

Лабораторная работа № 7

Элементы криптографии. Однократное гаммирование

Сухарев Кирилл

Цель работы

Освоить на практике применение режима однократного гаммирования

Генерация ключа

```
import random
import string

def generate_key(length):
    return ''.join(random.choice(string.ascii_letters + string.digits) for _ in range(length))
```

Шифрование

```
def single_gamming(message, key):  
    return ''.join(chr(ord(m) ^ ord(k)) for m, k in zip(message, key))  
  
def decrypt(encrypted, key):  
    return single_gamming(encrypted, key)  
  
def find_key(message, encrypted):  
    return single_gamming(message, encrypted)
```

Проверка работоспособности

Проверка работоспособности

```
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Установите последнюю версию PowerShell для новых функций и улучшения! https://aka.ms/PSWindows

PS D:\Information Security\lab7> python .\app.py
Enter message to encrypt: Hello world!
Key: pKkQ9thNDzr8 | 70 4b 6b 51 39 74 68 4e 44 7a 72 38
Encrypted message: 8.=VT!6—↓ | 38 2e 07 3d 56 54 1f 21 36 16 16 19
Decrypted message: Hello world!

Key: pKkQ9tz@Db~8 | 70 4b 6b 51 39 74 7a 40 44 62 7e 38
Decrypted wrong message: Hello earth!
PS D:\Information Security\lab7> |
```

Листинг программы

```
import random
import string

def generate_key(length):
    return ''.join(random.choice(string.ascii_letters + string.digits) \
                    for _ in range(length))

def to_hex(input_string):
    return ''.join('{:02x}'.format(ord(symbol)) \
                    for symbol in input_string)

def beuatify(text):
    return "{} | {}".format(text, to_hex(text))
```

```
def single_gamming(message, key):  
    return ''.join(chr(ord(m) ^ ord(k)) for m, k in zip(message, key))  
  
def decrypt(encrypted, key):  
    return single_gamming(encrypted, key)  
  
def find_key(message, encrypted):  
    return single_gamming(message, encrypted)
```

```
message = input("Enter message to encrypt: ")

key = generate_key(len(message))
print("Key:", beuatify(key))

encrypted = single_gamming(message, key)
print("Encrypted message:", beuatify(encrypted))

print("Decrypted message:", decrypt(encrypted, key))
```

```
wrong_message = "Hello earth!"  
wrong_key = find_key(encrypted, wrong_message)  
print("\nKey:", beuatify(wrong_key))  
  
print("Decrypted wrong message:", decrypt(encrypted, wrong_key))
```