ЛЕКЦИЯ №6

«Криптографическая защита в современных операционных системах»

по дисциплине

«Безопасность операционных систем»

Текст лекции рассмотрен и одобрен на заседании кафедры протокол № _____ от " " 201__ г.

(Слайд 1. Титульный слайд)

Уважаемые студенты! Сегодня вы продолжаете изучение дисциплины «Безопасность операционных систем». Лекция №7 «Криптографическая защита в современных операционных системах», за ней одноименное практическое занятие. Продолжительность лекции - два академических часа, практического занятия - 2 академических часа.

Слайд 2 (вопросы занятия)

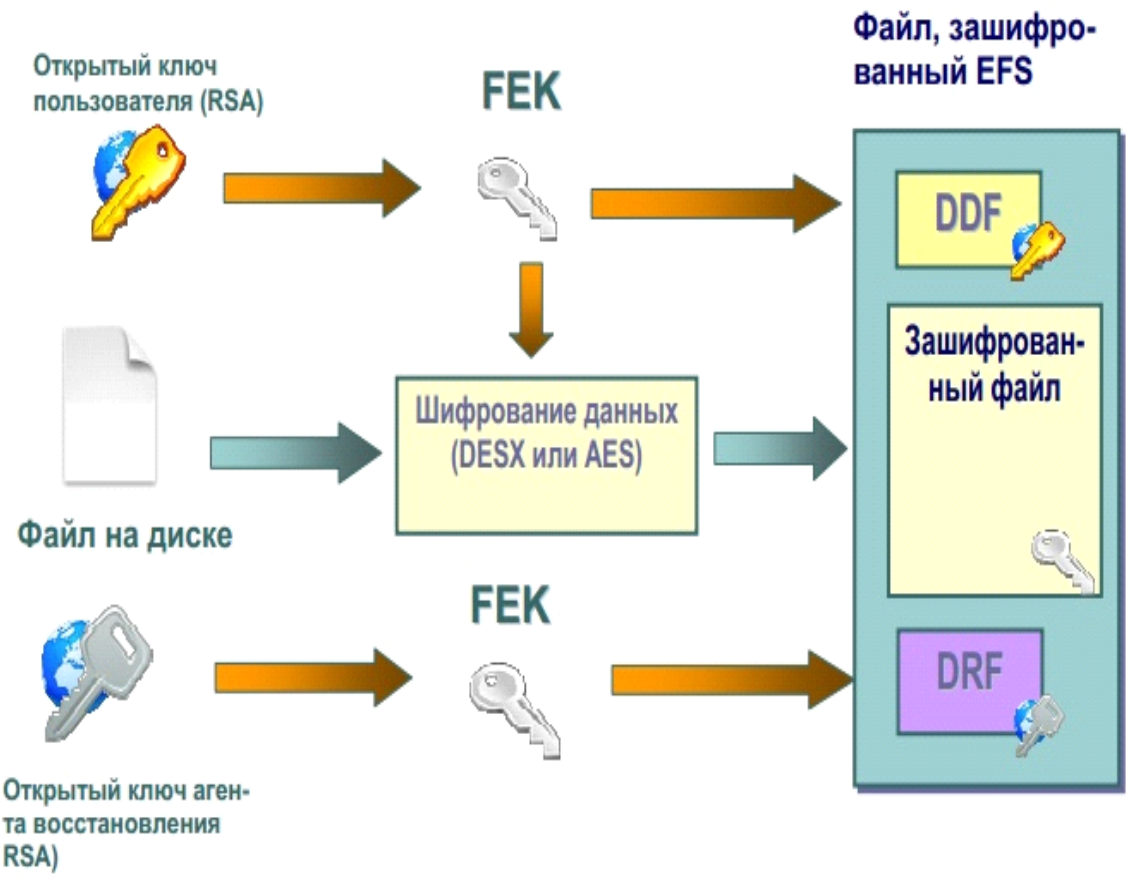
- Шифрование пользовательских данных в современных ОС.
- Аппаратное шифрование в современных ПК и серверах (TPM).
- Шифрование компонентов ОС.

Шифрование в Windows

Начиная с версии Windows 2000, в операционных системах семейства Windows NT поддерживается шифрование данных на разделах файловой системы NTFS с использованием шифрующей файловой системы (**Encrypted File System, EFS**). Основное ее достоинство заключается в обеспечении конфиденциальности данных на дисках компьютера за счет использования надежных симметричных алгоритмов для шифрования данных в реальном режиме времени.

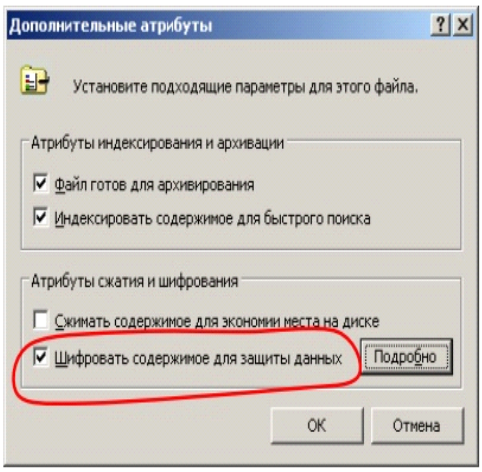
Для шифрации данных EFS использует симметричный алгоритм шифрования (AES или DESX) со случайным ключом для каждого файла **(File Encryption Key, FEK**). По умолчанию данные шифруются в Windows 2000 и Windows XP по алгоритму DESX, а в Windows XP с Service Pack 1 (или выше) и Windows Server 2003 — по алгоритму AES. В версиях Windows, разрешенных к экспорту за пределы США, драйвер EFS реализует 56-битный ключ шифрования DESX, тогда как в версии, подлежащей использованию только в США, и в версиях с пакетом для 128-битного шифрования длина ключа DESX равна 128 битам. Алгоритм AES в Windows использует 256-битные ключи.

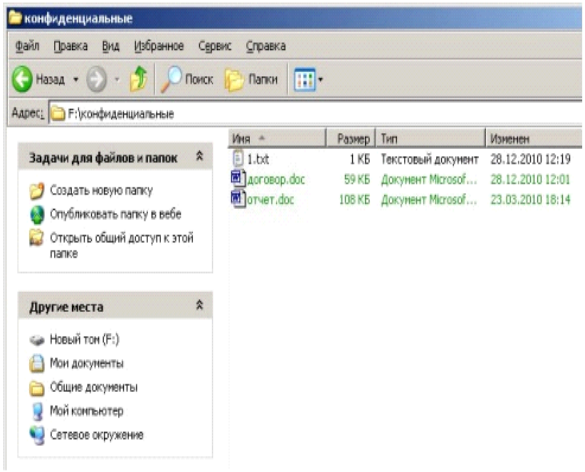
При этом для обеспечения секретности самого ключа FEK шифруется асимметричным алгоритмом RSA открытым ключом пользователя, результат шифрации **FEK** – **Data Decryption Field, DDF** – добавляется в заголовок зашифрованного файла. Такой подход обеспечивает надежное шифрование без потери эффективности процесса шифрования: данные шифруются быстрым симметричным алгоритмом, а для гарантии секретности симметричного ключа используется асимметричный алгоритм шифрования.



Для шифрования файлов с использованием EFS можно использовать графический интерфейс или команду **cipher**.

Графический интерфейс доступен в стандартном окне свойств объекта по нажатию кнопки «Дополнительно». Зашифрованные объекты в стандартном интерфейсе Windows Explorer отображаются зеленым цветом.





Необходимо отметить, что EFS позволяет разделять зашифрованный файл между несколькими пользователями. В этом случае FEK шифруется открытыми ключами всех пользователей, которым разрешен доступ к файлу, и каждый результат шифрования добавляется в DDF. Шифрование файла с использованием EFS защищает файл комплексно: пользователю, не

имеющему права на расшифрование файла, недопустимы, в том числе, такие операции, как удаление, переименование и копирование файла. Необходимо помнить, что EFS является частью файловой системы NTFS, и в случае копирования защищенного файла авторизованным пользователем на другой том с файловой системой, на поддерживающей EFS (например, FAT32), он будет расшифрован и сохранен на целевом томе в открытом виде.

Консольная команда cipher может быть использована для зашифрования/расшифрования файлов из командной строки или в bat-сценарии.

cipher [{/e|/d}] [/s:каталог] [/a] [/i] [/f] [/q] [/h] [/k] [/u[/n]] [путь [...]] |[/r:имя]

/e	Шифрует указанные папки. Папки помечаются таким образом, чтобы файлы, которые будут добавляться в папку позже, также шифровались.
/d	Расшифровывает указанные папки. Папки помечаются таким образом, чтобы файлы, которые будут добавляться в папку позже, не будут шифроваться
/s: ***каталог***	Выполняет выбранную операцию над указанной папкой и всеми подпапками в ней.
/a	Выполняет операцию над файлами и каталогами
/i	Продолжение выполнения указанной операции даже после возник-

	новения ошибок. По умолчанию выполнение **cipher** прекращается после возникновения ошибки
/f	Выполнение повторного шифрования или расшифровывания указанных объектов. По умолчанию уже зашифрованные или расшифрованные файлы пропускаются командой **cipher**
/k	Создание ключа шифрования файла для пользователя, выполнившего команду **cipher**. Если используется данный параметр, все остальные параметры команды **cipher** не учитываются.
/u	Обновление ключа шифрования файла пользователя или ключа агента восстановления на текущие ключи во всех зашифрованных файлах на локальном диске (если эти ключи были изменены). Этот параметр используется только вместе с параметром **/n**.
/n	Запрещение обновления ключей. Данный параметр служит для поиска всех зашифрованных файлов на локальных дисках. Этот параметр используется только вместе с параметром **/u**.
путь	Указывает шаблон, файл или папку.
/r: ***имя_файла***	Создание нового сертификата агента восстановления и закрытого ключа с последующей их записью в файлах с именем, указанным в параметре *имя_файла* (без расширения). Если используется данный параметр, все остальные параметры команды **cipher** не учитываются.

Например, чтобы определить, зашифрована ли какая-либо папка, необходимо использовать команду:

cipher путь\имя_папки

Команда cipher без параметров выводит статус (зашифрован или нет) для всех объектов текущей папки.

Для шифрования файла необходимо использовать команду
cipher /e /a путь\имя_файла

Для расшифровки файла, соответственно, используется команда
cipher /d /a путь\имя_файла

Допустима шифрование/расшифрование группы файлов по шаблону:
cipher /e /a d:\work*.doc

Пара открытый и закрытый ключ для шифрования FEK создаются для пользователя автоматически при первой шифрации файла с использованием EFS.

Если некоторый пользователь или группа пользователей зашифровали файл с использованием EFS, то его содержимое доступно только им. Это приводит к рискам утери доступа к данным в зашифрованных файлах в случае утраты пароля данным пользователем (работник забыл пароль, уволился и т.п.). Для предотвращения подобных проблем администратор может определить некоторые учетные записи в качестве агентов восстановления.

Агенты восстановления (Recovery Agents) определяются в политике безопасности **Encrypted Data Recovery Agents** (Агенты восстановления шифрованных данных) на локальном компьютере или в домене. Эта политика доступна через оснастку **Групповая политика (gpedit.msc)** раздел **«Параметры безопасности»->«Политика открытого ключа»->«Файловая система EFS»**. Пункт меню «Действие»-> «Добавить агент восстановления данных» открывает мастер добавления нового агента.

Добавляя агентов восстановления можно указать, какие криптографические пары (обозначенные их сертификатами) могут использовать эти агенты для восстановления шифрованных данных. Сертификаты для агентов восстановления создаются командой cipher с ключом /r. Для пользователя, который будет агентом восстановления, необходимо импортировать закрытый ключ агента восстановления из сертификата, созданного командой cipher. Это можно сделать в мастере импорта сертификатов, который автоматически загружается при двойном щелчке по файлу *.pfx.

EFS создает – DRF (Data Recovery Field)-элементы ключей для каждого агента восстановления, используя провайдер криптографических сервисов, зарегистрированный для EFS-восстановления. DRF добавляется в зашифрованный файл и может быть использован как альтернативное средство извлечения FEK для дешифрации содержимого файла.

Windows хранит закрытые ключи в подкаталоге Application Data\Microsoft\Crypto\RSA каталога профиля пользователя. Для защиты закрытых ключей Windows шифрует все файлы в папке RSA на основе симметричного ключа, генерируемого случайным образом; такой ключ называется мастер-ключом пользователя. Мастер-ключ имеет длину в 64 байта и создается стойким генератором случайных чисел. Мастер-ключ также хранится в профиле пользователя в каталоге Application Data\Microsoft\Protect и зашифровывается по алгоритму 3DES с помощью ключа, который отчасти основан на пароле пользователя. Когда пользователь меняет свой пароль, мастер-ключи автоматически расшифровываются, а затем заново зашифровываются с учетом нового пароля.

Для расшифровки FEK EFS использует функции Microsoft CryptoAPI (CAPI). CryptoAPI состоит из DLL провайдеров криптографических сервисов (cryptographic service providers, CSP), которые обеспечивают приложениям доступ к различным криптографическим сервисам (шифрованию,

дешифрованию и хэшированию). EFS опирается на алгоритмы шифрования RSA, предоставляемые провайдером Microsoft Enhanced Cryptographic Provider (\Windows\ System32\Rsaenh.dll).

Шифрование и расшифрование файлов можно осуществлять программно, используя API-функции EncryptFile и DecryptFile.

Шифрование в Linux

В Linux присутствуют различные системы шифрования файлов и разделов, например:

- LUKS
- TrueCrypt
- eCryptFS
- EncFS

Кроме того, для шифрования возможно использовать библиотеку OpenSSL.

LUKS.

LUKS (The Linux Unified Key Setup) — это спецификация шифрования дисков, созданная Clemens Fruhwirth, и первоначально предназначавшаяся для ОС Linux.

LUKS определяет платформо-независимый стандарт для использования в различных инструментах. Это не только облегчает совместимость и способность к взаимодействию различного программного обеспечения, но также гарантирует, что ПО осуществляют задокументированный и безопасный метод управления паролями.

Реализация LUKS существует для Linux и основана на расширенной версии cryptsetup, используя dm-crypt как вычислительную машину для дискового шифрования. Под Microsoft Windows зашифрованные диски LUKS могут использоваться с FreeOTFE.

В отличие от других решений LUKS сохраняет всю необходимую информацию в заголовке раздела и предоставляет пользователю интерфейс для управления этими данными.

LUKS с dm-crypt очень удобен для шифрования разделов диска, он позволяет иметь несколько паролей для одного раздела, а так-же с легкостью менять их.

Как создать зашифрованный раздел

```
# dd if=/dev/urandom of=/dev/sdc1           # Опционально. Только для параноиков
# cryptsetup -y luksFormat /dev/sdc1        # Это уничтожит все данные на sdc1
# cryptsetup luksOpen /dev/sdc1 sdc1
# mkfs.ext3 /dev/mapper/sdc1                 # Будет создана файловая система ext3
# mount -t ext3 /dev/mapper/sdc1 /mnt
# umount /mnt
# cryptsetup luksClose sdc1                  # Отсоединить зашифрованный раздел
```

Монтировать

```
# cryptsetup luksOpen /dev/sdc1 sdc1
# mount -t ext3 /dev/mapper/sdc1 /mnt
```

Размонтировать

```
# umount /mnt
# cryptsetup luksClose sdc1
```

Используются алгоритмы доступные в ядре ОС.

Для шифрования дисков в операционной системе FreeBSD присутствуют модули gbde и geli. Geli более быстрый т.к использует аппаратное ускорение.

Создание LUKS-контейнера:
touch /var/db/file
shred -n1 -s500M /var/db/file
modprobe dm-crypt
modprobe aes
losetup /dev/loop0 /var/db/file

cryptsetup -c aes -y create mycrypt /dev/loop0
или
cryptsetup -c aes --verify-passphrase luksFormat /dev/loop0

mkreiserfs /dev/mapper/mycrypt
losetup /dev/loop0 /var/db/file

cryptsetup create mycrypt /dev/loop0
или
cryptsetup luksOpen /dev/loop0 mycrypt

mount /dev/mapper/mycrypt /home/mnt
…
umount /dev/mapper/mycrypt /home/mnt

cryptsetup remove mycrypt
losetup -d /dev/loop0

EncFS

Свободная шифрованная файловая система, основанная на FUSE.

В EncFS существует исходная директория и точка монтирования. В исходной папке все файлы шифрованные, а в точке монтирования осуществляется представление файлов в расшифрованном виде. В исходной папке зашифрованы в том числе имена файлов. Ключ шифрования располагается в зашифрованном виде в исходной папке. Для расшифрования ключа используется пароль от пользователя.

Размер блока по умолчанию - 1024.

Используемые алгоритмы - Blowfish и AES.

eCryptFS

Аналогично EFS в Windows работает поверх другой файловой системы и прозрачно шифрует/расшифровывает содержимое файлов. Криптографические метаданные хранятся в заголовках каждого файла.

Поддержка реализована на уровне ядра, поэтому нет необходимости в FUSE.

Проверка наличия модуля ядра:
modprobe ecryptfs && lsmod | grep ecryptfs

Установка утилит для работы с ecryptfs
apt-get install ecryptfs-utils

Создание папки для размещения шифрованных файлов
mkdir /mnt/ecryptfs-data

Включение шифрования
mount -t ecryptfs /mnt/ecryptfs-data/ /mnt/ecryptfs-data/

После этого будет запрошен пароль, алгоритм шифрования, размер ключа, шифровать или нет имена файлов.

TrueCrypt

Изучался ранее на дисциплине ОБПИТИС. Более не поддерживается. Существует два проекта VeraCrypt и CipherShed, основанные на кодовой базе TrueCrypt.

Существует также консольная версия TrueCrypt для Unix-подобных ОС - tcplay.

Поддерживает шифры AES, Serpent и Twofish, а также их комбинации и три типа хешей - RIPEMD-160, SHA-512 и Whirlpool.

Вывод

В данной лекции мы рассмотрели такой важный механизм безопасности как криптографическая защита. Изучили конкретные реализации механизмов шифрования в ОС Linux, Windows.