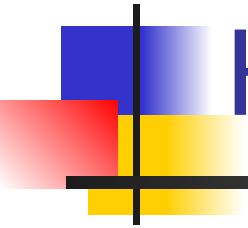


Криптография

Основные понятия.
Симметричные криптосистемы

Определение криптографии

- Криптография – совокупность методов преобразования данных, направленных на то, чтобы защитить эти данные, сделав их бесполезными для незаконных пользователей
- Для обеспечения безопасности данных необходимо поддерживать 3 основные функции:
  - Защита конфиденциальности передаваемых или хранимых в памяти данных
  - Подтверждение целостности и подлинности данных
  - Аутентификация абонентов при входе в систему и при установлении соединения

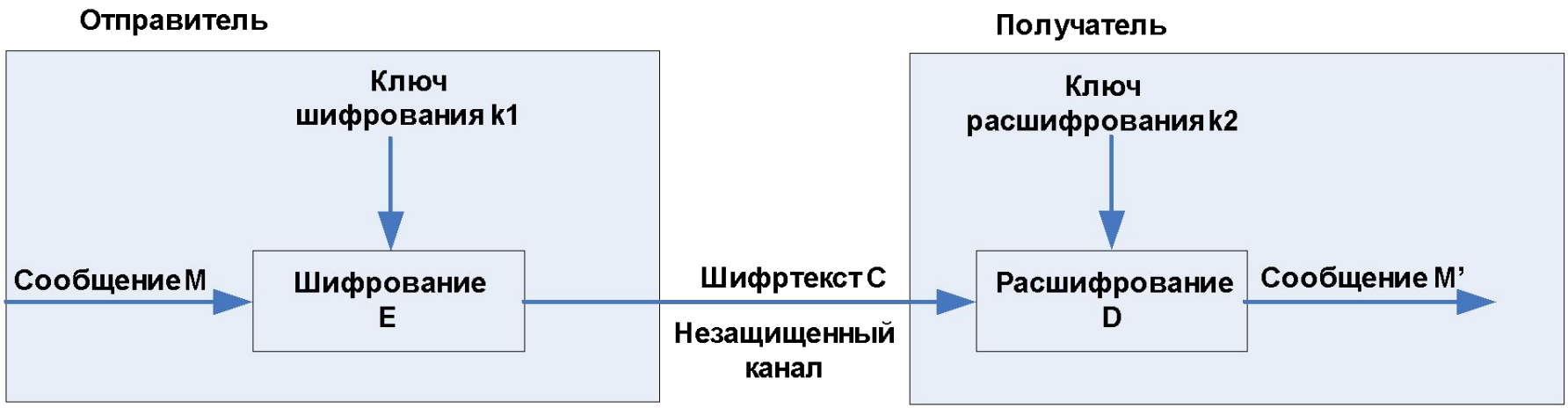
Реализация функций безопасности

- **Конфиденциальность** обеспечивается
  - алгоритмами симметричного и асимметричного шифрования
  - путем аутентификации абонентов на основе многоразовых и одноразовых паролей, цифровых сертификатов, смарт-карт и т.п.
- **Целостность и подлинность** достигается с помощью
  - различных вариантов технологии электронной подписи, основанных на односторонних функциях и асимметричных методах шифрования.
- **Аутентификация** - разрешается устанавливать соединения только между легальными пользователями и предотвращает доступ к средствам сети нежелательных лиц
  - асимметричные алгоритмы, ЭЦП, хэширование
  - парольная и биометрическая аутентификация

Определения

- **Шифр** – совокупность процедур и правил криптографических преобразований, используемых для зашифрования и расшифрования информации с использованием ключа шифрования
- **Зашифрование** информации – процесс преобразования открытой информации в зашифрованный текст
- **Расшифровани**е – процесс восстановления исходного текста по криптограмме с использованием ключа шифрования

Обобщенная схема криптосистемы шифрования

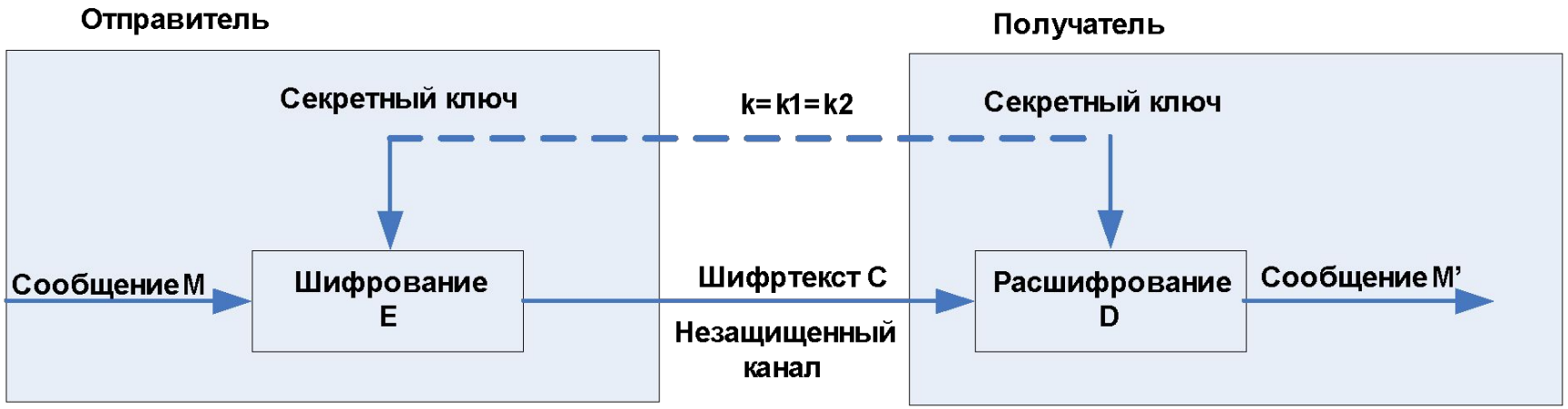


- Шифрование: $C = E_{k_1}(M)$
- Расшифрование: $M' = D_{k_2}(C)$
- D – функция, обратная к E
- *$к_2$ однозначно соответствует $к_1$*

Классификация криптосистем

- По наличию ключа
  - Ключевые криптосистемы
  - Бесключевые криптосистемы
- По виду используемого алгоритма
  - Симметричные криптосистемы
  - Асимметричные криптосистемы

Симметричное шифрование



- $к_1 = к_2$ (один ключ математически легко вычисляется из другого)

Виды симметричного шифрования

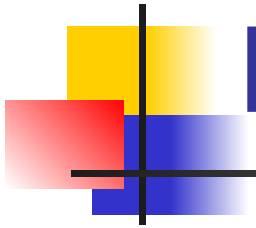
- Блочное шифрование
  - данные бьются на блоки фиксированной длины (64 или 128 бит)
  - существуют разные режимы шифрования блоков
- Потоковое шифрование
  - шифруются биты данных
  - используется при невозможности разбить данные на блоки

Преимущества симметричных алгоритмов

- По сравнению с асимметричными алгоритмами:
  - Работают быстрее
  - Более безопасные при одном и том же размере ключа
- Симметричные и асимметричные алгоритмы используются для решения разных задач

Примеры симметричных алгоритмов

Алгоритм	Описание
DES	Digital Encryption Standard (стандарт симметричного шифрования), блочный шифр
3DES	тройной DES, более сильная альтернатива DES
Rijndael	блочный шифр, пришел на смену DES
RC2	шифр изобретенный Рональдом Ривестом, блочный шифр



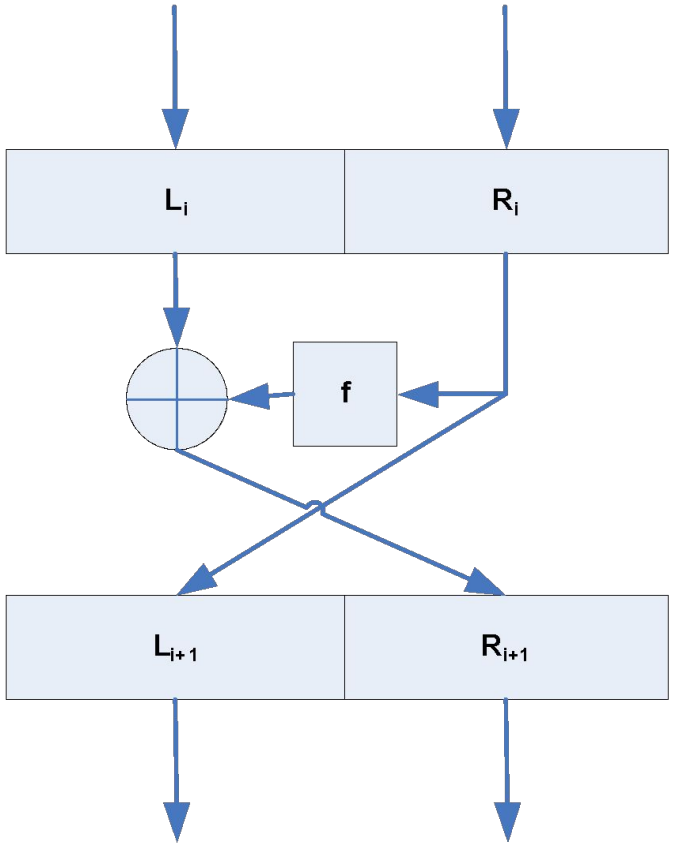
DES

- Разработчик - Horst Feistel (IBM)
- В 1977 г. DES был принят в США в качестве стандарта шифрования конфиденциальных (не строго секретных) данных
- Сейчас не считается полностью безопасным, т.к. несколько раз был взломан публично
- Временная рекомендация – использование более сильной модификации «тройной DES»
- В конце 1990-х в США принят алгоритм Rijndael в качестве стандарта симметричного шифрования AES (Advanced Encryption Standard)

Характеристики DES

- DES – симметричный блочный шифр
  - преобразует 64-битовые блоки данных
  - при помощи 56-битового секретного ключа
  - преобразование включает в себя 16 циклов перестановок и подстановок
- Подстановки усложняют связь между открытым и шифрованным текстами
- Транспозиции дают более равномерное распределение данных по шифрблоку - затрудняет обнаружение статистических закономерностей
- Дополнение последнего неполного блока до 64 бит

Цикл DES



- 64 битовый блок разбивается на 2 32-битовых полублока
- Правый полублок шифруется f , ключ - подмножество битов из 56-битового ключа
- Зашифрованный правый полублок объединяется XOR с левым полублоком
- Результат становится новым правым полублоком для следующего цикла.
- В левый полублок для следующего цикла подставляется прежний правый полублок

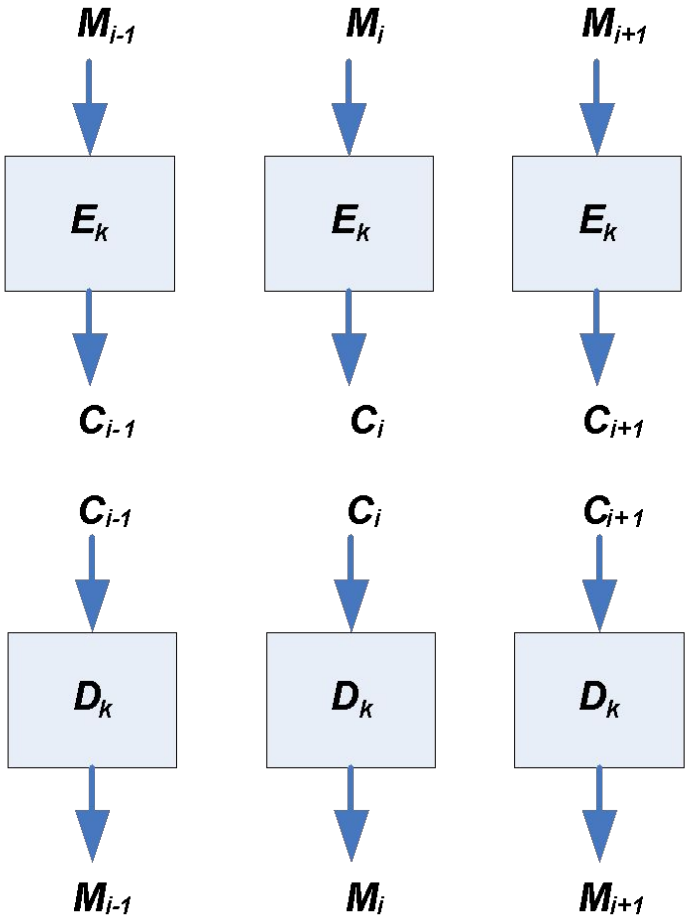
Операционные режимы

Режим	Аббревиатура
Электронная шифровальная книга (Electronic Codebook)	**ECB**
Сцепление шифрованных блоков (Cipher Block Chaing)	**CBC**
Шифрованная обратная связь (Cipher Feedback)	**CFB**
Обратная связь по выходу (Output Feedback)	**OFB**
Проскальзывание шифрованного текста (Cipher Text Stealing)	**CTS**

Режим ECB

- 16 циклов применяются к каждому очередному блоку данных индивидуально
- ошибка в одном блоке не распространяется на последующие блоки
- возможна параллельная обработка блоков
- слабая криптостойкость
- можно скомпилировать «шифровальную книгу»
- хорошо подходит для шифрования ключей

Схема режима ECB



- Шифрование

$$C_i = E_k(M_i)$$

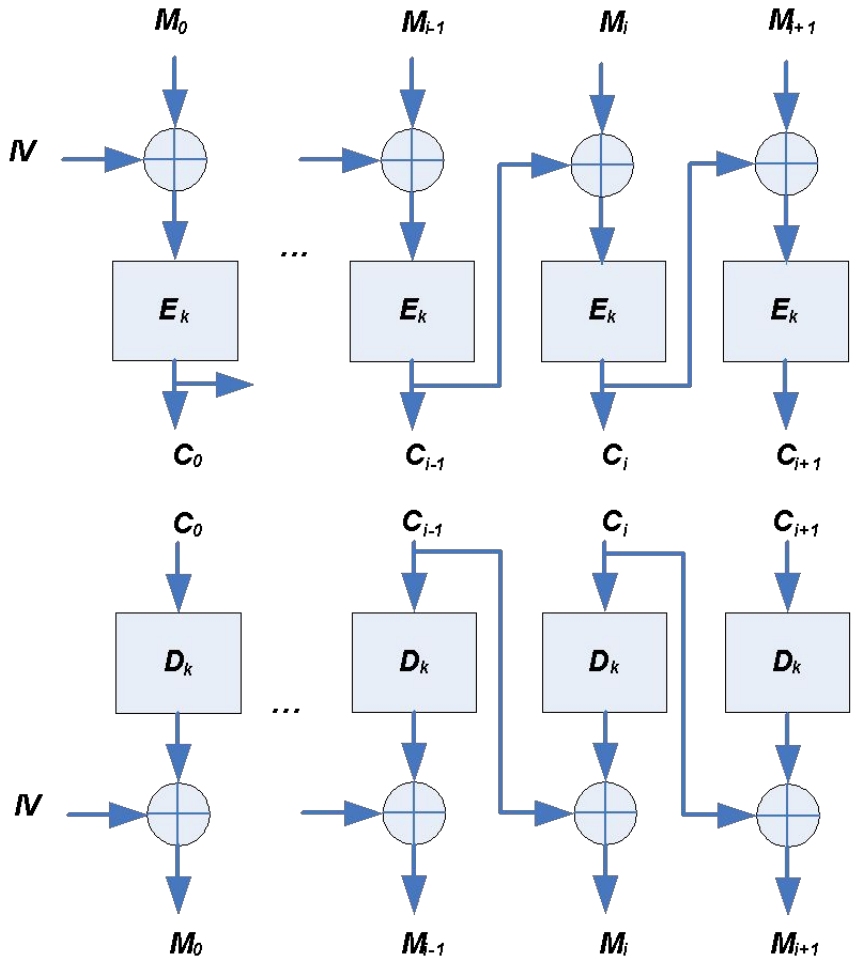
- Дешифрование

$$M_i = D_k(C_i)$$

Режим CBC

- Более защищенная технология, не позволяющая создать «шифровальную книгу»
- Перед началом 16 циклов каждый блок открытого текста суммируется XOR с предыдущим зашифрованным блоком
- Первый блок суммируется со случайным 64-битовым вектором инициализации IV
- Пригоден для аутентификации данных

Схема режима CBC



- Шифрование

$$C_0 = E_k(M_0 \oplus IV)$$

$$C_i = E_k(M_i \oplus C_{i-1})$$

- Дешифрование

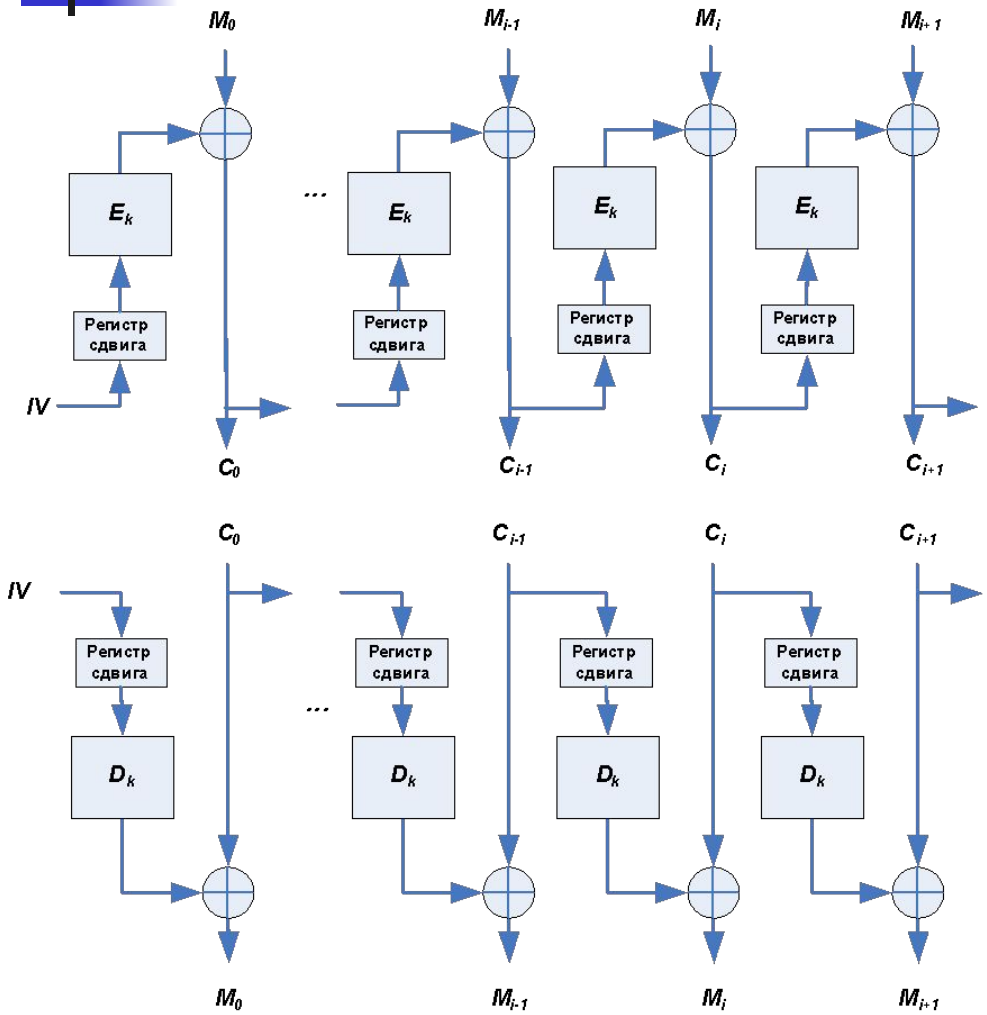
$$M_0 = D_k(C_0) \oplus IV$$

$$M_i = D_k(C_i) \oplus C_{i-1}$$

Режимы CFB и OFB

- Оперируют данными, меньшими стандартного 64-битового блока
- Применимы для потоковых шифров
- Шифрование очередного блока зависит от предыдущего блока, применяется вектор инициализации
- Затруднен криптоанализ
- OFB – потеря бита при передаче испортит все последующие блоки
- OFB применяется для шифрования в спутниковых системах связи
- CFB - самосинхронизируется – при порче блока правильность расшифровки будет восстановлена в следующем блоке
- CFB предназначен для шифрования отдельных символов, пригоден для аутентификации данных

Схема режима CFB



- Шифрование

$$C_0 = M_0 \oplus E_k(IV)$$

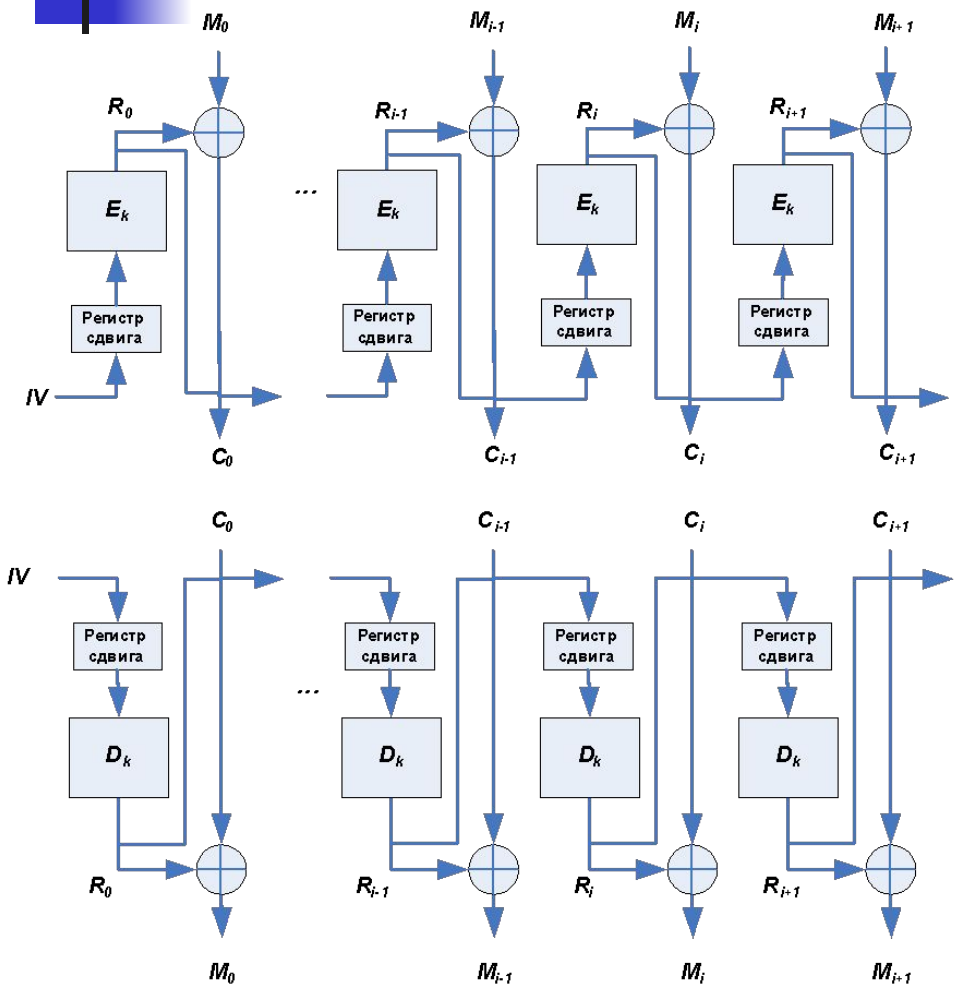
$$C_i = M_i \oplus E_k(C_{i-1})$$

- Дешифрование

$$C_i = M_i \oplus E_k(C_{i-1})$$

$$M_i = C_i \oplus D_k(C_{i-1})$$

Схема режима OFB



- Шифрование

$$C_0 = M_0 \oplus E_k(IV)$$

$$C_i = M_i \oplus E_k(R_{i-1}),$$

- Дешифрование

$$M_0 = C_0 \oplus D_k(IV)$$

$$M_i = C_i \oplus D_k(R_{i-1})$$

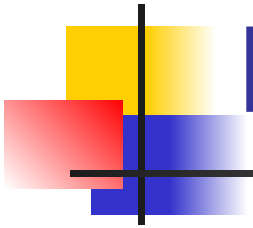
Тройной DES

- Временная альтернатива DES
- Каждый 64-битовый блок шифруется 3 раза алгоритмом DES с 3 ключами (56-бит)
- Шифрование тройным DES:

$$C = E_{k_3}(D_{k_2}(E_{k_1}(M)))$$

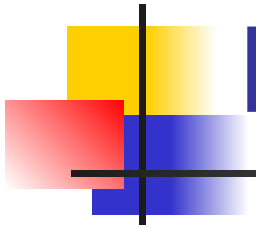
- Дешифрование тройным DES:

$$M = D_{k_1}(E_{k_2}(D_{k_3}(C)))$$



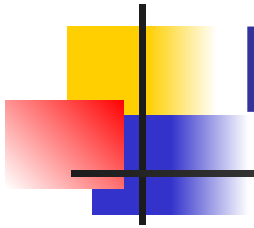
Rijndael

- Преемник DES
- Предназначен для защиты чувствительной, но не секретной информации
- Размер ключа нефиксированный (128, 192 и 256 бит)
- Используются 128, 192 и 256-битовые блоки данных
- Число циклов зависит от размеров ключа и блока:
  - размеры ключа и блока одновременно =128 битам-> 9 циклов
  - размер ключа и блока >128 и один из них <= 192-> 11 циклов
  - ключ либо блок данных = 256 бит-> 13 циклов
- Существенно сложнее DES:
  - алгебра полиномов с коэффициентами над полями Галуа GF(28)



RC2

- RC2 – зарегистрированная торговая марка RSA Data Security, Incorporated
- Блоковый шифр, разработанный Рональдом Ривестом в 1987
- Блоки входных данных 64 бита
- Ключ переменной длины (от 1 до 128 байт) – управление криптостойкостью
- Скорость работы RC2 в 2 раза выше DES
- Лицензирован для Lotus Notes, MS Internet Explorer, Outlook Express и Netscape Communicator и т.д.



Криптография в .NET

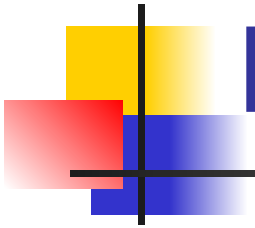
- Пространство имен System.Security.Cryptography
- Абстрактный класс для симметричных алгоритмов SymmetricAlgorithm
- Производные классы:
  - DES
  - TripleDES
  - Rijndael
  - RC2

Некоторые свойства SymmetricAlgorithm

Публичное свойство	Описание
IV	Возвращает или задает вектор инициализации для режима CBC (byte [])
Key	Возвращает или задает секретный ключ для шифрования/дешифрования (byte [])
Mode	Возвращает или задает операционный режим. Перечисляемый тип CipherMode=(ECB, CBC, CFB, OFB, CTS). CTS – вариация CBC
Padding	Возвращает или задает режим дополнения последнего неполного блока до стандартного размера. Перечисляемый тип PaddingMode=(PKCS7, Zeros, None): •PKCS7 – байты дополнения, равные общему числу таких байтов •Zeros – дополнение нулями •None – отсутствие дополнения (алгоритм шифрования использует свою специальную схему дополнения, игнорируя данное свойство)

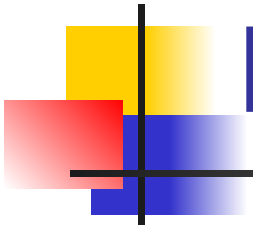
Некоторые методы SymmetricAlgorithm

Публичный метод	Описание
Create	Перегруженный статический метод для создания объекта, производного от SymmetricAlgorithm
GenerateKey	Генерирует случайным образом ключ шифрования (byte []). Используется генератор псевдослучайных чисел RandomNumberGenerator, производящий криптографически сильную последовательность случайных значений, а не класс Random
GenerateIV	Генерирует случайный вектор инициализации (byte [])
CreateEncryptor	Перегруженный статический метод для создания объекта Encryptor обеспечивающего реализацию интерфейса ICryptoTransform, использующегося для трансформации блоков данных



Параметры алгоритмов .NET

- ECB, CBC, CBF - только для DES, 3DES, RC2
- Для Rijndael доступны ECB, CBC
- OFB, CTS не реализованы ни для одного алгоритма
- Режимы дополнения данных (PKCS7, Zero, None) также не для всех алгоритмов/режимов работы алгоритмов
  - Например - для Rijndael не применимы Zero, None



Криптографические потоки

- Поддержка CLR криптографических функций, ориентированных на потоки
- CryptoStream - позволяет записывать и читать данные через криптографический поток (как через файл или сокет)
- CryptoStream можно использовать для шифрования (режим записи) или дешифрования (режим чтения)
- Использование в качестве основного поточного класса (операции ввода-вывода) - MemoryStream, FileStream и т.д.

Пример шифрования по методу DES в .NET

```
byte[] cipherbytes;          byte[] Key;          byte[] IV;
SymmetricAlgorithm sa = DES.Create();
sa.GenerateKey();          Key = sa.Key;
sa.GenerateIV();           IV = sa.IV;
sa.Mode = CipherMode.ECB;
sa.Padding = PaddingMode.PKCS7;
MemoryStream ms = new MemoryStream();
CryptoStream cs = new CryptoStream(ms, sa.CreateEncryptor(),
CryptoStreamMode.Write);
byte[] plainbytes = Encoding.UTF8.GetBytes("Произвольный текст");
cs.Write(plainbytes, 0, plainbytes.Length);
cipherbytes = ms.ToArray();
cs.Close();                ms.Close();
Console.WriteLine(Encoding.UTF8.GetString(cipherbytes));
```

Пример дешифрования по методу DES в .NET

```
SymmetricAlgorithm sa = DES.Create();
sa.Key = Key;
sa.IV = IV;
sa.Mode = CipherMode.ECB;
sa.Padding = PaddingMode.PKCS7;
MemoryStream ms = new MemoryStream(cipherbytes);
CryptoStream cs = new CryptoStream(
                    ms,
                    sa.CreateEncryptor(),
                    CryptoStreamMode.Read);

byte[] plainbytes = new Byte[cipherbytes.Length];
cs.Read(plainbytes, 0, cipherbytes.Length);
cs.Close();
ms.Close();
Console.WriteLine(Encoding.UTF8.GetString(plainbytes));
```