РТУ МИРЭА

ЗАДАНИЕ

на практическое и лабораторное занятие

по учебной дисциплине

«Технологии обеспечения информационной безопасностью»

Занятия №№ 7.2, 7.3 Настройка и применение криптографических протоколов

Москва 2023

Перечень отрабатываемых учебных заданий:

1. Настройка GnuPG.
2. Управление ключами.
3. Настройка цифровых подписей.
4. Подготовка отчета.

Общие термины и определения криптографии

Криптография

В общем повседневном использовании криптография – это акт или искусство письма в секретных символах. В техническом жаргоне ее можно определить как науку использования математики для шифрования и расшифрования данных.

Криптоанализ – это изучение способов компрометации (поражения) криптографических механизмов. Это наука взлома кода, расшифровки секретов, нарушения схем аутентификации и, в общем, нарушения криптографических протоколов. Криптология¶

Криптология – это дисциплина, объединяющая криптографию и криптоанализ. Другими словами, **криптология** – это раздел математики, изучающий математические основы криптографических методов.

Шифрование преобразует данные в практически невозможную для чтения форму без соответствующих знаний (например, ключа). Его цель – обеспечить конфиденциальность, удерживая информацию скрытой от всех, для кого она не предназначена.

Расшифрование – это процесс, обратный шифрованию, который преобразует зашифрованные данные в понятную форму.

Метод шифрования и расшифрования называется **шифром**.

Хэш-функции (алгоритмы дайджеста)

Криптографические хэш-функции используются в различных контекстах, например, для вычисления дайджеста сообщения при создании цифровой подписи. Хэш-функция сжимает биты сообщения до хэш-значения фиксированного размера для равномерного распределения возможных сообщений среди возможных значений хэша. Криптографическая хэш-функция делает это таким образом, что крайне сложно придумать сообщение, которое бы хэшировалось в определенное значение хэша. Ниже приведены примеры наиболее известных и широко используемых хэш-функций.

а) SHA-1 (Secure Hash Algorithm). Это криптографический хэш-алгоритм, опубликованный правительством Соединенных Штатов. Он создает хэш-значение из произвольной строки фиксированной длины в 160 бит. Данный алго считается устойчивым.

б) MD5 (Message Digest Algorithm 5). Это криптографический хэш-алгоритм, разработанный в лабораториях RSA. Его можно использовать для хэширования произвольной строки байтов в значение размером 128 бит.

Криптографические алгоритмы

Существуют два класса алгоритмов с ключом:

а) Симметричные алгоритмы шифрования (секретный ключ)

Симметричные алгоритмы используют один и тот же ключ для шифрования и расшифрования (или ключ расшифрования легко вытекает из ключа шифрования). Алгоритмы с секретным ключом используют один и тот же ключ как для шифрования, так и для расшифрования (или один можно легко вывести из другого). Это более прямой подход к шифрованию данных, математически менее сложный, чем криптография с открытым ключом. Симметричные алгоритмы могут быть разделены на поточные шифры и блочные шифры. Поточные шифры могут шифровать один бит открытого текста за раз, тогда как блочные шифры берут несколько бит (обычно 64 бита в современных шифрах) и шифруют их как единицу. Симметричные алгоритмы намного быстрее выполнять на компьютере, чем асимметричные.

Примеры симметричных алгоритмов: AES, 3DES, Blowfish, CAST5, IDEA и Twofish.

б) Асимметричные алгоритмы (алгоритмы с открытым ключом)

Асимметричные алгоритмы используют разные ключи для шифрования и расшифрования, и ключ расшифрования не может быть выведен из ключа зашифрования. Асимметричные шифры позволяют сделать ключ шифрования общедоступным, позволяя любому зашифровать с ключом в то время, как только правильный получатель (который знает ключ расшифрования) может расшифровать сообщение. Ключ шифрования также называется открытым ключом, а ключ расшифрования - закрытым или секретным ключом.

RSA, вероятно, самый известный асимметричный алгоритм шифрования.

Цифровая подпись

Цифровая подпись привязывает документ к владельцу определенного ключа.

Цифровая подпись документа – это информация, основанная как на документе, так и на закрытом ключе подписанта. Обычно она создается через хэш-функцию и частную функцию подписи (шифрование с использованием закрытого ключа подписанта).

Цифровая подпись – это небольшое количество данных, созданное с использованием какого-то секретного ключа. При этом публичный ключ используется для проверки того, что подпись была создана с использованием соответствующего закрытого ключа.

Существует несколько методов создания и проверки цифровых подписей, но общеизвестен алгоритм с открытым ключом RSA.

Криптографические протоколы

Криптография работает на многих уровнях. На одном уровне у вас есть алгоритмы, такие как блочные шифры и криптосистемы с открытым ключом. На их основе вы получаете протоколы, и на основе протоколов вы находите

приложения (или другие протоколы). Вот список типичных приложений, использующих криптографические протоколы. Эти протоколы построены на криптографических алгоритмах более низкого уровня.

1) Безопасность сервера имен домена (DNSSEC)

DNSSEC, или система проверки подписей доменных имен (Domain Name System Security Extensions), представляет собой набор расширений для протокола DNS (Domain Name System). Она предназначена для обеспечения дополнительного уровня безопасности в работе DNS, которая играет ключевую роль в процессе преобразования доменных имен в IP-адреса. Основная цель DNSSEC – предотвращение подделки данных в системе DNS, таких как манипуляции с DNS-запросами и ответами. Система достигает этой цели с помощью добавления цифровых подписей к DNS-записям. Эти подписи могут быть проверены для подтверждения подлинности информации, предоставляемой DNS.

2) Протокол защищенного сокета (SSL)

SSL – это один из двух протоколов, используемых для безопасных подключений к сети Интернет (другой – SHTTP).

3) Протокол защищенного гипертекстового передачи протокола (SHTTP)

Протокол для обеспечения большей безопасности транзакций в сети Интернет.

4) Безопасность электронной почты и связанные службы

GnuPG – GNU Privacy Guard – соответствует предложенному интернет-стандарту OpenPGP, описанному в RFC2440.

5) Протокол SSH2

Протокол используется для обеспечения безопасности терминальных сессий и TCP-соединений.

Задание 1. Настройка GnuPG

GnuPG (GNU Privacy Guard) представляет собой набор программ для шифрования открытым ключом и создания цифровых подписей. Эти инструменты могут использоваться для шифрования данных и создания цифровых подписей. Кроме того, в состав входит средство управления ключами. GnuPG использует криптографию открытого ключа для обеспечения безопасной коммуникации между пользователями.

Выполните следующие упражнения от имени обычного пользователя, например, пользователя «user1».

Для создания новой пары ключей

1. Войдите в систему под пользователем «user1».

2. Убедитесь, что пакет GnuPG установлен в вашей системе.

Если он не установлен, установить его под суперпользователем.

3. Перечислите и запишите все скрытые каталоги в вашем домашнем каталоге.

4. Выведите список ключей, которые в данный момент есть в вашем ключевом хранилище.

5. Используйте gpg для генерации новых ключей.

Вариант по умолчанию – две пары ключей: DSA будет основной – для создания цифровых подписей, а второстепенная пара ключей ELGamel – для шифрования данных.

6. Создайте ключ ELG-E с размером 1024.

7. Создайте ключи, которые будут действительны в течение года.

8. Наберите «y» для указания срока истечения действия ключа.

9. Создайте идентификатор пользователя (User-ID) для вашего ключа:

Вам нужен идентификатор пользователя для вашего ключа; программа формирует идентификатор пользователя из поля «Real Name», «Comment» и «Email Address».

10. Выберите пароль для доступа к ключу.

Задание 2. Управление ключами

1. Под пользователем «user1», отобразите ключи в вашем хранилище.
2. Введите команду для того, чтобы убрать предупреждение о «insecure memory».
3. Выполните команду еще раз для отображения ваших ключей и убедитесь, что ваше изменение вступило в силу.
4. Выведите список ваших ключей вместе с их подписями.
5. Выведите только ваши секретные ключи.
6. Отобразите отпечатки ключей.

Создание отозванного сертификата (revocation certificate)

1. Находясь в системе под пользователем «user1», создайте отозванный сертификат. Он будет отображен в стандартном выводе.
2. Создайте отозванный сертификат, который будет сохранен в ASCII-формате в файле с именем «revoke.asc».

Экспорт открытых ключей

Для того, чтобы безопасно обмениваться информацией с другими людьми, используя криптосистему на основе открытого ключа, необходимо обменяться открытыми ключами или сделать свой открытый ключ общедоступным (на веб-страницах, серверах ключей и т.д.).

Для экспорта ваших открытых ключей:

1. Экспортируйте свой открытый ключ в бинарном формате в файл с именем «user1-pub.gpg».
2. Экспортируйте свой открытый ключ в файл с именем «user1-pub.asc», но в формате ASCII.
3. Используйте команду **cat**, чтобы просмотреть бинарную версию открытого ключа user1 (user1-pub.gpg).
4. Чтобы сбросить терминал введите: **reset**

5. Используйте команду **cat**, чтобы просмотреть ASCII-версию открытого ключа user1 (user1-pub.asc).

6. Вы заметите, что ASCII-версия более подходит для публикации на веб-страницах или использования в рассылках и т. д.

Задание 3. Настройка цифровых подписей

Создание и проверка подписей использует открытый/закрытый ключ, что отличается от процессов шифрования и дешифрования. Использование закрытого ключа подписанта для создания подписи упрощает проверку с использованием соответствующего открытого ключа.

Для создания цифровой подписи файла выполните следующие шаги:

1. Создайте файл с именем «secret-file.txt» и текстом «Hello All».

2. Используйте команду `cat` для просмотра содержимого файла. С помощью команды `file` определите тип файла.

3. Подпишите файл своей цифровой подписью. Введите пароль при запросе.

Эта команда создаст другой файл «secret1.txt.gpg», который сжат и имеет прикрепленную подпись. Запустите команду «file» для проверки этого. Просмотрите файл с помощью `cat`.

4. Проверьте подпись на подписанном файле «secret1.txt.gpg».

5. Создайте еще один файл «secret2.txt» с текстом «Hello All».

6. Подпишите файл «secret2.txt», но на этот раз сделайте файл в формате ASCII.

Должен быть создан файл в формате ASCII с именем «secret2.txt.asc» в вашем текущем каталоге.

7. Используйте команду `cat` для просмотра содержимого созданного файла в формате ASCII.

8. Создайте еще один файл с именем «secret3.txt» с текстом «hello world».

9. Добавьте свою подпись к содержимому файла.

Запишите команду для проверки подписи файла.

10. Откройте файл для просмотра содержимого с помощью любого просмотрщика.

Можете ли вы прочитать текст, который вы ввели в файл?

Задание 4. Подготовка отчета

Подготовить отчет со скриншотами и пояснением порядка выполнения заданий в файле Фамилия_ИО_7.2.docx.

Отчет по результатам выполнения заданий предоставить до 28.12.2023.