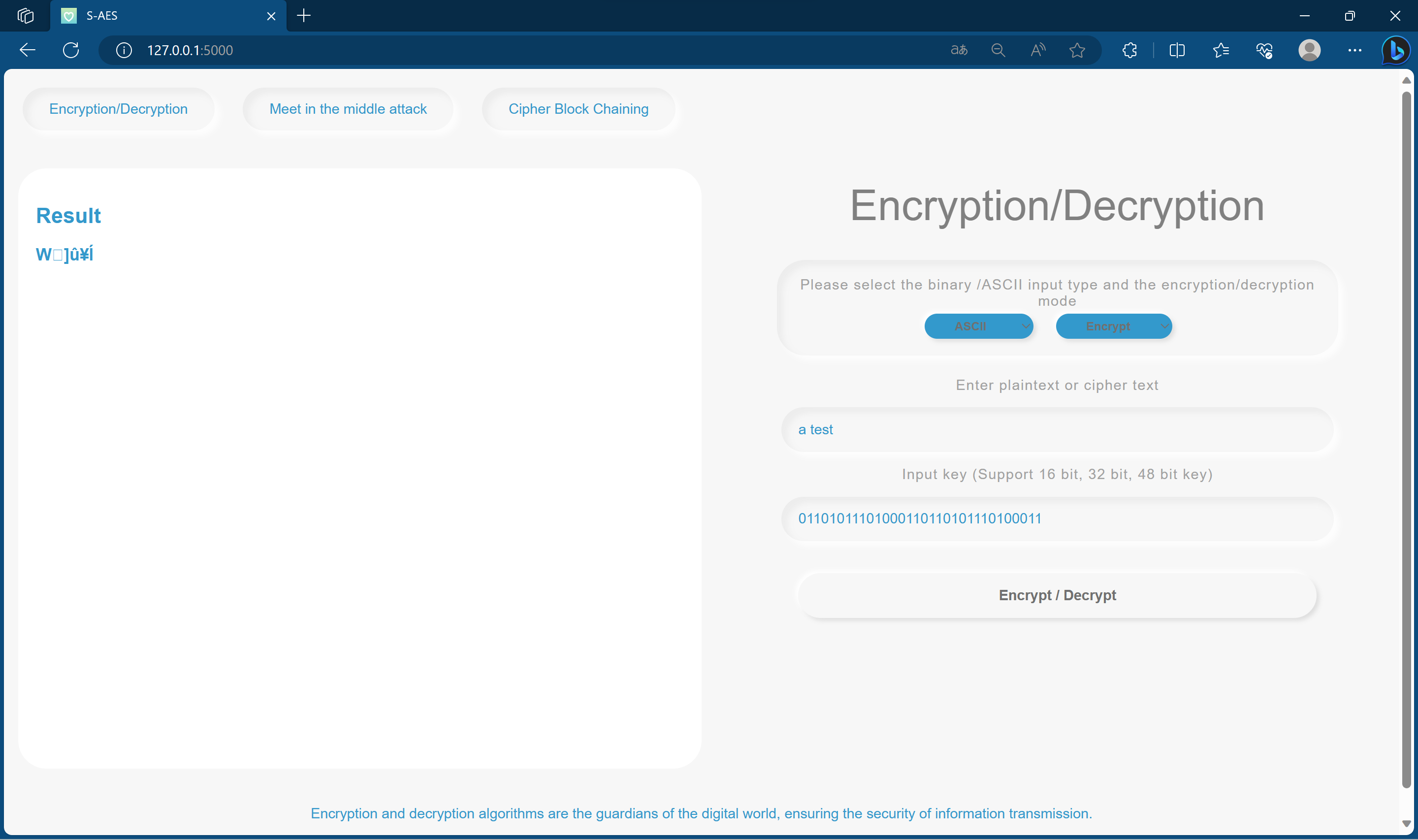
|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：0110101110100011 密钥：01101011101000110110101110100011 | 密文：1111101100111000 |
| 密文：1111101100111000  密钥：01101011101000110110101110100011 | 明文：0110101110100011 |

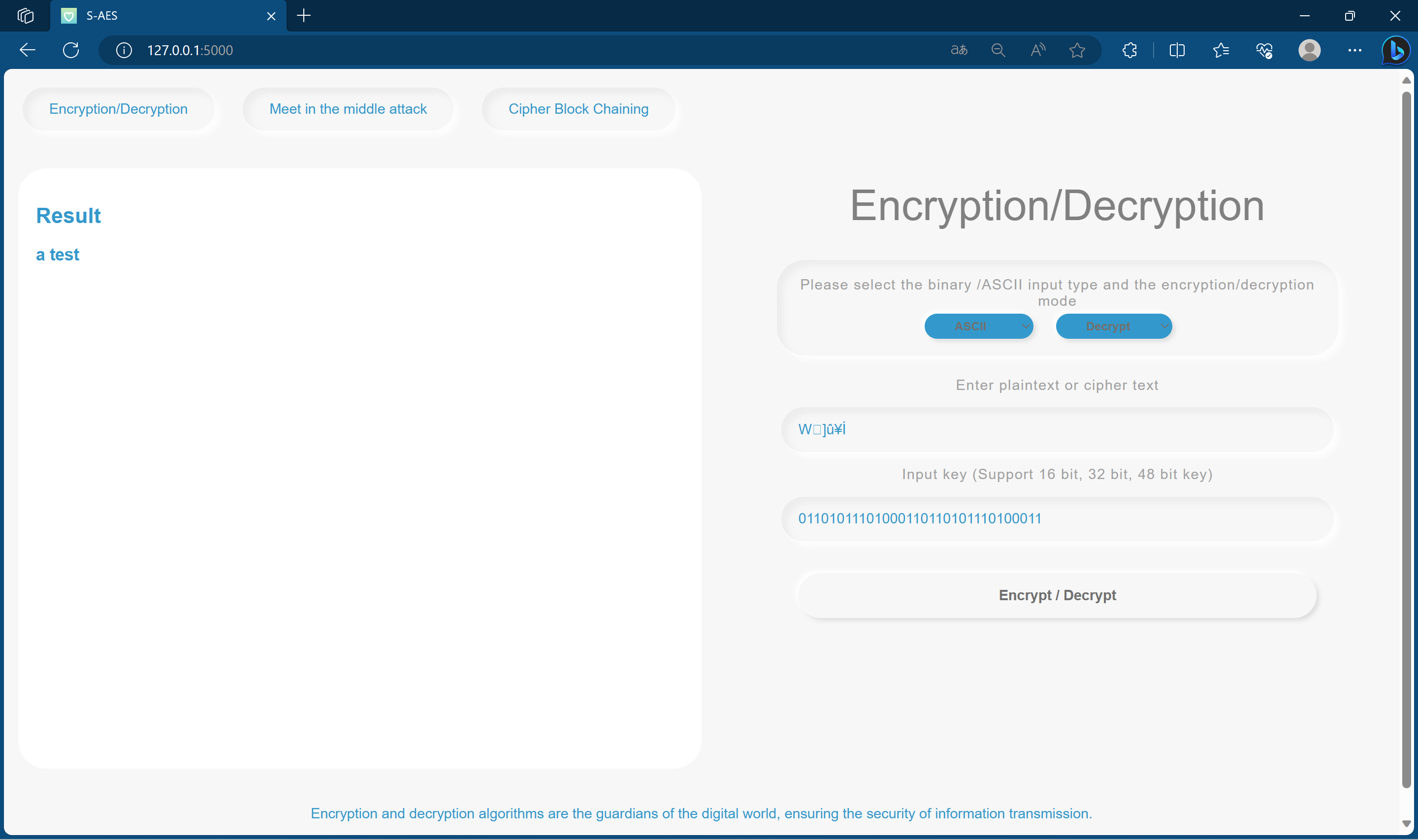




* ASCII字符串进行双重加密/解密

|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：a test 密钥：01101011101000110110101110100011 | 密文：W]û¥Í |
| 密文：W]û¥Í  密钥：01101011101000110110101110100011 | 明文：a test |





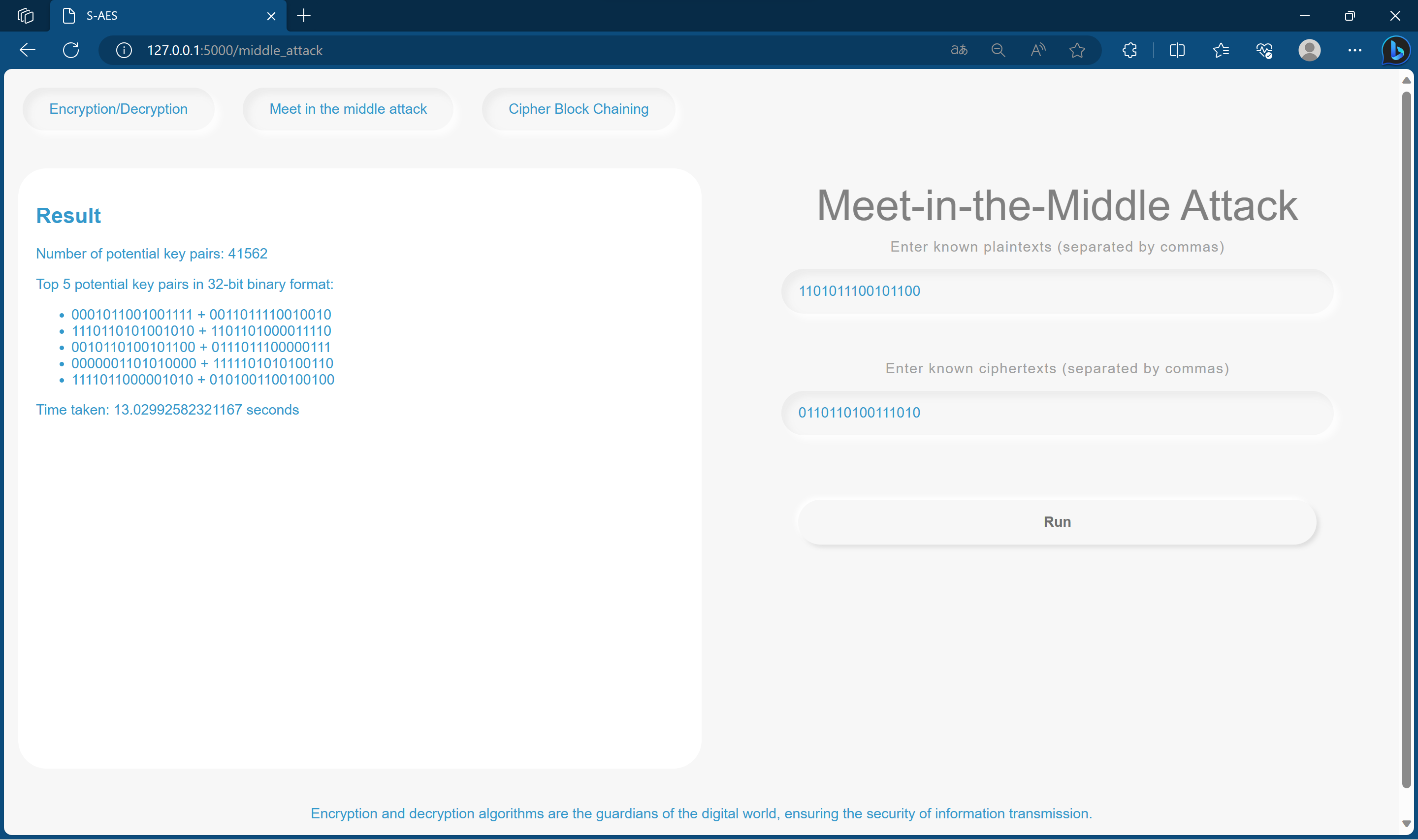
**4.2中途相遇攻击**

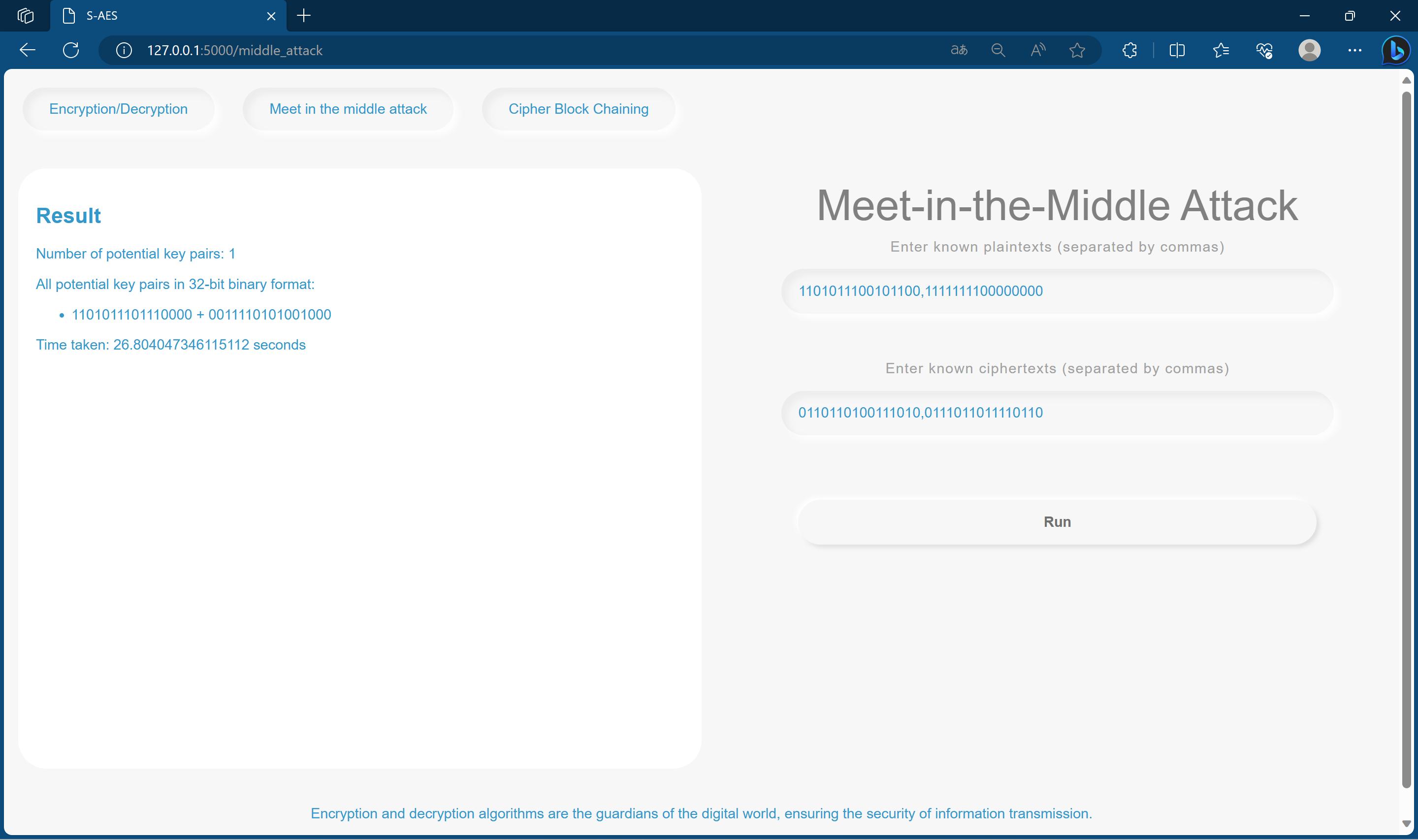
**测试目标：**假设已经找到了使用相同密钥的明、密文对(一对或多对)，尝试使用中间相遇攻击的方法找到正确的密钥Key(K1+K2)。

**测试结果：**经过测试， S-AES算法的双重加密机制能够在中途相遇攻击中找到所有可能的密钥集合。

测试步骤和结果：

|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 已知明文： 1101011100101100 已知密文：0110110100111010 | Number of potential key pairs: 41562  Top 5 potential key pairs in 32-bit binary format:  密钥：  0001011001001111 + 0011011110010010  1110110101001010 + 1101101000011110  0010110100101100 + 0111011100000111  0000001101010000 + 1111101010100110  1111011000001010 + 0101001100100100 |
| 已知明文：1101011100101100,1111111100000000  已知密文：0110110100111010,0111011011110110 | Number of potential key pairs: 1 密钥：  1101011101110000+0011110101001000 |





**4.3三重加密（k1,k2,k3）**

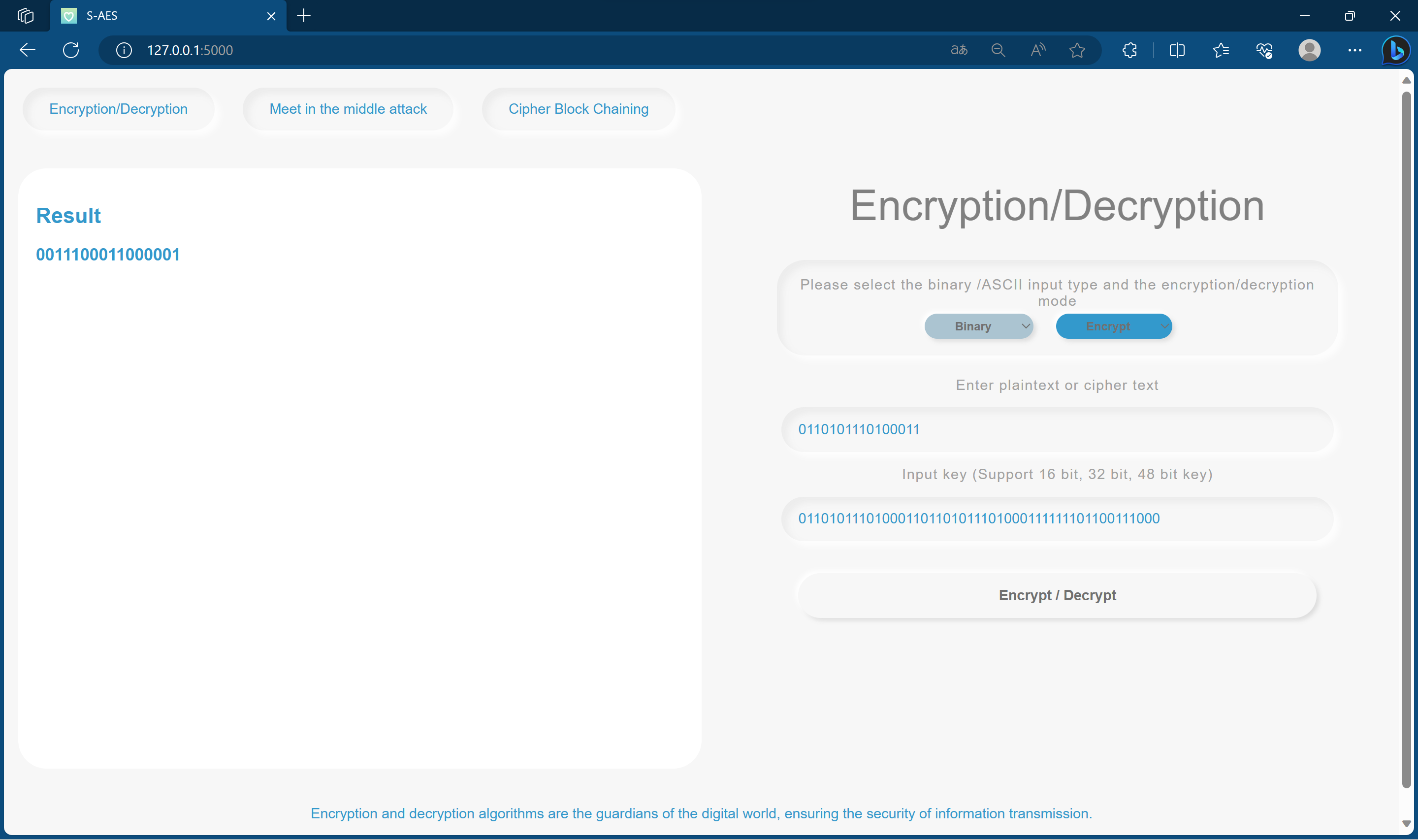
**测试目标：**使用48bits(K1+K2+K3)的模式进行三重加解密。

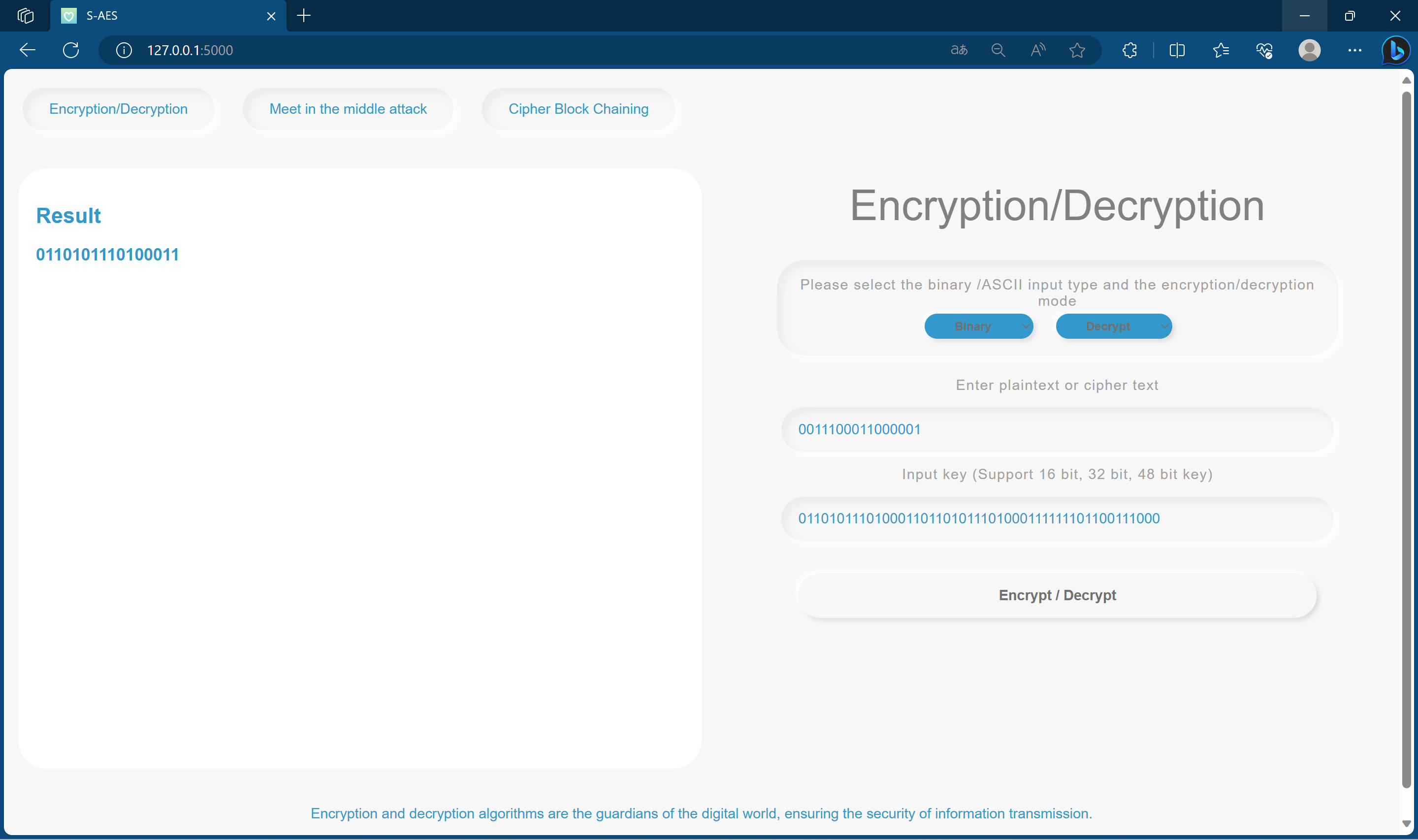
**测试结果：**经过测试，S-AES算法通过要求，实现对密钥长度进行判断，当密钥为48 位时进行三重加密和解密。

测试步骤和结果：

* 二进制进行双三重加密/解密

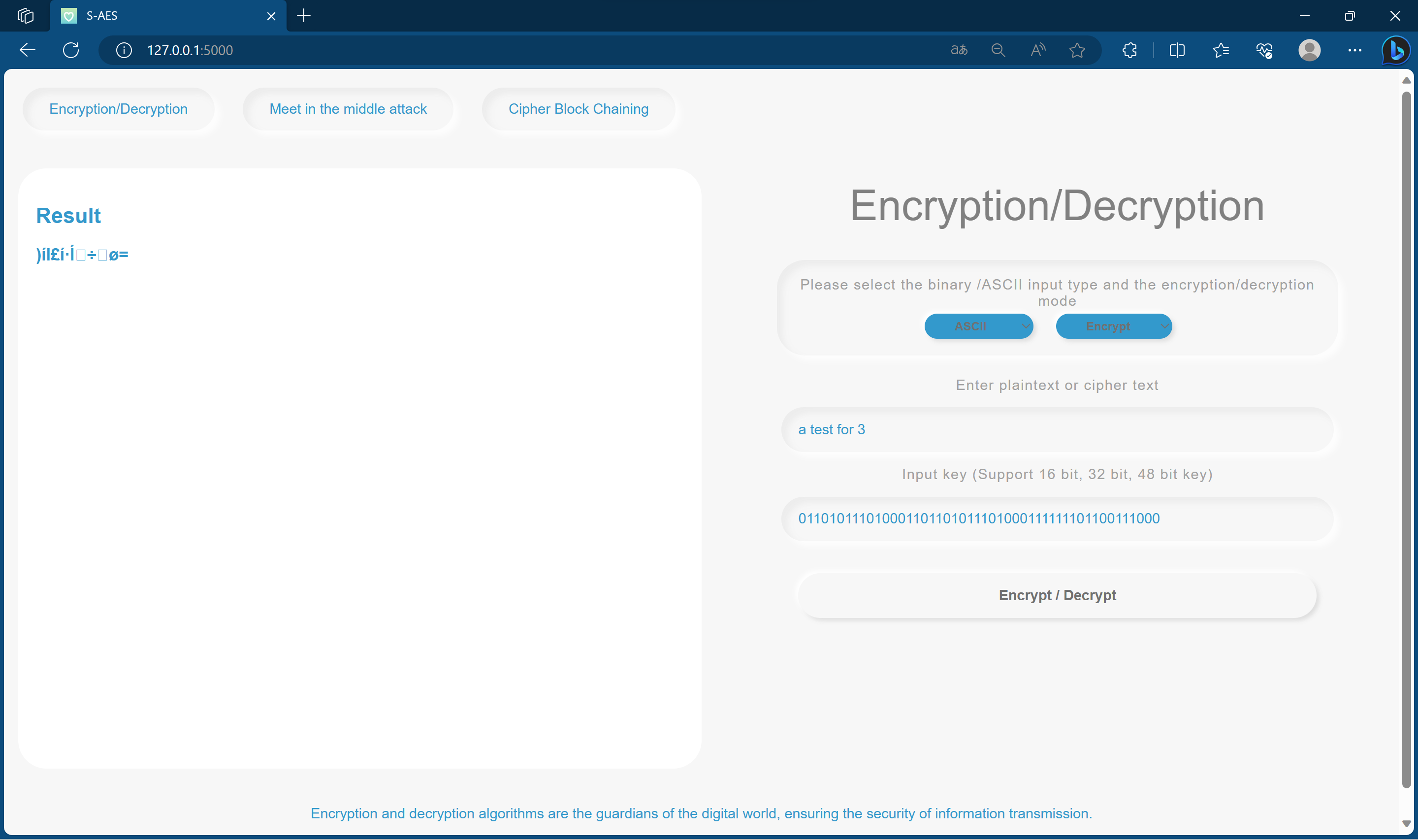
|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：  0110101110100011 密钥：  011010111010001101101011101000111111101100111000 | 密文：  0011100011000001 |
| 密文：  0011100011000001  密钥：  011010111010001101101011101000111111101100111000 | 明文：  0110101110100011 |

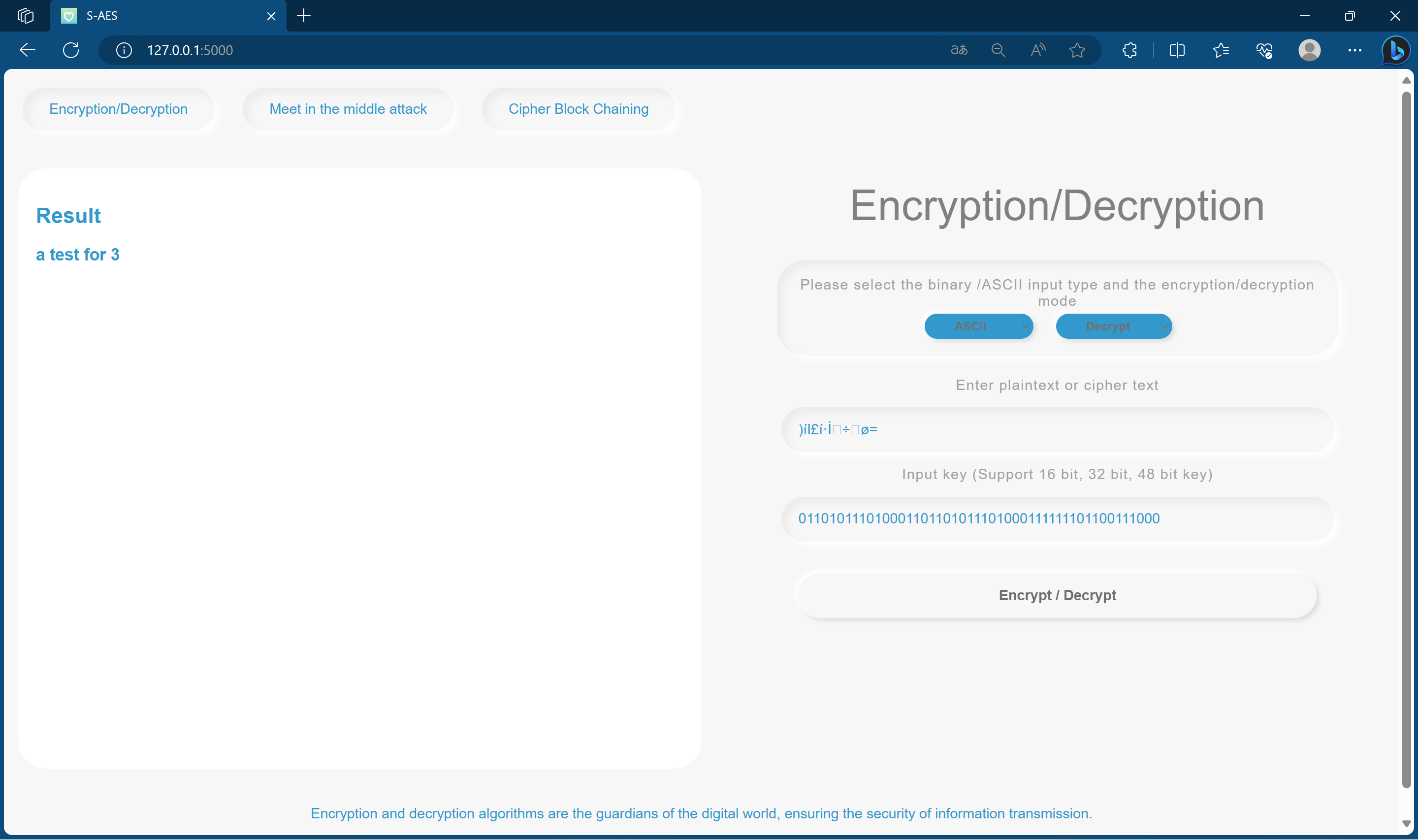




* ASCII字符串进行三重加密/解密

|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：a test for 3 密钥：01101011101000110110101110100011 | 密文：)íI£í·Í÷ø= |
| 密文：)íI£í·Í÷ø=  密钥：01101011101000110110101110100011 | 明文：a test for 3 |





**第5关：工作模式**

**测试目标：**基于S-AES算法，使用密码分组链(CBC)模式对较长的明文消息进行加密。注意初始向量(16 bits) 的生成，并需要加解密双方共享。

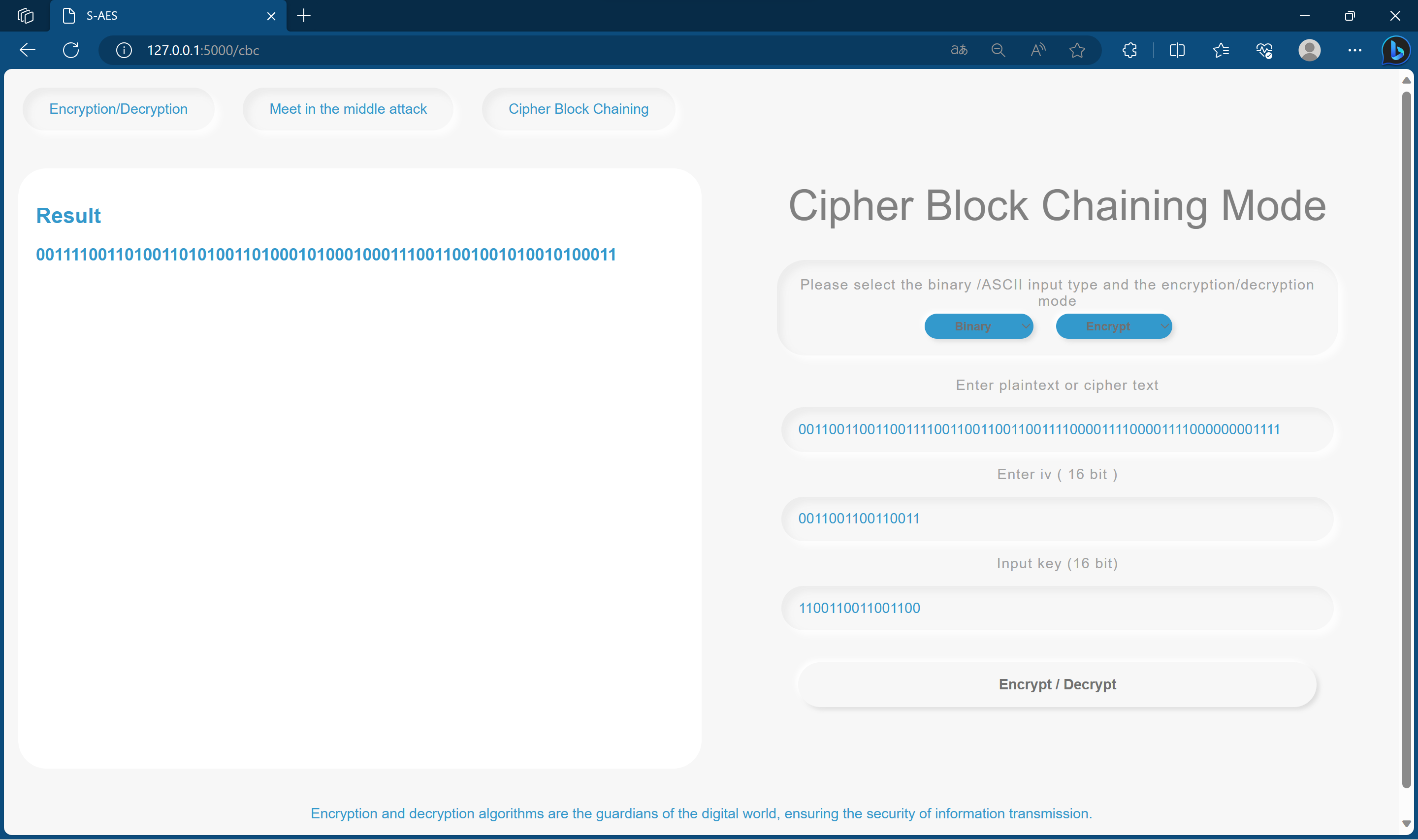
在CBC模式下进行加密，并尝试对密文分组进行替换或修改，然后进行解密，请对比篡改密文前后的解密结果。

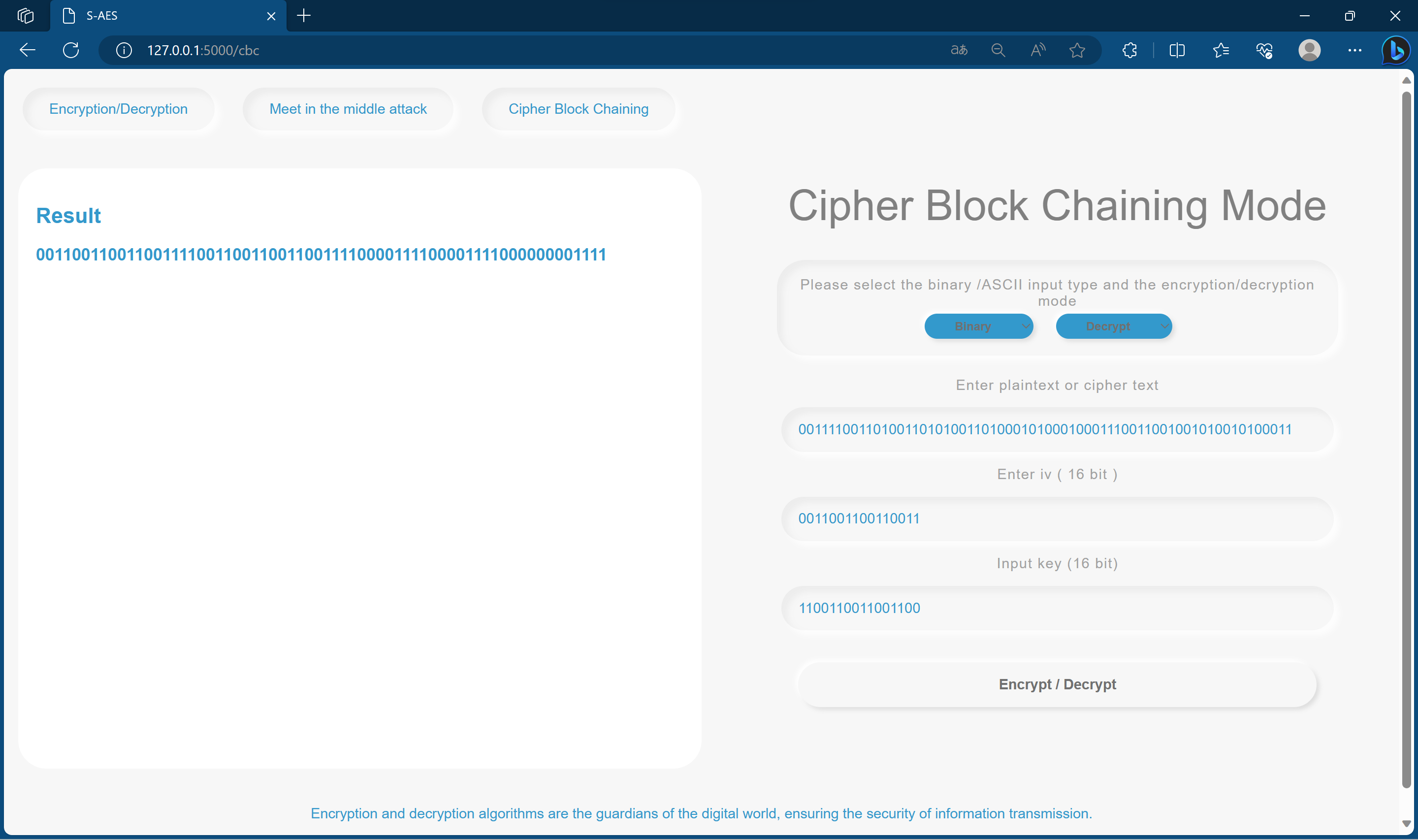
**测试结果：**经过测试，已实现通过密码分组链(CBC)模式对较长的明文消息进行加密、解密，并对比了篡改密文前后的解密结果。

测试步骤和结果：

* 二进制进行CBC模式加密/解密

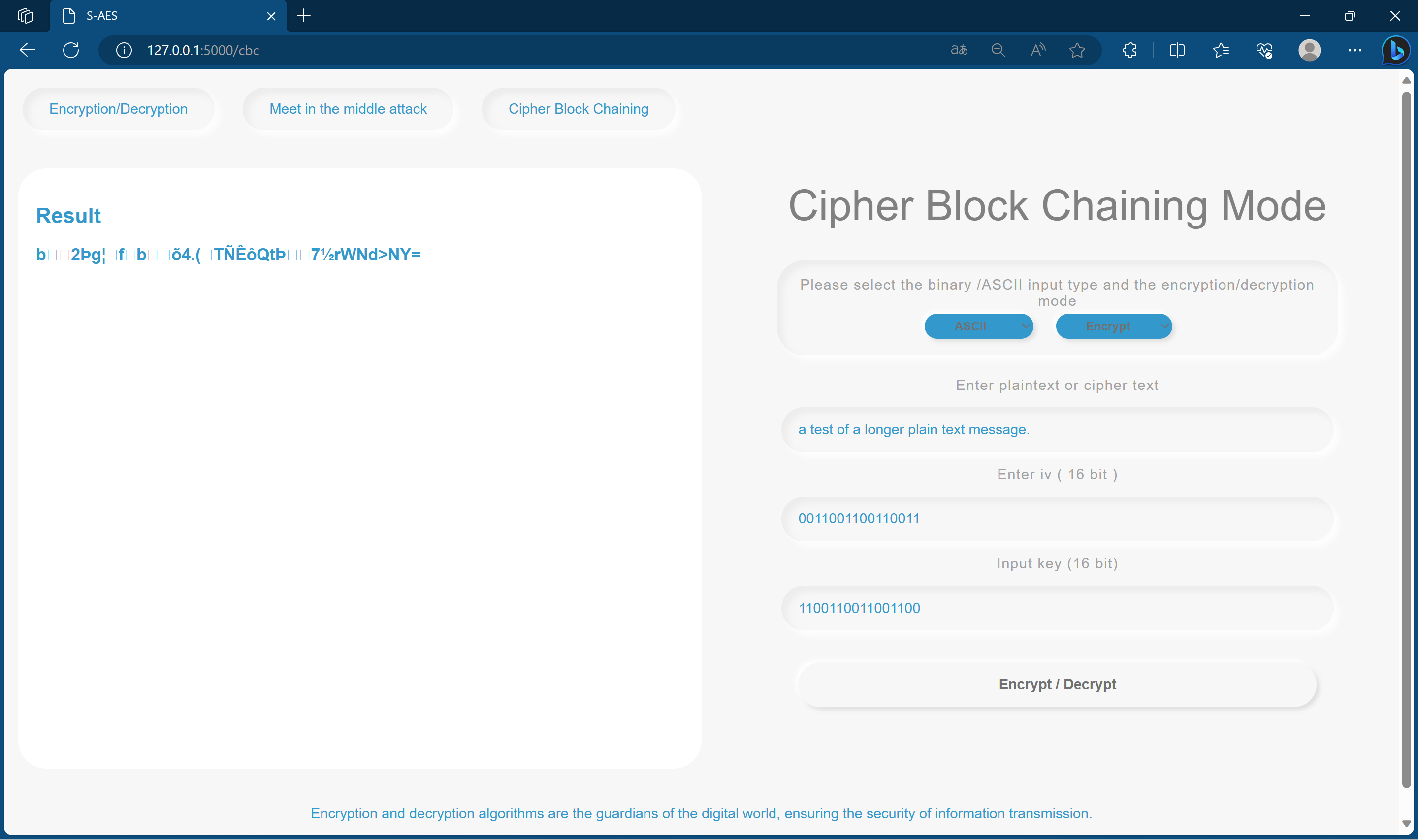
|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文（48位）：  0011001100110011  1100110011001100  1111000011110000  1111000000001111  初始向量iv: 0011001100110011 密钥：  1100110011001100 | 密文：  0011110011010011  0101001101000101  0001000111001100  1001010010100011 |
| 密文：  0011110011010011  0101001101000101  0001000111001100  1001010010100011  初始向量iv: 0011001100110011 密钥：  1100110011001100 | 明文：  0011001100110011  1100110011001100  1111000011110000  1111000000001111 |

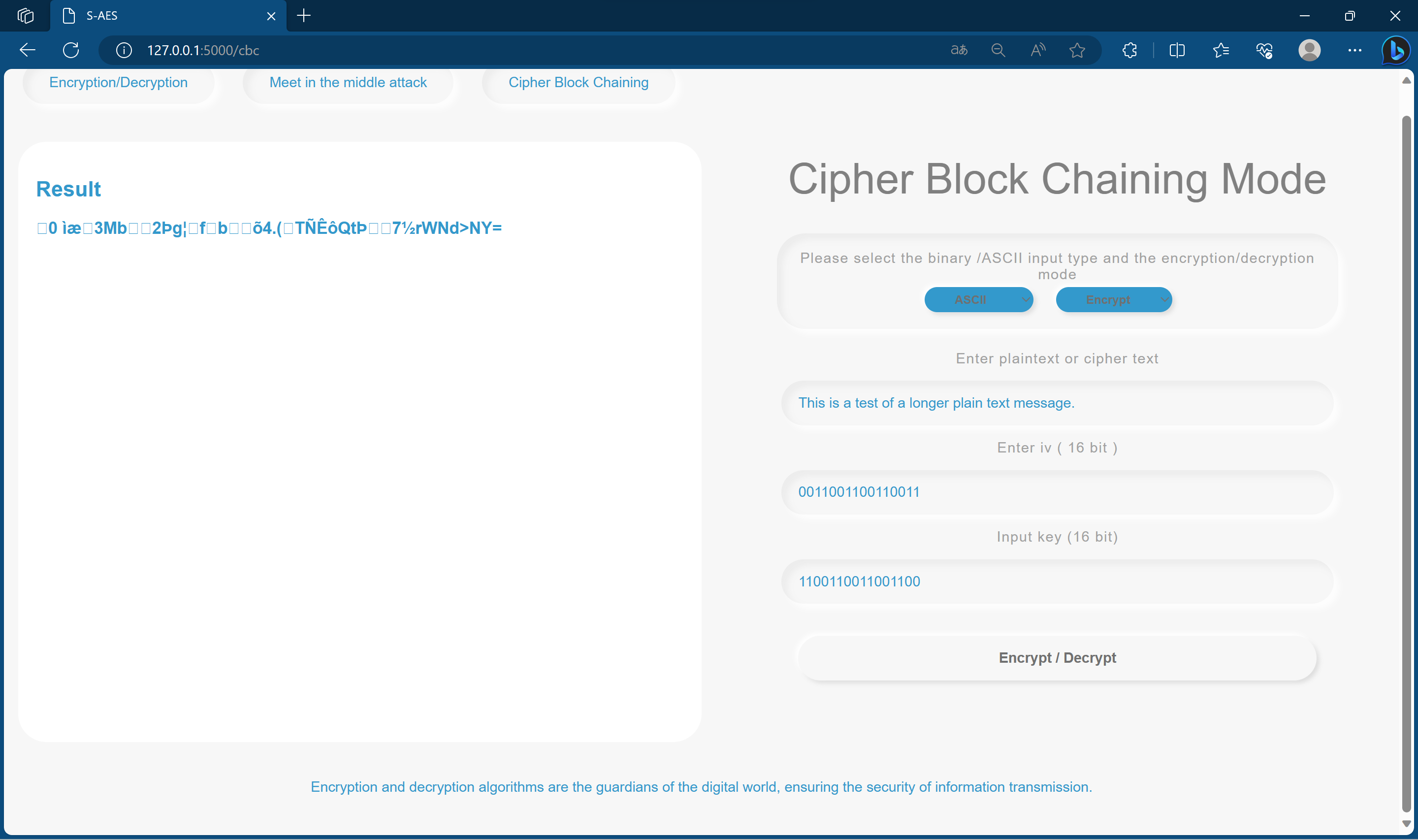




* ASCII字符串进行CBC模式加密/解密

|  |  |
| --- | --- |
| **测试输入** | **输出** |
| 明文：  a test of a longer plain text message.  初始向量iv: 0011001100110011 密钥：  1100110011001100 | 密文：  b2Þg¦fbõ4.(TÑÊôQ­tÞ  7½rWNd>NY= |
| 密文：  b2Þg¦fbõ4.(TÑÊôQ­tÞ7½rWNd>NY=  初始向量iv: 0011001100110011 密钥：  1100110011001100 | 明文：a test of a longer plain text message. |

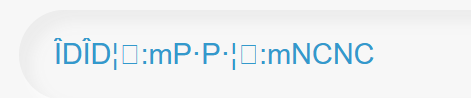




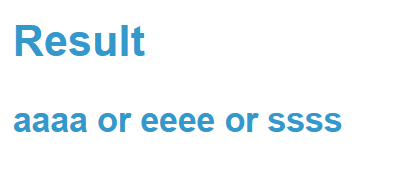
* 测试与分析：在CBC模式下进行加密，并尝试对密文分组进行替换或修改，然后进行解密，请对比篡改密文前后的解密结果。

### 密文：ÎDÎD¦:mp·p·¦:mNdNd

**替换后密文01：**



篡改密文前后的解密结果：

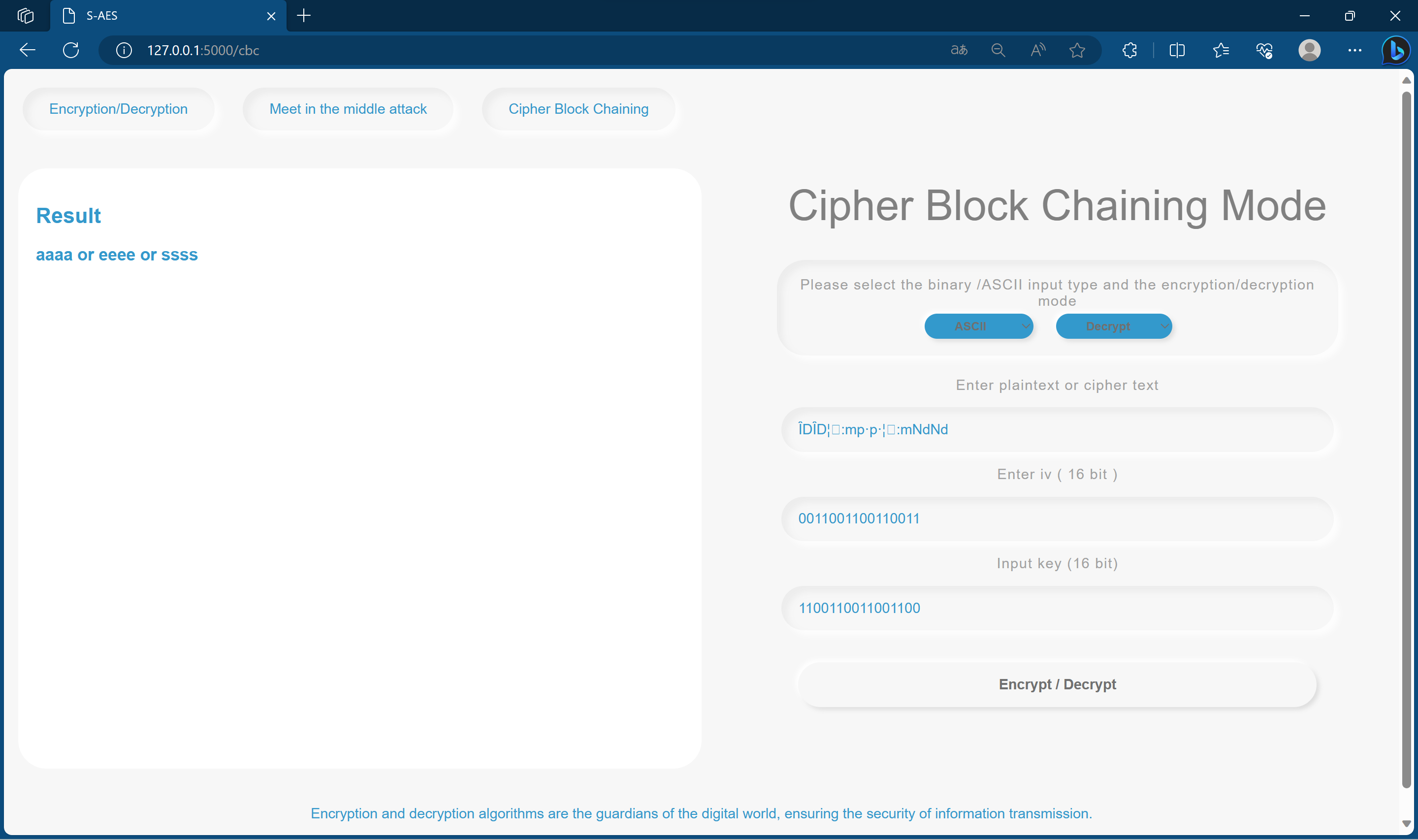
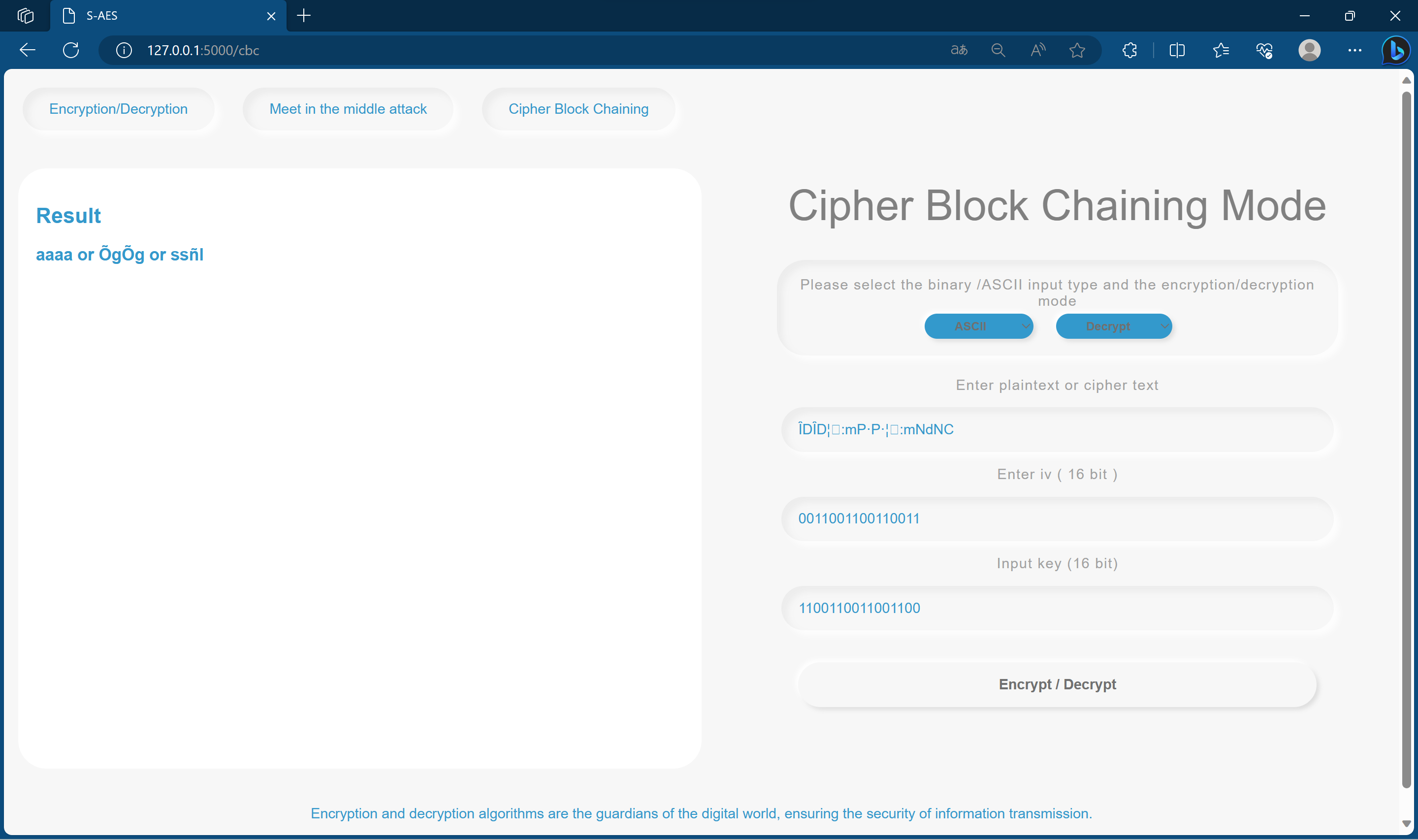
前：  


后：  


**分析：**

由对比结果可知，在CBC模式下进行加密后得到的密文，经过篡改后，只会影响其所在的分组，因为当前分组长度设定为16bit，故输入类型为ASCII时，每两个字符为一组明文块进行处理，故改动后以两个字符为单位发生变化（如ss-> ñl）。

详细运行图：

* 验证：仅修改初始向量iv的值，实现相同明文和密钥得到不同密文

