Цель работы

Освоить на практике применение режима однократного гаммирования
на примере кодирования различных исходных текстов одним ключом

Порядок выполнения работы

```
import os
# XOR две строки
def xor_strings(s1, s2):

    return bytes([a ^ b for a, b in zip(s1, s2)])

# ключ
key = bytes.fromhex('050C177F0E4E37D29410092E2257FFC80BB27054')

# Исходное сообщение
p1 = 'НаВашисходящийот1204'.encode('utf-8')
p2 = 'ВСеверныйфилиалБанка'.encode('utf-8')

# Цикл XOR-обработки
# Шифрование или дешифрование сообщений с помощью ключа
def encrypt_decrypt(message, key):

    key_length = len(key)
    message_length = len(message)
    extended_key = (key * (message_length // key_length)) + key[:message_length %
key_length]
    return xor_strings(message, extended_key)

# Процесс шифрования
c1 = encrypt_decrypt(p1, key)
c2 = encrypt_decrypt(p2, key)

# Процесс расшифровки
p1_decrypted = encrypt_decrypt(c1, key).decode('utf-8')
p2_decrypted = encrypt_decrypt(c2, key).decode('utf-8')

# результаты печати
print(f"C1: {c1.hex()}")
print(f"C2: {c2.hex()}")
print(f"P1 После расшифровки: {p1_decrypted}")
print(f"P2 После расшифровки: {p2_decrypted}")
```

Метод для чтения двух текстов без получения ключа

Принцип работы

1. Известные данные: У атакующего есть два зашифрованных текста (C1) и (C2).
2. Цель: Прочитать оригинальные тексты (P1) и (P2) без знания ключа.

Теоретические основы

Предположим, что у нас есть следующие формулы: [C1 = P1 ⊕ K] [C2 = P2 ⊕ K]

Если мы сложим эти две формулы по модулю 2 (используем операцию XOR), получим: [C1 ⊕ C2 = (P1 ⊕ K) ⊕ (P2 ⊕ K) = P1 ⊕ P2]

Таким образом, атакующий может вычислить (P1 ⊕ P2) как (C1 ⊕ C2).

Конкретные шаги

1. Вычисление (C1 ⊕ C2): [C1 ⊕ C2 = (P1 ⊕ K) ⊕ (P2 ⊕ K) = P1 ⊕ P2]
2. Использование известной информации:

- Если атакующий знает часть одного из текстов, например (P1), он может использовать эту информацию для восстановления (P2).
- Если атакующий знает часть (P1), обозначим её как (P1*{\text{known}}), можно вычислить соответствующую часть (P2): [P2*{\text{part}} = (C1 ⊕ C2) ⊕ P1_{\text{known}}]

3. Итеративное восстановление:

- Постепенно заменяйте известные части (P1) и используйте их для восстановления (P2).
- Повторяйте этот процесс до полного восстановления (P2).

4. Использование языковых особенностей:

- Используйте известные языковые особенности (например, распространённые слова и фразы) для предположений о содержании (P1) и (P2).
- Сравнивайте результаты (C1 ⊕ C2) с известными языковыми паттернами для пошагового восстановления (P1) и (P2).

5. Статистический анализ:

- Используйте статистические методы для анализа результата (C1 ⊕ C2) и определения наиболее вероятных слов и фраз.
- Многократно повторяйте процесс, чтобы постепенно приблизиться к правильным (P1) и (P2).

Контрольные вопросы

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Ответ:

Конкретные шаги

1. **Вычисление (C1 ⊕ C2)**:
   [C1 ⊕ C2 = (P1 ⊕ K) ⊕ (P2 ⊕ K) = P1 ⊕ P2]
2. **Использование известной информации**:
   - Если атакующий знает часть одного из текстов, например (P1), он может использовать эту информацию для восстановления (P2).
   - Если атакующий знает часть (P1), обозначим её как (P1*{text{known}}), можно вычислить соответствующую часть (P2):*
     [P2{text{part}} = (C1 ⊕ C2) ⊕ P1_{text{known}}]

2. Что будет при повторном использовании ключа при шифровании текста?

Ответ:

Конкретные последствия

1. **Линейная зависимость**:
   - Каждый шифртекст становится линейно зависимым от других шифртекстов: (C1 ⊕ C2 = P1 ⊕ P2).
2. **Предсказуемость**:
   - Атакующий может использовать известный текст для предсказания других текстов.
3. **Статистический анализ**:
   - Атакующий может использовать статистические методы для анализа шифртекстов и предположения содержания других текстов.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

Ответ:

Конкретные шаги

1. **Расширение ключа**:
   - Расширить ключ до длины открытого текста.
2. **Шифрование текстов**:
   - Произвести операцию XOR для каждого открытого текста: (C1 = P1 ⊕ K), (C2 = P2 ⊕ K).
3. **Результат**:
   - Этот метод прост в реализации, но вводит линейную зависимость между шифртекстами, что снижает безопасность.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

Ответ:

Недостатки

1. **Снижение безопасности**:
    - Шифртексты становятся линейно зависимыми, что позволяет атакующему использовать известный текст для предсказания других текстов.
2. **Предсказуемость**:
    - Повторное использование ключа увеличивает предсказуемость шифртекстов.
3. **Статистический анализ**:
    - Атакующий может использовать статистические методы для анализа шифртекстов и предположения содержания других текстов.
4. **Управление ключами**:
    - Ключ должен быть строго конфиденциальным и использоваться только один раз, иначе безопасность значительно снижается.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Ответ:

Преимущества

1. **Упрощение управления ключами**:
    - Необходимо управлять только одним ключом, что уменьшает сложность управления ключами.
2. **Простота реализации**:
    - Шифрование и дешифрование относительно просты в реализации.
3. **Экономия ресурсов**:
    - Не требуется генерировать разные ключи для каждого открытого текста, что экономит ресурсы.

ВЫВОДЫ

Освол на практике применение режима однократного гаммирования
на примере кодирования различных исходных текстов одним ключом