Лабораторная работа №2

Шифрование с имитовставкой

Цель работы: ознакомится с принципами работы шифрования с имитовставкой. Разработать консольное приложение, использующее алгоритм HMAC для генерации имитовставки и осуществляющее шифрование и дешифрование файла с подтверждением целостности данных.

Задание:

Разработать консольное приложение, реализующее шифрование/расшифрование файла с имитовставкой. В работе использовать:

- симметричный алгоритм шифрования AES-256 в режиме CFB;
- алгоритм генерации имитовставки HMAC;
- хэш-функцию SHA-256.

Алгоритм выполнения шифрования с имитовставкой:

1. На основе хеша полученных от пользователя данных вырабатывается ключ:
    a. с помощью случайных данных генерируется “соль”;
    b. формируется ключ
        i. вариант 1:
            1. “соль” конкатенируется с парольной фразой пользователя;
            2. берется хэш от полученных данных: *SHA256(salt + password)*.
        ii. вариант 2: PBKDF2
2. Случайным образом вырабатывается инициализирующий вектор.
3. Одновременно с шифрованием файла вырабатывается имитовставка по алгоритму HMAC от открытого текста.
4. Все необходимые для расшифрования данные записываются в файл подряд без разделителей в бинарном виде. Формат файла следующий:
   [соль(32Б)][HMAC(32Б)][инициализирующий вектор(16Б)][шифротекст]
5. Производится расшифрование данных из файла, выводится результат проверки HMAC.

Для генерации “соли” можно использовать свой метод или один из следующих:

- накопление последних бит кодов нажимаемых пользователем кнопок;
- накопление последних бит от координат пикселя, находящегося под курсором пользователя;
- накопление последних бит от значений одного или нескольких цветовых каналов пикселя, находящегося под курсором.

Требования к консольному приложению

Консольное приложение должно иметь два режима:

- шифрование файла;
- расшифрование файла с последующим подтверждением целостности данных.

Консольное приложение должно принимать на вход следующие аргументы:

- имя файла для шифрования/расшифрования;
- имя файла для сохранения зашифрованного/расшифрованного файла;

- *(опционально)* отключение проверки целостности, по умолчанию проверка целостности включена.

Приветствуется использование аргументов командной строки для передачи вышеописанных параметров. Наличие интерактивного режима не обязательно - при его отсутствии или при запуске с неверными аргументами приложение должно показывать инструкцию по использованию.

После успешного запуска в любом из режимов приложение должно сгенерировать ключ шифрования. Для этого необходимы следующие компоненты:

- парольная фраза. Пользователь должен ввести парольную фразу в консоль, при этом необходимо использовать такие методы, как *console.readPassword* в JAVA, или специальные модули, например read в NodeJS;
- модификатор ключа. Пользователь должен предоставить программе случайные данные для генерации “соли” при шифровании файла. На этом этапе должен происходить сбор данных (пользователь перемещает курсор по экрану, нажимает случайные клавиши на клавиатуре и т.д.) до тех пор, пока “соль” не будет сгенерирована или пользователь не остановит выполнение программы. При расшифровании соль берется из предоставленного для расшифровки файла.

После того, как все необходимые данные для выполнения операции получены, необходимо зашифровать/расшифровать файл и в случае расшифрования произвести проверку целостности следующим образом:

- сгенерировать на основе имеющихся данных (расшифрованного текста, парольной фразы и модификатора ключа) имитовставку;
- сравнить с имитовставкой, предоставленной в файле.

В случае подтверждения факта искажения данных сохранять расшифрованный файл не обязательно.

О реализации приложения

Выбор языка не ограничен, приветствуется использование любых языков программирования - как молодых (NodeJS, Go, Rust), так и пожилых (C/C++, Java, C#, Python, Ruby). Большинство из них имеют встроенные модули или сторонние пакеты, предоставляющие интерфейсы к реализациям криптографических алгоритмов:

- Java Crypto Interface
- C# System.Security.Cryptography
- pycrypto
- NodeJS crypto
- Go crypto
- rust-crypto
- и т.д.

Если вам хотелось попробовать новый язык, но не было задачи - теперь она есть. Готов оказать помощь в разумных пределах и помочь найти ответы на интересные вопросы.

О шифровании с имитовставкой и процессе демонстрации работы

Текст лекции о шифровании с имитовставкой.

Процесс демонстрации следующий:

1. Один пользователь шифрует файл.
2. Опционально с помощью HEX-редактора содержимое файла изменяется.
3. Второй пользователь расшифровывает файл и подтверждает либо факт успешной передачи данных, либо факт их искажения.